

Оглавление

ВВЕДЕНИЕ	3
1. ПОСТАНОВКА ЗАДАЧИ	4
2. ОПИСАНИЕ ПРЕДМЕТНОЙ ОБЛАСТИ	5
3. АКТУАЛЬНОСТЬ АВТОМАТИЗАЦИИ	6
4. ОПИСАНИЕ ПРОГРАММЫ	7
4.1. АЛГОРИТМИЧЕСКИЕ РЕШЕНИЯ	7
Клиент	7
Сервер.....	8
4.2. ОПИСАНИЕ ИНТЕРФЕЙСА ПРОГРАММЫ	10
Окно авторизации пользователя.....	10
Окно с информацией о программе и создателе	11
Окно с информацией о продажах	12
Окно создания новой продажи	13
Окно с информацией об автомобилях	14
Окно с информацией о клиентах	16
Окно с информацией о сотрудниках.....	17
4.3. СОСТАВ ПРИЛОЖЕНИЯ	18
База данных	18
Сервер.....	20
Клиент	21
5. НАЗНАЧЕНИЕ И СОСТАВ КЛАССОВ ПРОГРАММЫ	22
Сервер.....	22
Клиент	23
СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ	26

ВВЕДЕНИЕ

В настоящее время большое количество компаний в различных сферах переносят свою работу в цифровой вид. Так как с цифровой информацией работать намного удобнее. Обработать различную информацию в цифровом виде можно намного быстрее, чем в аналоговом. Кроме этого, с помощью компьютерных технологий возможно автоматизировать многие процессы, которые ранее приходилось делать тем или иным лицам.

Таким образом, практически в каждой компании используются компьютерные технологии для различных целей. К примеру, сейчас на автозаправках вы можете заправить свой автомобиль, не выходя из машины. Данная возможность стала доступна с помощью цифровых технологий. Можно скачать мобильное приложение данной заправки и произвести оплату, находясь в своём авто. Это один из тысячи примеров реализации компьютерных технологий.

В сфере продаж повсеместно используются различные информационные системы. Они помогают хранить информацию о деятельности компании в различных сферах. На самом деле данные технологии используются практически везде, так как они помогают сгруппировать много различной информации, которую имеется возможность автоматизировано обрабатывать, не тратя на это много времени и человеческих сил.

Разработаем информационно-справочную систему для вымышленного дилерского центра автомобилей «DeAuto» на языке Java, содержащий в себе 2 решения, которые будут взаимодействовать между собой с помощью REST архитектуры. А именно, серверная часть и клиентская часть, подробнее о реализации рассказано далее.

1. ПОСТАНОВКА ЗАДАЧИ

В соответствии с выбранной темой требуется разработать клиент-серверное решение с использованием библиотек Spring Boot для сервера и JavaFX для GUI клиента в виде пользовательских классов и таблиц для СУБД.

Со стороны сервера необходимо использовать ORM для связи Spring с СУБД, а также модель MVC для отдельного расположения контроллеров, сервисов и репозитория с логикой таблиц СУБД.

Со стороны клиента необходимо разработать несколько окон и логику переходов пользователей между ними. Также продумать их дизайн и расположение элементов интерфейса для взаимодействия с пользователем.

Решение должно выполнять следующие операции:

- Для информационной модели, основанной на БД, таблицы должны быть предварительно заполнены записями;
- Обновлять изменения источника данных в базе данных;
- Отображать в таблице данные предметной области;
- Обновлять изменения источника данных в базе данных;
- Загружать данные из БД;
- Фильтровать записи в БД, которые удовлетворяют введенному пользователем сложному критерию;
- Добавлять в БД новые объекты, удалять и редактировать их;
- Сортировать записи;

Программа не должна завершаться аварийным путём. Необходимо выводить сообщения с ошибками. Программа должна содержать комментарии, которые могут генерировать автоматически составляемую документацию при помощи инструмента JavaDoc.

2. ОПИСАНИЕ ПРЕДМЕТНОЙ ОБЛАСТИ

Дилерский центр автомобилей является предметной областью автоматизации. Логика работы заключается в том, что работник организации может просматривать, добавлять, видоизменять или удалять различную информацию о работе компании. В частности необходимо хранить информацию о продажах, работниках и всех автомобилях автосалона.

Со стороны сервера и клиента используются четыре сущности для представления данных:

- Автомобиль – сущность представления конкретного автомобиля, который может быть уже продан, либо же находится в наличии и доступен к покупке. Содержит всю необходимую информацию об автомобиле.
- Клиент – сущность представления конкретного клиента дилерского центра автомобилей. Содержит всю необходимую информацию о клиенте.
- Работник – сущность представления конкретного работника дилерского центра автомобилей. Содержит всю необходимую информацию о работнике, включая пароль и логин для входа в информационно-справочную систему.
- Продажа – сущность представления информации о конкретной продаже. Содержит все необходимые данные. Показывает, какой клиент купил тот или иной автомобиль, а также фиксирует работника, проводившего эту операцию.

Более подробно об атрибутах сущностей доступна информация в пункте «База данных» главы «СОСТАВ ПРИЛОЖЕНИЯ»

3. АКТУАЛЬНОСТЬ АВТОМАТИЗАЦИИ

Автоматизация работы дилерского центра автомобилей с помощью собственно разработанного решения позволяет хранить информацию о основных рабочих процессах компании. Позволяет сохранять информацию о каждом сотруднике и клиенте автосалона, а также хранить данные о продажах. Информация сохраняется в текстовом виде на своих серверах и базах данных, что может быть опасно для компаний, которым необходимо хранить конфиденциальную информацию. Запуск приложения клиента возможен на всех компьютерных информационных системах, а именно Windows, OS X и Linux.

Если серверная часть решения размещается на территории Российской Федерации, то программа будет исполнять ФЗ-152 «О защите персональных данных», что также важно для компаний, которые работают с данными пользователей РФ.

4. ОПИСАНИЕ ПРОГРАММЫ

4.1. АЛГОРИТМИЧЕСКИЕ РЕШЕНИЯ

Клиент

На рисунке 1 ниже представлены переходы пользователя различными формами с данными. На данной схеме не отображены окна с подтверждением операции и окнами с обработками различных ошибок. Данные окна используются повсеместно для подтверждения тех или иных действий, либо же отображения различных ошибок, связанных с проблемами подключения, неверным вводом данных от пользователя и других.

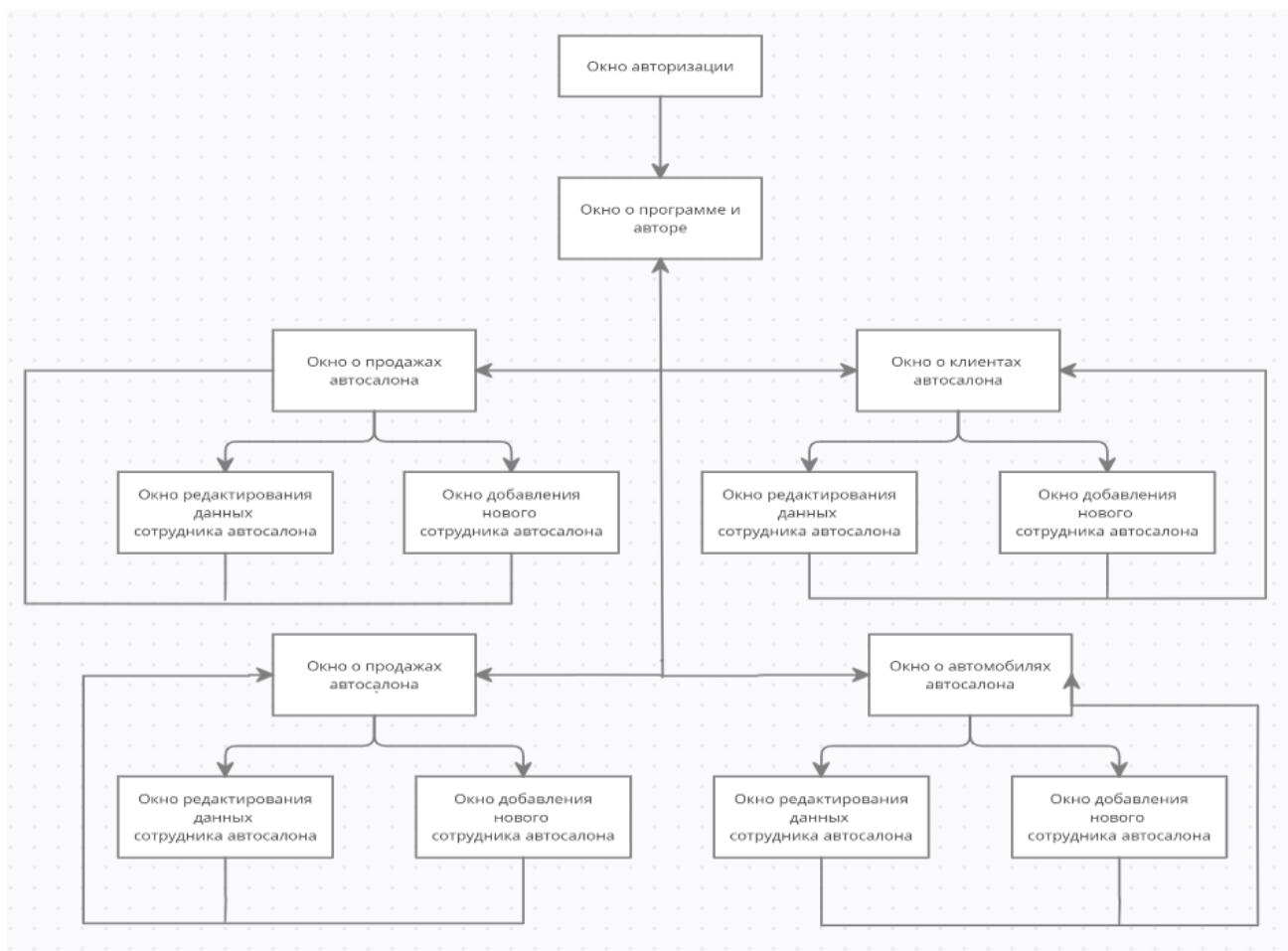


Рис. 1. Переходы пользователя между формами в программе-клиенте

При запуске программы пользователя предоставляется возможность ввода логина и пароля. После получения данных запрос с авторизацией отправляется на сервер. После получения ответа от сервера пользователь либо переходит к

работке с АИС автосалона, либо же выводится ошибка о неправильном пароле или логине.

После авторизации пользователь может перемещаться между окнами с информацией о продажах, клиентах, сотрудниках и автомобилях с помощью кнопок меню в левой части приложения. Точно также в любой момент пользователь может переключиться на окно с информацией о программе и её создателе.

При переходе пользователя на новое окно на сервер отправляется запрос для получения необходимых данных. Данные автоматически обновляются при изменении пользователем их. Сначала на сервер отправляется запрос на изменение/добавление/удаление информации, после получаем изменённые данные с сервера и отображаем их на клиенте.

Сервер

При получении запроса от клиента, сервер сопоставляет переданный путь URL с прописанными контроллерами, либо обрабатывается с помощью Spring Boot Data Rest. Подробнее об этом можно узнать в «4.3. Состав приложения. Сервер».

Если полученный URL корректен, сервер обрабатывает запрос с учётом параметров, если они необходимы. После результата обработки возвращает клиенту ответ на его запрос. Ответ может содержать в себе различные элементы данных: описание автомобиля, клиента, сотрудника, продажи и различную другую информацию, опираясь на ранее перечисленные сущности.

Если же полученный URL не существует, то сервер возвращает ошибку 404 (Not Found). Что обозначает, что запрос не корректен, так как не реализован на стороне сервера. В данном случае необходимо отправить ещё один запрос, но на доступный URL адрес для получения необходимой информации на стороне клиента.

Может сложиться другая ситуация, URL корректен, но параметры переданные серверу не отвечают необходимым требованиям, в данном случае сервер вернёт ошибку 400 (Bad Request). Это означает, что необходимые

параметры не указаны, либо же не найдены на стороне сервера. Необходимо повторить запрос на сервер с верными параметрами по тому же URL для получения необходимой информации.

Для работы сервера необходимы несколько элементов: контроллеры, репозитории, сущности и сервисы. С помощью Spring Boot Data Rest достаточно использовать только сущности и репозитории, об этом подробнее будет описано в «4.3. Состав приложения. Сервер». Но так как необходимо реализовать собственную логику некоторых процессов, то дополнительно будет использован набор из Контроллеров/Сервисов, которые позволят реализовывать собственную логику по некоторым URL запросам.

Сервер соединяется с базой данных для получения необходимой информации в зависимости от запроса. Сервер должен уметь находить определенный элемент в базе данных, видоизменять и удалять информацию. Сервер не должен завершаться аварийным путём, необходимо отрабатывать все возможные ошибки, появившиеся в следствие запроса от клиента.

4.2. ОПИСАНИЕ ИНТЕРФЕЙСА ПРОГРАММЫ

У сервера нет графического интерфейса, в связи с этим в этом разделе пойдёт речь о клиенте реализованном с помощью JavaFX.

Всего в клиенте представлено 22 окна:

- Окно авторизации в системе;
- Окно с информацией о программе и создателе;
- Окна с информацией о продажах, клиентах, сотрудниках и автомобилях;
- Окна редактирования данных продажи, автомобиля, клиента и сотрудника;
- Окно добавления новой продажи;
- Окна добавления нового автомобиля, клиента и сотрудника;
- Окна с ошибками;
- Окна для подтверждения операций;
- Окна с выбором необходимого клиента, сотрудника или автомобиля для продажи
- Окна с подробной информацией о сотруднике, клиенте или автомобиле

Рассмотрим основные окна ниже.

Окно авторизации пользователя

Запрашивает логин/пароль пользователя. После ввода необходимых данных пользователь имеет возможность перейти к работе с программой с помощью кнопки «Авторизация». Если пользователь ввёл неверные или некорректные значения для логина или пароля, выводится окно с информацией о некорректных введённых данных. Регистрация возможно только через управляющего автосалоном, данная особенность реализована для защиты информации от посторонних личностей. Пользователю доносится информация

о том, что для регистрации необходимо обратиться к руководству компании. Окно авторизации продемонстрировано на рисунке 2 ниже.

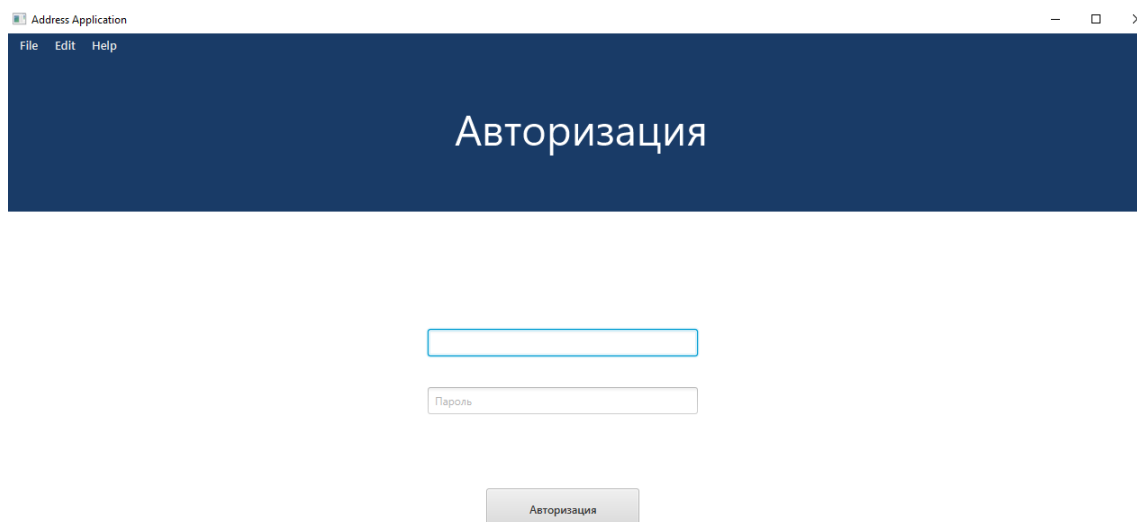


Рис. 2. Окно «Авторизация»

На форме существуют следующие валидаторы входных данных:

- Фильтрация на пустые поля e-mail и пароля (на стороне клиента);
- Фильтрация на некорректную авторизацию по паре логин/пароль (на стороне сервера).

При успешной авторизации осуществляется переход на окно с информацией о программе и создателе, если же валидация данных не прошла, то пользователю отображается соответствующая ошибка.

Окно с информацией о программе и создателе

На данном окне находится информация о АИС дилерского центра автомобилей и краткое руководство по использованию приложения. Так же указывается создатель приложения. В левой части экрана находится меню для перемещения между окнами. Кнопка «Продажи» реализует переход к окну с информацией о «Продажах», кнопка «Клиенты» реализует переход к окну с информацией о клиентах, кнопка «Сотрудники» реализует переход к окну с

информацией о сотрудниках, кнопка «Автомобили» реализует переход к окну с информацией об автомобилях. Окно представлено на рисунке 3.



Рис. 3. Окно «О программе и создателе»

С этого окна пользователь может переместиться в раздел с необходимой ему информацией.

Окно с информацией о продажах

В середине окна располагается таблица с информацией о продажах, под таблицей можно посмотреть подробную информацию о выбранной продаже из таблицы. Чтобы посмотреть всю информацию о клиенте, работнике или автомобиле необходимо нажать на соответствующую кнопку с названием «Информация» напротив интересующего элемента. В данном случае откроется новое окно, на котором находится вся необходимая информация.

С помощью кнопки «Добавить» пользователь имеет возможность добавить новую продажу. Подробнее данное окно описано далее. С помощью кнопки «Изменить» пользователь может изменить данные выбранной продажи, если же продажа не выбрана, появится окно с требованием выбрать продажу из таблицы. С помощью кнопки «Удалить» пользователю предоставляется возможность

[illegible]

Пользователь может фильтровать таблицу с продажами с помощью собственного критерия и помощью графы «Поиск». Фильтрация реализуется по ID, покупателю, автомобилю и сотруднику.

Данное окно предоставляет пользователю возможность добавить продажу автомобиля, при создании новой продажи пользователю необходимо выбрать автомобиль, клиента и сотрудника с помощью кнопки «Выбрать». После нажатия на данную кнопку откроется окно с выбором доступного авто, клиента или сотрудника, в зависимости от выбираемого элемента. Чтобы выбрать тот или иной элемент, необходимо из таблицы выбрать подходящее вам значение и после нажать на кнопку «Подтвердить выбор». Имеется возможность сразу же добавить новый элемент, если имеется такая необходимость.

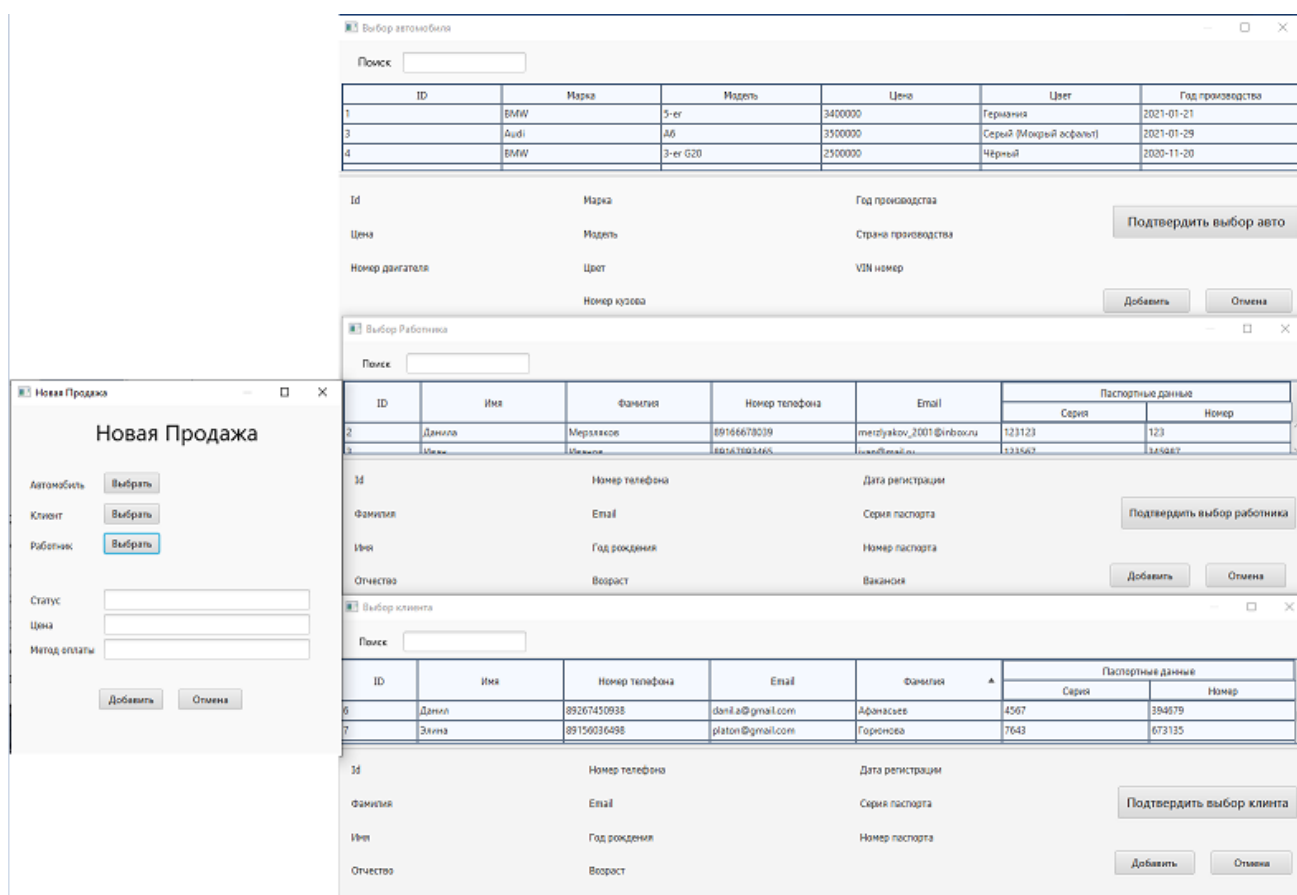


Рис. 5. Окно «Новая продажа»,

На рисунке 5 выше продемонстрировано окно «Новая продажа», которое используется для добавления нового элемента, также можно видеть таблицы с выбором необходимого авто, клиента, сотрудника. Для авто выбор предоставляется только из автомобилей в наличии. После выбора авто его стоимость автоматически будет установлена в графе «Цена», если же необходимо её изменить на иной значения, пользователь может установить её собственноручно. Выбрать клиента или сотрудника возможно только из уже зарегистрированных в системе.

Для добавления продажи пользователю необходимо нажать на кнопку «Добавить». На стороне клиента проводится проверка на вводимые данные, если всё верно, то запрос на добавление отправляется на сервер.

Окно с информацией об автомобилях

[illegible]

С помощью кнопки «Добавить» пользователь имеет возможность добавить новый автомобиль. При нажатии на кнопку откроется новое окно с формами ввода данных. При завершении пользователем ввода данных и нажатии на кнопку «Добавить» реализуется проверка вводимых данных на стороне клиента. Если всё верно, то POST запрос отправляется на сервер. Если же введены некорректные данным, то выводится окно с ошибкой.

15

проверка введенных значений, если вы верно, отправляется PUT запрос на сервер. В ином случае пользователю выводится окно с ошибкой.

С помощью кнопки «Удалить» пользователю предоставляется возможность удалить определённый авто. Если авто не выбрана выведется окно с ошибкой, в противном случае появится окно с подтверждением данной операции и предупреждение, что если данный авто зафиксирован в какой-либо продажи, то данная продажа тоже удалится.

Меню для перехода между окнами находится в левой части экрана, работу данных кнопок можно посмотреть в описании окна «Информация о программе и создателе»

Пользователь может фильтровать таблицу с автомобилями с помощью собственного критерия и графы «Поиск». Фильтрация реализуется по ID, марке и модели.

Окно с информацией о клиентах

В середине окна располагается таблица с информацией о клиентах. Под таблицей можно посмотреть подробную информацию о выбранном авто из таблицы. На рисунке 7 ниже изображено данное окно.

Данное окно работает наподобие окна «Информация о автомобилях». Более подробное описание кнопок и возможностей можно прочить там.

Пользователь может фильтровать таблицу с клиентами с помощью собственного критерия и графы «Поиск». Фильтрация реализуется по ID, имени, фамилии и паспортным данным.

Меню для перехода между окнами находится в левой части экрана, работу данных кнопок можно посмотреть в описании окна «Информация о программе и создателе»

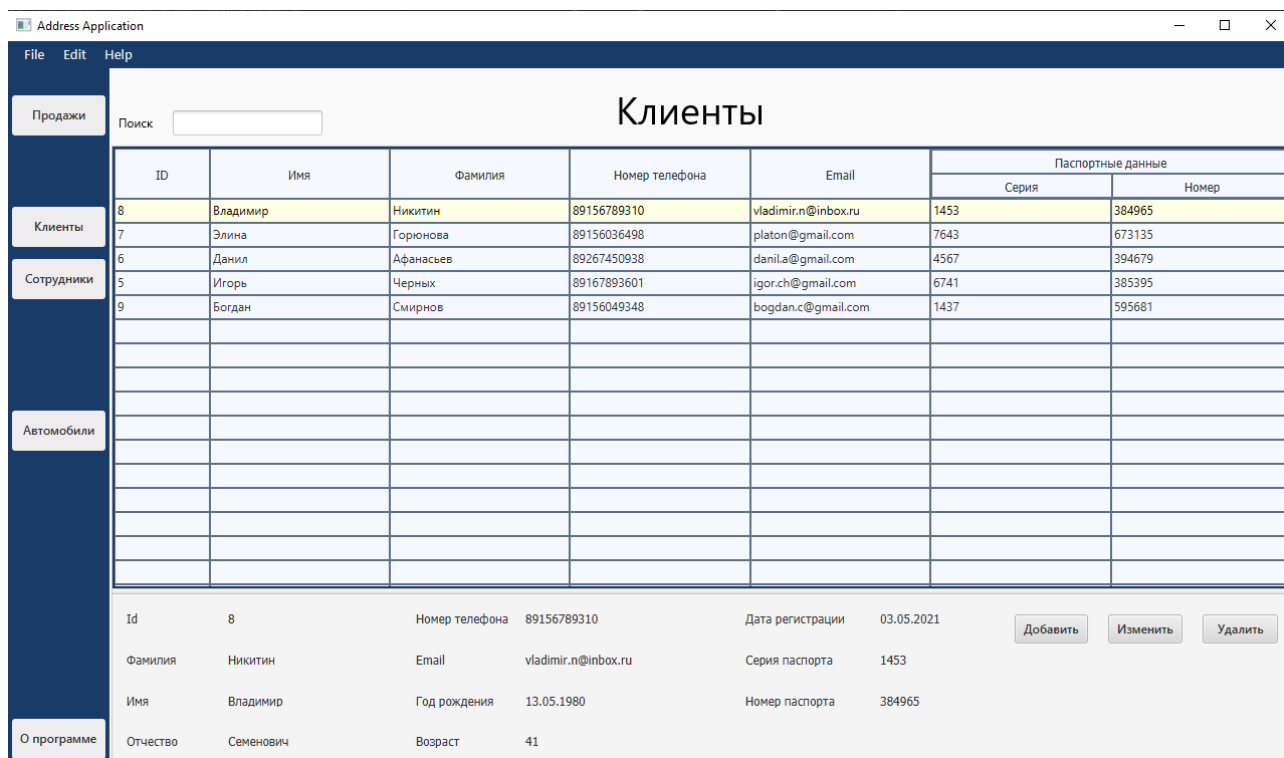


Рис. 7. Окно «Информация о клиентах»

Окно с информацией о сотрудниках

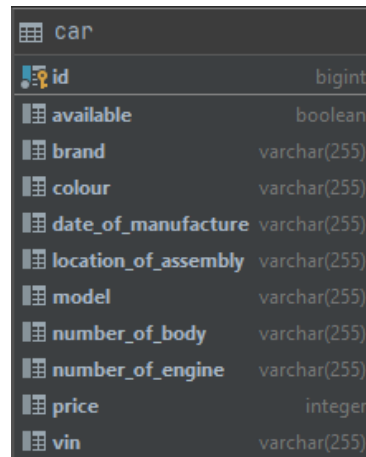
В середине окна располагается таблица с информацией о сотрудниках компании. Под таблицей можно посмотреть подробную информацию о выбранном авто из таблицы. Ниже на рисунке 8 изображено данное окно.

Данное окно работает наподобие окна «Информация о автомобилях» и «Информация о клиентах». Более подробное описание кнопок и возможностей можно прочить там. Различие заключается в том, что данное окно используется для предоставления новому сотруднику доступа к АИС “DeAuto”. Необходимо добавить нового сотрудника и прописать и указать пароль для входа. Для логина используется Email почта работника.

Пользователь может фильтровать таблицу с сотрудниками с помощью собственного критерия и графы «Поиск». Фильтрация реализуется по ID, имени, фамилии и паспортным данным.

Атрибуты данной сущности можно увидеть на рисунке 9 выше.

2. Car – сущность для хранения информации о автомобилях:

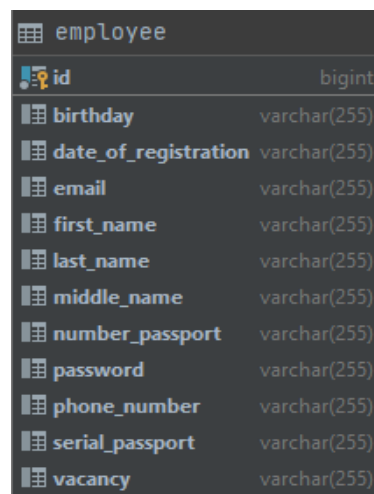


car	
id	bigint
available	boolean
brand	varchar(255)
colour	varchar(255)
date_of_manufacture	varchar(255)
location_of_assembly	varchar(255)
model	varchar(255)
number_of_body	varchar(255)
number_of_engine	varchar(255)
price	integer
vin	varchar(255)

Рис. 10. Атрибуты таблицы «Car»

Атрибуты данной сущности можно увидеть на рисунке 10 выше.

3. Employee - сущность для хранения информации о сотрудниках:



employee	
id	bigint
birthday	varchar(255)
date_of_registration	varchar(255)
email	varchar(255)
first_name	varchar(255)
last_name	varchar(255)
middle_name	varchar(255)
number_passport	varchar(255)
password	varchar(255)
phone_number	varchar(255)
serial_passport	varchar(255)
vacancy	varchar(255)

Рис. 11. Атрибуты таблицы «Employee»

Атрибуты данной сущности можно увидеть на рисунке 11 выше.

4. Sale - сущность для хранения информации о продажах:

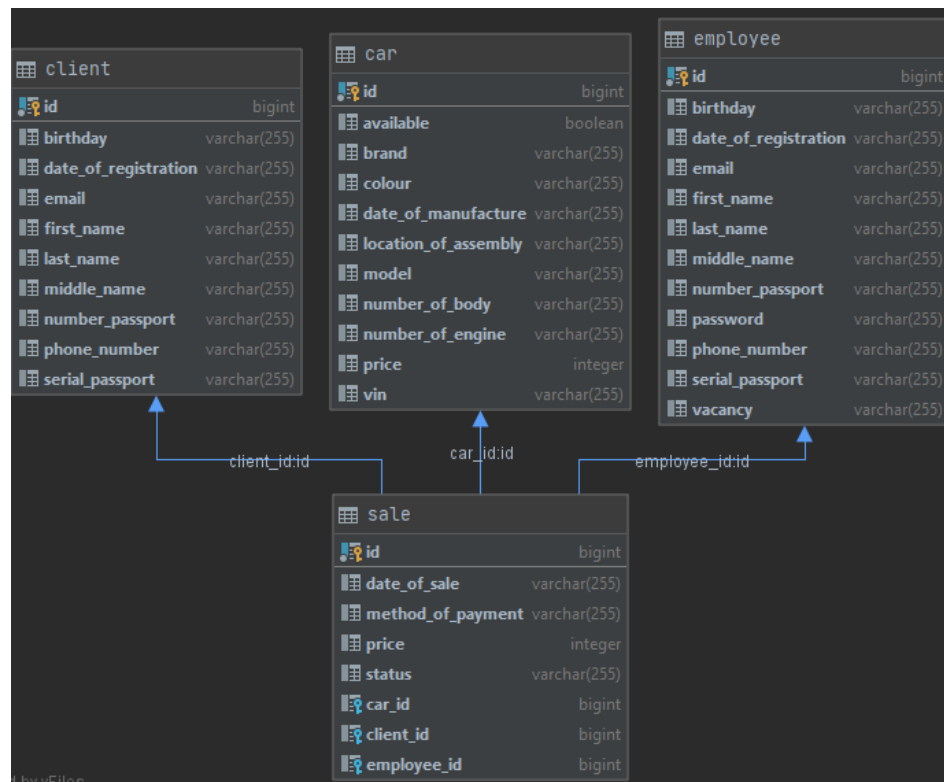


Рис. 12. Атрибуты таблицы «Sale»

Атрибуты данной сущности можно увидеть на рисунке 12.

С помощью данной СУБД будет храниться все необходимая информация для работы АИС «DeAuto».

Сервер

В состав сервера входят следующие компоненты:

- Spring Web для создания web-приложений, в том числе RESTful, с использованием Spring MVC (Model-View-Controller).
- Spring Data JPA – используется для представления сущностей в виде таблиц в PostgreSQL.
- Spring Data Rest – используется для прямого обращение к JPA репозиториям Spring. Нет необходимости реализовывать контроллеры и сервисы, так как Data Rest предоставляет вам данную возможность без дополнительных доработок.
- PostgreSQL – используется для соединения СУБД PostgreSQL с Spring.
- Lombok – удобный набор аннотаций для работы со Spring.

- Apache maven – основной сборщик проекта.
- Spring-boot-maven-plugin – плагин для сборки решения.

Сервер работает на порте 8080.

Клиент

В состав клиента входят следующие компоненты/внешние библиотеки:

- JavaFX (javafx-controls, javafx-fxml) – библиотека для отображения GUI и её компонентов.
- Google GSON – модуль для работы с запаковкой/распаковкой данных в формате JSON;
- Apache maven – основной сборщик проекта.
- Javafx-maven-plugin – плагин для автоподстановки флагов виртуальной машины Java (VM Options) библиотеки JavaFX при компиляции и запуске проекта.
- Commons-Codec – используется для MD5 шифрования данных

5. НАЗНАЧЕНИЕ И СОСТАВ КЛАССОВ ПРОГРАММЫ

Сервер

На сервере реализован 21 класс для его работы. Можно разделить их на группы:

- Контроллеры (4 шт.) – классы для обработки входящих запросов от клиентов, а точнее:

CarController – используется для обработки Post запросов, остальные запросы реализуются за счёт Spring Data Rest. Обеспечивает работу с данными об автомобилях.

ClientController – используется для обработки Post запросов, остальные запросы реализуются за счёт Spring Data Rest. Обеспечивает работу с данными о клиентах.

SaleController – используется для обработки Post запросов, остальные запросы реализуются за счёт Spring Data Rest. Обеспечивает работу с данными о продажах.

EmployeeController – используется для обработки Post запросов и проверки пары логин/пароль, остальные запросы реализуются за счёт Spring Data Rest. Обеспечивает работу с данными о сотрудниках.

– Сущности СУБД (4 шт.) и Projection к данным сущностям – конкретная коллекция в базе данных для последующей работы с её элементами, Projection используется для получение выборочных данных с сущности:

Car/CarProjection – Информация об автомобилях

Client/ClientProjection – Информация о клиентах

Employee/EmployeeProjection – Информация о работниках

Sale/SaleProjection – Информация о продажах

– Сервисы (4 шт.) – необходимо для обработки данных, полученных с контроллеров и работа с репозиториями, большинство запросов обрабатывается с помощью Spring Data Rest: CarService, ClientService, SaleService, EmployeeService;

– Репозитории (4 шт.) – представления таблиц СУБД, используется для получения информации из СУБД с помощью прописанных контроллеров и сервисов или с помощью Spring Data Rest: CarRepository, ClientRepository, SaleRepository и EmployeeRepository;

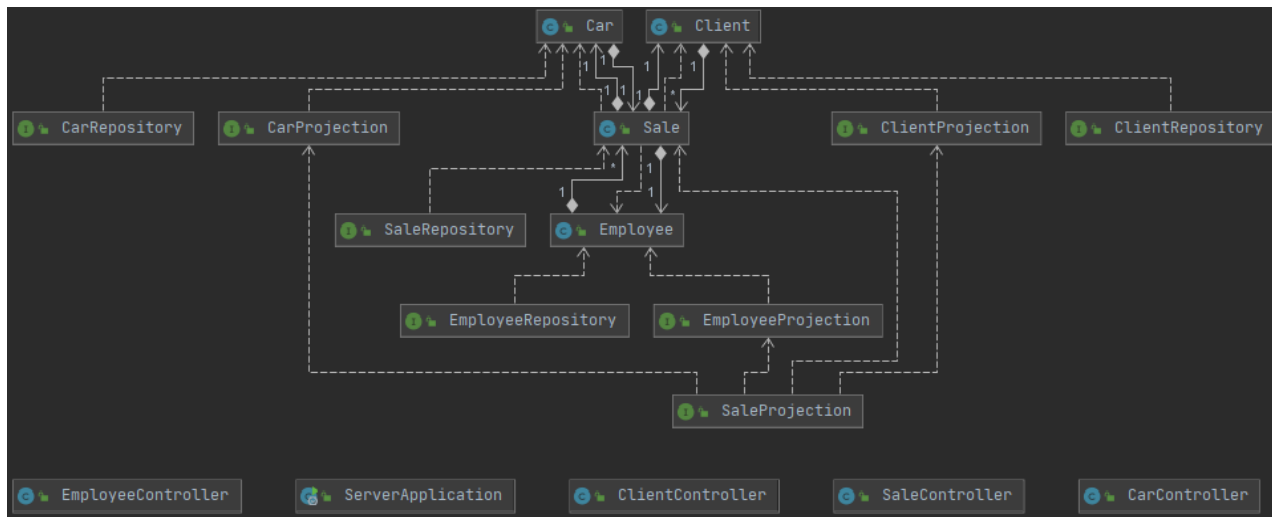


Рис. 13. Диаграмма классов Сервера

Выше на рисунке 13 представленная диаграмма классов. С её помощью видно логическую модель сервера.

Клиент

Со стороны клиента было разработано 33 класса, которые также можно объединить в группы:

- Классы работы с API (5 шт.) – Реализуют запросы к пользователю: CarRequest, ClientRequest, EmployeeRequest, SaleRequest, SuperRequest.
- Классы контроллеры (21 шт.) – отвечают за работу различных окон JavaFx приложения. Можно разделить по группам на контроллеры для работы с Информацией об автомобилях, продажах, клиентах, сотрудниках и дополнительных окон по типу авторизации и других.
- Классы сущностей (4 шт.) – модели для обработки сущностей, которые определены со стороны сервера, а именно клиент, работник, продажа и автомобиль: Car, Client, Employee, Sale.

– Классы-утилиты (2 шт.) – необходимы для обработки дополнительных операций, а именно для работы с датами и шифрованию данных: DateUtil и MD5Util.

– Точка входа в программу (1 шт.) – базовый класс, от которого запускается программа: App

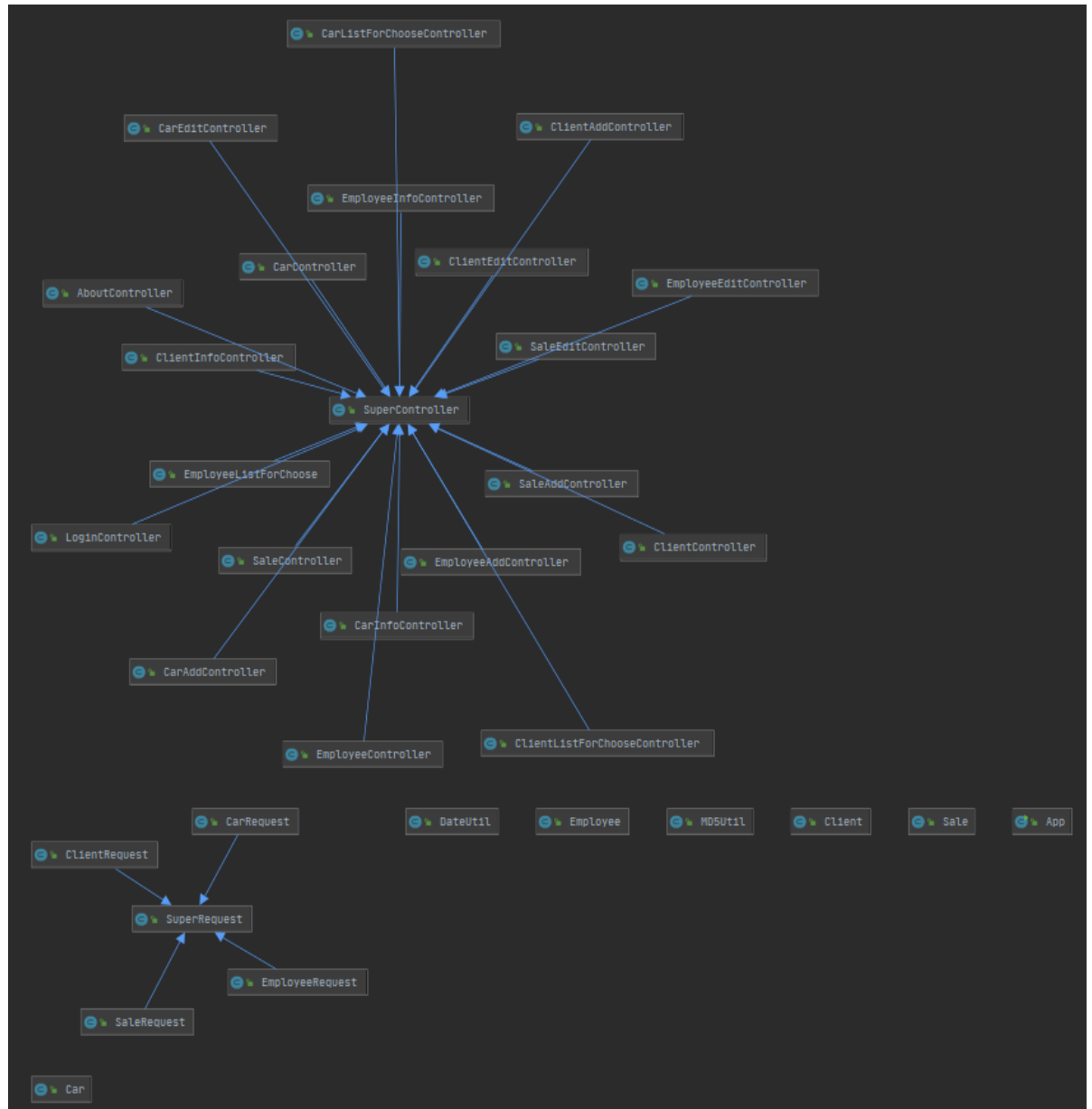


Рис. 14. Диаграмма классов Клиента

Диаграмма классов представлена выше на рисунке на рисунке 14. На ней можно проанализировать как устроена логика работы клиента.

ЗАКЛЮЧЕНИЕ

В процессе выполнения курсовой работы было разработано клиент-серверное приложение для дилерского центра автомобилей, которое включает в себя клиент с применением библиотеки JavaFX и сервер, основанный на библиотеке Spring Boot.

Приложение полностью готово к работе и никогда не завершается аварийным путём за счёт обработки всех возможных ошибок. Клиент взаимодействует с сервером через RESTful API с помощью протокола http. Данные полученные с сервера отображаются в элементах графического интерфейса клиента.

Приложение может обновляться и модернизироваться. К примеру, может понадобиться хранить в АИС информацию о поставщиках товара или любую другую информацию. Может понадобиться реализация данной АИС для мобильных устройств, что можно реализовать и после опять же подключить к написанному нами серверу.

В процессе разработки получены обширные навыки языка программирования Java и дополнительными необходимыми фреймворками и библиотеками по типу Spring Boot, JavaFx, PostgreSQL, Google GSON и другими.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

Учебная и научная литература

1. Володченкова Л.А., Козырев Д.В. Разработка серверной части программного приложения для удаленного хранения данных // МСиМ. 2020. №1 (53).
2. Гасанов Заурбек Зубаирович Анализ производительности многопоточных программ, написанных на языках Java и Go // Наука и образование сегодня. 2018. №6 (29).
3. Байдыбеков А.А., Гильванов Р.Г., Молодкин И.А. СОВРЕМЕННЫЕ ФРЕЙМВОРКИ ДЛЯ РАЗРАБОТКИ WEB-ПРИЛОЖЕНИЙ // Интеллектуальные технологии на транспорте. 2020. №4 (24).
4. Коузен К. Современный Java. Рецепты программирования . - М.: ДМК Пресс, 2018. - 274 с.
5. Прохоренок Н.А. JavaFX. - СПб: БХВ-Петербург, 2020. - 768 с.
6. Козмина Ю., Харроп Р. Spring 5 для профессионалов. - Киев: Диалектика-Вильямс, 2019. - 1120 с.
7. Чистов Д.В., Мельников П.П., Золотарюк А.В. Проектирование информационных систем: Учебник/ Под общ. ред. Чистова Д.В. - М.: Юрайт, 2019. -238 с.
8. Машнин Т. JavaFX 2.0. Разработка RIA-приложений // БХВПетербург. 2012. 320 с.
9. DiMarzio J.F. Quick Start Guide to JavaFX // McGraw-Hill Osborne Media. 2011. 266с
10. Хорстманн К.С. Java. Библиотека профессионала. Том 1. Основы// Питер. 2017. 864 с.