Министерство образования Республики Беларусь

Учреждение образования

«Белорусский государственный технологический университет»

Кафедра информационных систем и технологий

**Отчет по лабораторной работе на тему**:

**«Объектно-ориентированное моделирование. Структурные диаграммы UML»**

Выполнил:

студент 3 курса 1 группы ФИТ

Вакуленчик В.С.

Цель:

Изучение методологии объектно-ориентированного моделирования средствами UML. Ознакомление с основными принципами объектно-ориентированного проектирования программного обеспечения, получение навыков проектирования структуры информационной системы с применением UML.

Минск 2023

**1 Функциональное назначение**

**1.1 Функциональное назначение для менеджера**

Регистрация автомобилей в базу данных: указание характеристик автомобиля (марка, модель, стоимость, цвет кузова).

Управление каталогом: возможность редактирования информации об авто и удаление моделей.

Управление заказами: просмотр, обработка и управление заказами, включая подтверждение, отмену заказов и оповещение покупателя об отказе/одобрении брони.

**1.2 Функциональное назначение для покупателя**

Аутентификация пользователей: позволяет покупателям входить в свои учетные записи для доступа к каталогу автомобилей.

Бронирование автомобилей: позволяет просматривать каталог, производить бронь автомобилей и отказываться от брони. Также можно просмотреть забронированные авто.

Поиск и фильтрация: наличие поиска автомобилей по марке, модели, году выпуска или цвету. Также добавление фильтров, чтобы покупатели могли уточнить результаты поиска по различным критериям.

**2. Описание программных средств**

Для создания диаграмм компонентов и развертывания использовалось приложение draw.io. draw.io — инструмент для создания диаграмм, блок-схем, интеллект-карт, бизнес-макетов, отношений сущностей, программных блоков и другого. Сервис распространяется на бесплатной основе с открытым исходным кодом. Draw.io обладает богатым набором функций для визуализации большинства задач пользователя.

Изучение методологии объектно-ориентированного моделирования средствами UML. Ознакомление с основными принципами объектно-ориентированного проектирования программного обеспечения, получение навыков проектирования функциональности информационной системы с применением UML.

Особенности draw.io:

* Более 500 шаблонов элементов и фигур;
* облегчённый интерфейс, в котором за короткий промежуток времени можно создать готовый проект;
* поддержка горячих клавиш, задействованных в большинстве графических редакторов;
* экспорт в форматы: JPG, PNG, SVG, VDSX;
* возможность совместной работы;
* наличие различных фоновых тем;
* мультиязычный интерфейс.

# 3. Описание практического задания

В данной лабораторной работе было необходимо построить диаграмму

классов. На рисунке 3.1 представлена диаграмма с классами уровня модели приложения.

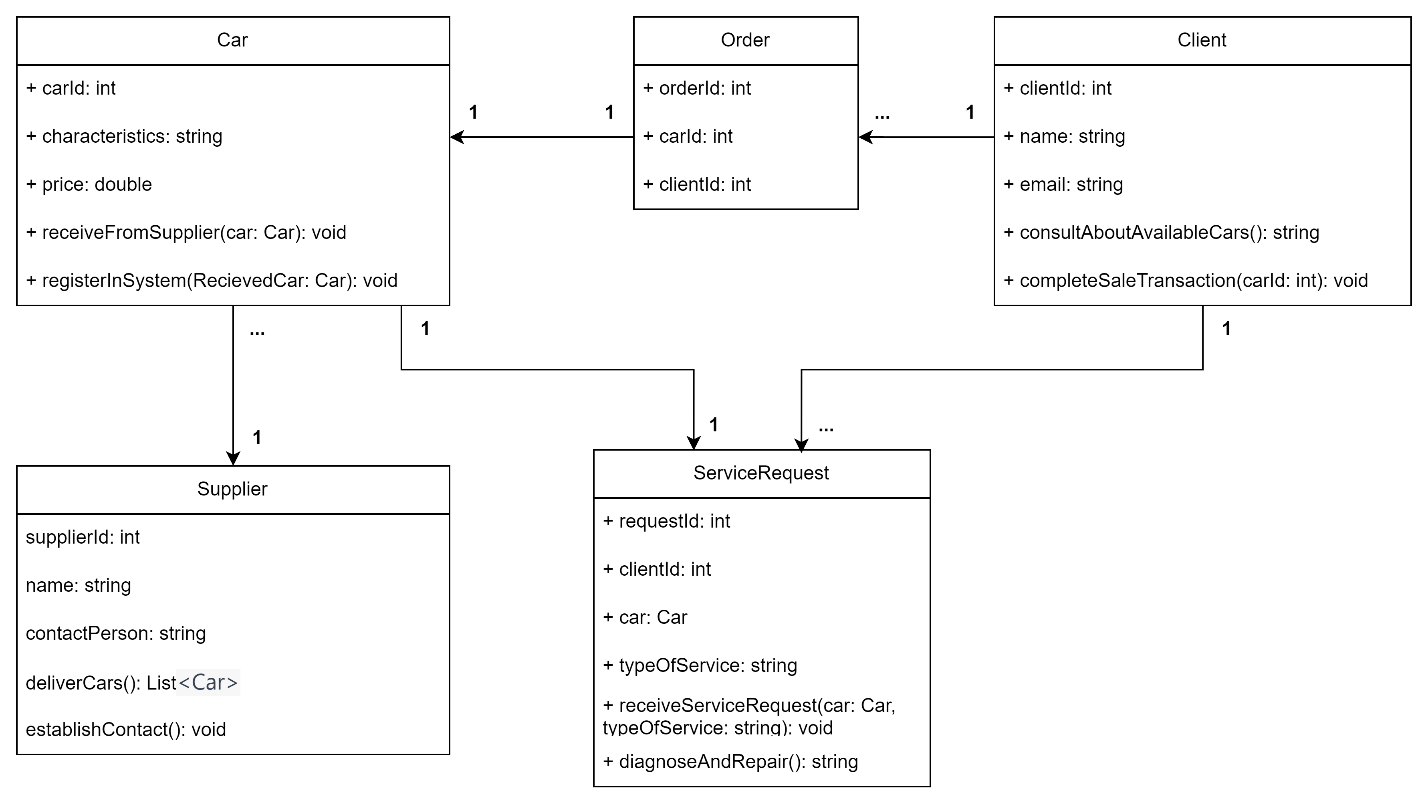


Рисунок 3.1. Классы модели клиентского приложения

В блоках мы указываем название класса, поля и методы, из которых он состоит. Через двоеточие мы обозначаем тип данных поля. Знаком «+» мы говорим о том, что поле доступно для использования вне класса. «1…» обозначает тип связи один-ко-многим.

Car: receiveFromSupplier(carId: int): void: Этот метод используется для принятия новых автомобилей от поставщика. Он принимает идентификатор автомобиля и выполняет необходимые действия для включения его в систему.

Car: registerInSystem(RecievedCar Car): void: Этот метод регистрирует автомобиль в системе с указанием характеристик и цен. После вызова этого метода информация о машине будет доступна для продажи и сервисного обслуживания.

Client: consultAboutAvailableCars(): string: Клиент может использовать этот метод для получения консультации относительно доступных автомобилей. Метод возвращает строку с информацией о наличии машин.

Client: completeSaleTransaction(carId: int): void: Когда клиент решит купить автомобиль, этот метод вызывается для завершения сделки. Он принимает идентификатор автомобиля и выполняет необходимые действия для оформления документов по продаже.

ServiceRequest: receiveServiceRequest(car: Car, typeOfService: string): void: Метод принимает запрос на техническое обслуживание и ремонт. Он получает объект автомобиля и тип запрашиваемого сервиса.

ServiceRequest: diagnoseAndRepair(): string: Этот метод выполняет диагностику и ремонт автомобиля. Возвращает строку с результатами диагностики и выполненных работ.

Supplier: deliverCars(): List<Car>: Поставщик использует этот метод для поставки новых автомобилей в автосалон. Возвращает список поставленных машин.

Supplier: establishContact(): void: Устанавливает контакт с автосалоном для согласования поставок и других вопросов.