

**МИНОБРНАУКИ РОССИИ**  
**САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ**  
**ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ**  
**«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)**  
**Кафедра вычислительной техники**

**Пояснительная записка к курсовому проекту**  
**по дисциплине «Объектно-ориентированное программирование»**  
**Тема: Создание программного комплекса средствами объектно-**  
**ориентированного программирования**

Студент гр. 3312

\_\_\_\_\_

Сабиров Р. Д.

Преподаватель

\_\_\_\_\_

Павловский М. Г.

Санкт-Петербург

2024

## Содержание

1. Техническое задание .....	3
1.1 Введение.....	3
1.2 Основания для разработки .....	3
1.3 Назначение разработки.....	3
1.4 Требования к программе .....	3
1.5 Требования к программной документации .....	5
1.6 Стадии и этапы разработки.....	5
1.7 Порядок контроля и приемки .....	5
2. Процесс проектирования программного комплекса .....	6
2.1 Описание вариантов использования ПК.....	6
2.2 Создание прототипа интерфейса пользователя .....	7
2.3 Разработка объектной модели ПК.....	13
2.4 Построение диаграммы программных классов .....	17
2.5 Описание поведения ПК.....	19
2.6 Построение диаграммы действий.....	21
2.7 Реализация хранения данных многие-ко-многим.....	22
3. Руководство оператора .....	23
3.1 Назначение программы .....	23
3.2 Условия выполнения программы .....	23
3.3 Описание задачи.....	24
3.4 Входные и выходные данные .....	25
3.5 Выполнение программы.....	27
3.6 Проверка программы .....	35
3.7 Сообщения оператору.....	35
Заключение .....	37

## **1. Техническое задание**

### **1.1 Введение**

Программный комплекс (ПК) для диспетчеризации автобусного парка. ПК предназначен для управления данными о водителях, маршрутах и графиках движения автобусов.

### **1.2 Основания для разработки**

Основанием для разработки ПК диспетчера автобусного парка является создание удобного инструмента для работы диспетчеров.

### **1.3 Назначение разработки**

Функциональным назначением ПК является управление данными о водителях, маршрутах и графиках движения автобусов. Эксплуатационным назначением ПК – использование диспетчерами в автобусных парках.

### **1.4 Требования к программе**

#### *1.4.1 Требования к функциональным характеристикам*

1. Возможность управления данными, такими способами как, добавление, редактирование и удаление позволяют обеспечить ведение и хранение данных:
  - Список водителей, их стаж работы или класс;
  - Маршруты, время их движения, начальные и конечные остановки;
  - График движения автобусов.
2. Отображение данных в табличном виде с сортировкой и поиском данных;
3. Сохранение и загрузка данных из внешних файлов (XML).
4. Хранение всей информации в базе данных Access, с возможностью смены БД в любой момент.
5. Формирование отчета в PDF или HTML форматах.

#### *1.4.2 Требования к надежности*

Программный комплекс должен устойчиво функционировать при соблюдении гарантии устойчивого функционирования операционной системы и системы управления базой данных, т.е. программа должна непрерывно функционировать в отсутствии критических сбоев, приводящих к аварийному завершению работы.

Должен быть обеспечен контроль входных данных на предмет соответствия предполагаемому типу.

#### *1.4.3 Условия эксплуатации*

Программный комплекс обеспечивает выполнение своих функций для однопользовательского режима с монопольным доступом к базе данных Access.

#### *1.4.4 Требование к составу и параметрам технических характеристик*

Программный комплекс должен функционировать на ПК со следующими минимальными характеристиками:

1. Процессор: Intel Core i3 или аналогичный;
2. Видеокарта: интегрированная (IntelHD Graphics или аналогичный);
3. ОЗУ: 256 Мб;
4. ПЗУ: 256 Мб свободного пространства;
5. Монитор: с поддержкой разрешения 1366x768;
6. Мышь и клавиатура

#### *1.4.5 Требование к информационной и программной совместимости*

1. Программа должна быть разработана на языке программирования Java;
2. Совместимость с ОС Windows и Linux;
3. Использование базы данных Access;
4. Поддержка русскоязычного и англоязычного интерфейсов;
5. Использование ключевых конструкций языка Java.

## **1.5 Требования к программной документации**

Программная документация должна быть представлена в следующем составе:

1. Техническое задание;
2. Описание процесса проектирования ПК;
3. Руководство оператора;
4. Исходные тесты ПК.

## **1.6 Стадии и этапы разработки**

В стадии разработки ПК входят:

1. Анализ задания и требований, разработка технического задания
2. Проектирование интерфейса базы данных
3. Разработка модулей управления данными
4. Тестирование и исправление ошибок
5. Написание программной документации.

Сроком разработки ПК является один учебный семестр. Исполнителем является студент.

## **1.7 Порядок контроля и приемки**

В ходе тестирования проверялись все функции программного комплекса: добавление, редактирование, удаление, сохранение и загрузка, сортировка, поиск, формирование отчета, используя различные данные, включая некорректные. Также были обработаны ошибки, связанные с доступом и запросами к базе данных.

## **2. Процесс проектирования программного комплекса**

### **2.1 Описание вариантов использования ПК**

Диаграмма прецедентов (use case diagram) демонстрирует функциональные требования к системе диспетчеризации автобусного парка, которые были сформулированы на этапе проектирования. Одним из ключевых шагов в разработке является определение процессов, происходящих в системе, и их описание через прецеденты. Прецеденты отображают сценарии использования системы, в которых она взаимодействует с внешними элементами.

На диаграмме показаны два типа участников взаимодействия: акторы и прецеденты. Акторы представляют собой внешние сущности, такие как пользователь системы (диспетчер) и база данных (Access), которые участвуют в выполнении операций. Актор «Диспетчер» инициирует все основные действия в системе, такие как добавление, редактирование и удаление данных о водителях, маршрутах и графике движения. Актор «База данных» обеспечивает хранение, извлечение и обновление информации, необходимой для функционирования системы.

Прецеденты на диаграмме описывают функциональные возможности системы, такие как: добавление, редактирование, удаление записей водителей, маршрутов, расписаний, поиск, сортировка, загрузка и выгрузка данных, а также создание отчета

Связь коммуникации между актором «Диспетчер» и прецедентами отражает, что пользователь инициирует выполнение указанных действий. Например, стрелка от «Диспетчер» к прецеденту «Добавить водителя» указывает на то, что данное действие начинается с запроса пользователя.

Связь использования между прецедентами подчеркивает их взаимозависимость. Например, прецеденты «Добавить водителя» и «Удалить водителя» используют функциональность прецедента «Обновить интерфейс»

для отражения изменений на экране после выполнения операций. Аналогично, прецеденты взаимодействуют с базой данных для выполнения задач, таких как добавление, удаление или поиск информации.

Диаграмма прецедентов позволяет визуализировать взаимодействие пользователя с системой, акцентируя внимание на ключевых функциях, требованиях к системе и взаимосвязях с внешними данными и интерфейсами.

Диаграмма прецедентов представлена на рисунке 2.1.

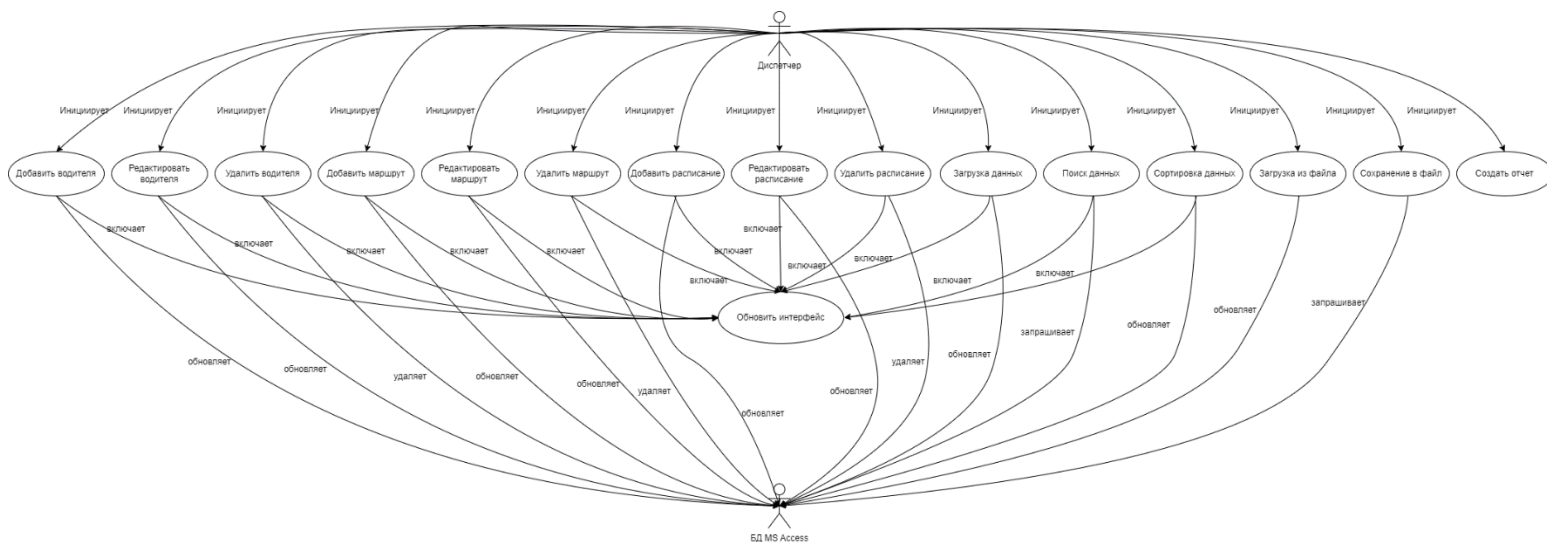


Рисунок 2.1 – Диаграмма прецедентов

## 2.2 Создание прототипа интерфейса пользователя

Описание прецедента выражает общую сущность процессов взаимодействия пользователя с системой, без детализации реализации. Проектные решения, связанные с пользовательским интерфейсом, при этом формализуются в терминах конкретных экранных форм и элементов управления. Для разработки пользовательского интерфейса программы «Диспетчеризация автобусного парка» были выделены ключевые процессы, которые представлены в виде экранных форм. Каждая экранная форма описывает: элементы управления (кнопки, меню, таблицы). Действия пользователя (нажатие кнопок, ввод текста). (см. рис. 2.2 - 2.5, табл. 2.1)

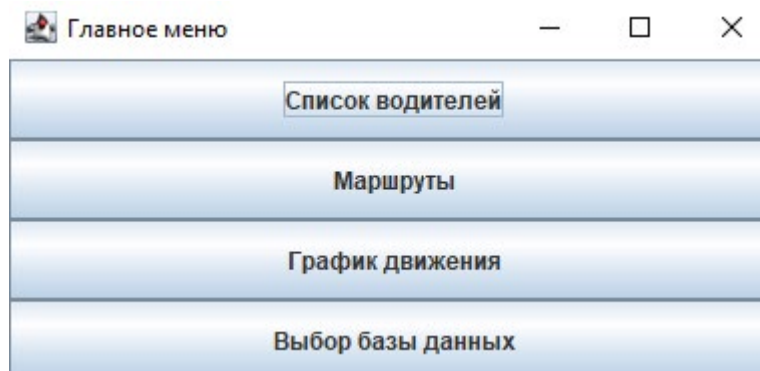


Рис 2.2 - Экранная форма «Главное меню»

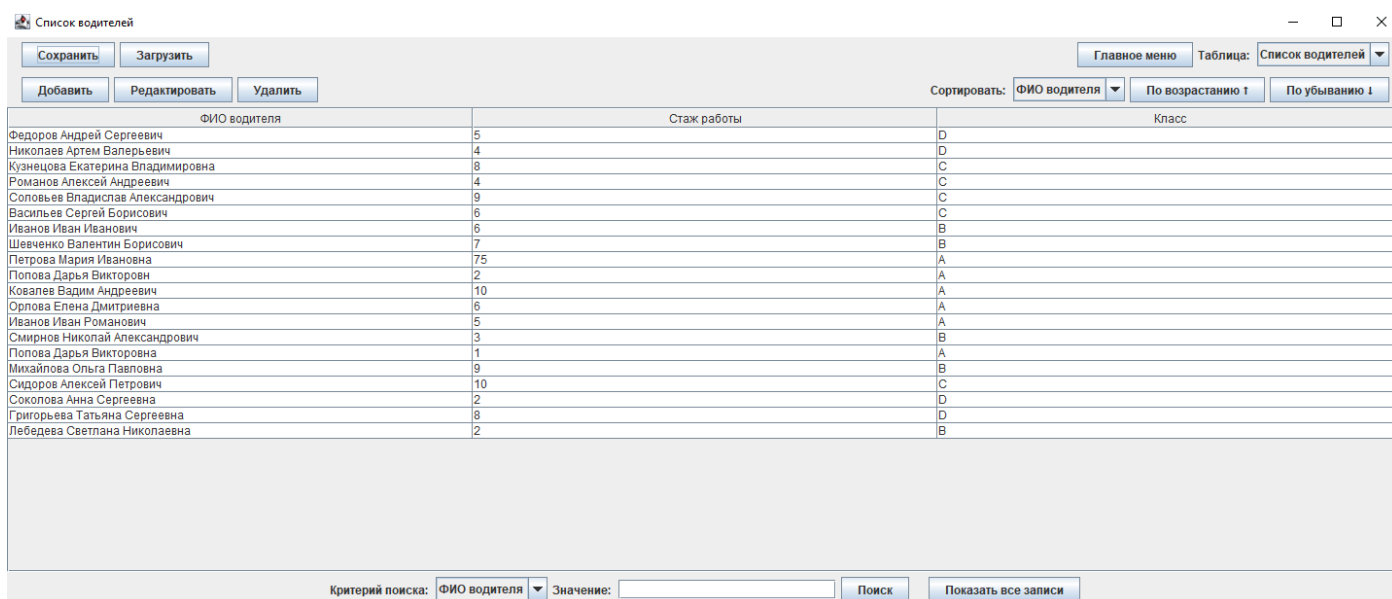


Рис 2.3 - Экранная форма «Список водителей»

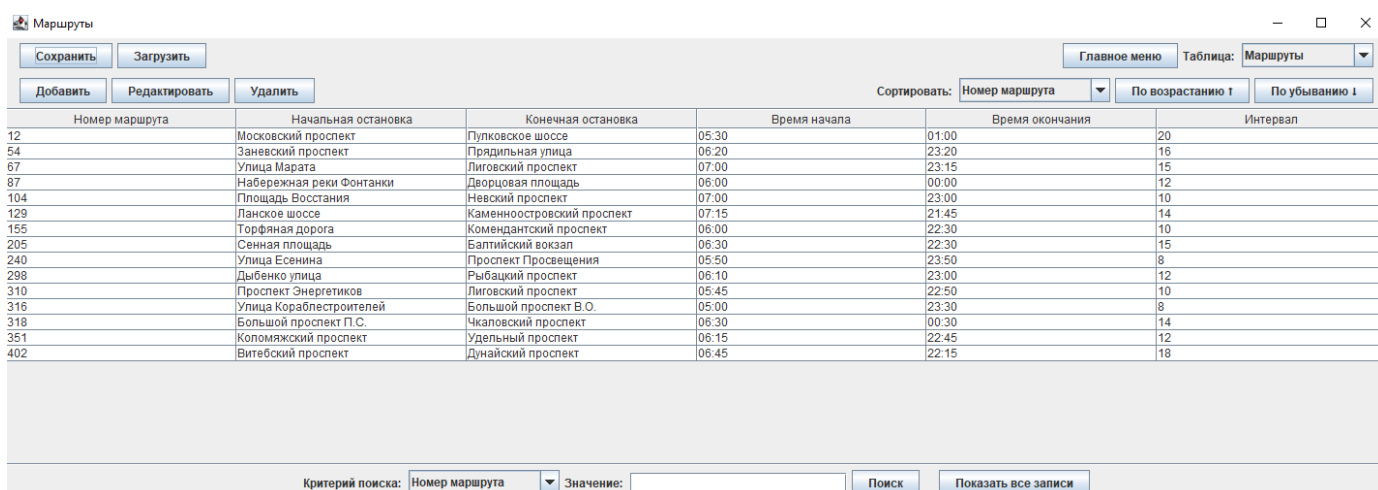


Рис 2.4 - Экранная форма «Маршруты»



График движения

Сохранить Загрузить

Главное меню Таблица: График движения

Добавить Редактировать Удалить

Создать отчет Сортировать: Номер маршрута По возрастанию t По убыванию t

Номер маршрута	ФИО водителя	Номер автобуса	Время отправления	Время прибытия	Факт. время отправления	Факт. время прибытия	Нарушения
12	Федоров Андрей Сергеевич	105	05:30	06:00	05:30	06:05	Опоздание на прибытие
12	Попова Дарья Викторовна	107	06:00	06:20	06:00	06:18	Нет
67	Иванов Иван Иванович	109	05:00	05:30	05:00	05:30	Нет
87	Федоров Андрей Сергеевич	108	12:00	12:20	12:00	12:18	Нет
104	Иванов Иван Иванович	102	09:00	09:30	09:00	09:28	Нет
205	Петрова Мария Ивановна	101	07:00	07:30	07:05	07:35	Опоздание на отправление
205	Иванов Иван Иванович	104	08:00	08:30	08:05	08:33	Нет
316	Попова Дарья Викторовна	110	01:00	01:30	01:00	01:29	Нет
318	Петрова Мария Ивановна	101	07:00	07:30	07:05	07:35	Опоздание на отправление

Критерий поиска: Номер маршрута Значение: Поиск Показать все записи

Рис 2.5 - Экранная форма «График движения»

Таблица 2.1

Экранная форма	Элементы управления	Действия пользователя	Отклик системы
Главное меню	Кнопки: «Список водителей», «Маршруты», «График движения», «Выбор базы данных»	Нажатие на кнопки: «Список водителей», «Маршруты», «График движения»	Открытие соответствующей экранной формы: «Список водителей», «Маршруты», «График движения»
		Нажатие на кнопку «Выбор базы данных»	Открытие окна по выбору базы данных, находящейся на ПК пользователя, выбор базы данных и дальнейшая работа с ней
Список водителей	Кнопки: «Добавить», «Редактировать», «Удалить», «Сохранить», «Загрузить», «По возрастанию», «По убыванию», «Поиск», «Показать все записи», «Главное меню». Выпадающие списки: «Сортировка»,	Нажатие кнопки «Добавить»	Открытие экранной формы по добавлению, добавление новой записи о водителе
		Нажатие кнопки «Редактировать»	Открытие экранной формы по редактированию, редактирование выбранной записи водителя
		Нажатие кнопки «Удалить»	Открытие экранной формы по удалению, удаления выбранной записи водителя

	«Критерий поиска», «Таблица». Текстовое поле «Поиск»	Нажатие кнопки «Сохранить»	Открытие экранной формы по выбору места сохранения данных, сохранение всех таблиц в один XML файл
		Нажатие кнопки «Загрузить»	Открытие экранной формы по выбору файла, загрузка всех таблиц (добавление данных) из XML файла
		Нажатие кнопки «По возрастанию» и «По убыванию»	Соответствующая сортировка таблицы с данными
		Нажатие кнопки «Поиск»	Поиск по таблице и отображение соответствий по выбранному и введенному критерию поиска
		Нажатие кнопки «Показать все записи»	Отображение всех данных в таблице после использования поиска
		Нажатие кнопки «Главное меню»	Переход к экранной форме «Главное меню»
		Нажатие на выпадающий список «Сортировка»	Выбор поля для сортировки
		«Нажатие на выпадающий список «Критерий поиска»	Выбор поля для поиска в таблице
		Нажатие на выпадающий список «Таблица»	Выбор таблицы, переход к ней
Маршруты	Кнопки: «Добавить», «Редактировать», «Удалить», «Сохранить», «Загрузить», «По возрастанию», «По убыванию», «Поиск»,	Нажатие кнопки «Добавить»	Открытие экранной формы по добавлению, добавления новой записи о маршруте
		Нажатие кнопки «Редактировать»	Открытие экранной формы по редактированию, редактирование

	«Показать все записи», «Главное меню». Выпадающие списки: «Сортировка», «Критерий поиска», «Таблица». Текстовое поле «Поиск»		выбранной записи маршрута
		Нажатие кнопки «Удалить»	Открытие экранной формы по удалению, удаления выбранной записи маршрута
		Нажатие кнопки «Сохранить»	Открытие экранной формы по выбору места сохранения данных, сохранение всех таблиц в один XML файл
		Нажатие кнопки «Загрузить»	Открытие экранной формы по выбору файла, загрузка всех таблиц (добавление данных) из XML файла
		Нажатие кнопки «По возрастанию» и «По убыванию»	Соответствующая сортировка таблицы с данными
		Нажатие кнопки «Поиск»	Поиск по таблице и отображение соответствий по выбранному и введенному критерию поиска
		Нажатие кнопки «Показать все записи»	Отображение всех данных в таблице после использования поиска
		Нажатие кнопки «Главное меню»	Переход к экранной форме «Главное меню»
		Нажатие на выпадающий список «Сортировка»	Выбор поля для сортировки
		«Нажатие на выпадающий список «Критерий поиска»	Выбор поля для поиска в таблице
		Нажатие на выпадающий список «Таблица»	Выбор таблицы, переход к ней
График движения	Кнопки: «Добавить», «Редактировать»,	Нажатие кнопки «Добавить»	Открытие экранной формы по добавлению,

«Удалить», «Сохранить», «Загрузить», «Создать отчет», «По возрастанию», «По убыванию», «Поиск», «Показать все записи», «Главное меню», Выпадающие списки: «Сортировка», «Критерий поиска», «Таблица».		добавления новой записи о маршруте
	Нажатие кнопки «Редактировать»	Открытие экранной формы по редактированию, редактирование выбранной записи маршрута
	Нажатие кнопки «Удалить»	Открытие экранной формы по удалению, удаления выбранной записи маршрута
	Нажатие кнопки «Сохранить»	Открытие экранной формы по выбору места сохранения данных, сохранение всех таблиц в один XML файл
	Нажатие кнопки «Загрузить»	Открытие экранной формы по выбору файла, загрузка всех таблиц (добавление данных) из XML файла
	Нажатие кнопки «Создать отчет»	Открывается экранная форма с выбором формата отчета и места его сохранение, создание и сохранение отчета
	Нажатие кнопки «По возрастанию» и «По убыванию»	Соответствующая сортировка таблицы с данными
	Нажатие кнопки «Поиск»	Поиск по таблице и отображение соответствий по выбранному и введенному критерию поиска
	Нажатие кнопки «Показать все записи»	Отображение всех данных в таблице после использования поиска
	Нажатие кнопки «Главное меню»	Переход к экранной форме «Главное меню»
	Нажатие на выпадающий	Выбор поля для сортировки

		список «Сортировка»	
		«Нажатие на выпадающий список «Критерий поиска»	Выбор поля для поиска в таблице
		Нажатие на выпадающий список «Таблица»	Выбор таблицы, переход к ней

### 2.3 Разработка объектной модели ПК

Объектная модель не описывает структуру программы, а отображает основные понятия предметной области в виде совокупности типов объектов (сущностей). Сущности формируются путем анализа предметной области и выделения ключевых объектов на основе анализа сценариев использования.

На диаграмме каждая сущность представляется прямоугольником, внутри которого указываются: имя сущности, атрибуты (свойства), операции (методы).

Атрибуты описывают свойства сущности. Включаются те атрибуты, которые необходимы для выполнения требований системы или хранения данных для дальнейшей обработки. Каждый атрибут характеризуется именем и типом данных. Для описания атрибутов обычно используются простые типы, такие как целое число, строка, дата, время и т. д.

Операции представляют действия, выполняемые сущностью. В рамках объектной модели указываются основные операции для каждой сущности, но без детального описания их взаимодействия с другими объектами.

На диаграмме операций отображается только их имя, тогда как подробное описание включается в отдельную таблицу, где приводятся: краткое описание, назначения операции, имя операции, список входных и выходных параметров.

Ассоциации между сущностями отражают бинарные отношения, существующие между объектами. Ассоциация отображается линией,

соединяющей соответствующие сущности, с указанием имени, которое должно выражать семантический смысл связи.

Кратность в связях определяет количество экземпляров одной сущности, связанных с экземплярами другой. Примеры кратности:

0..\* — ноль или больше,

1..\* — один или больше,

1 — ровно один.

На диаграмме объектной модели системы автобусного парка отображены ключевые сущности: водитель, маршрут, расписание, а также база данных.

Диаграмма объектов представлена на рис. 2.6, а подробное описание операций каждой сущности — в таблице 2.2.

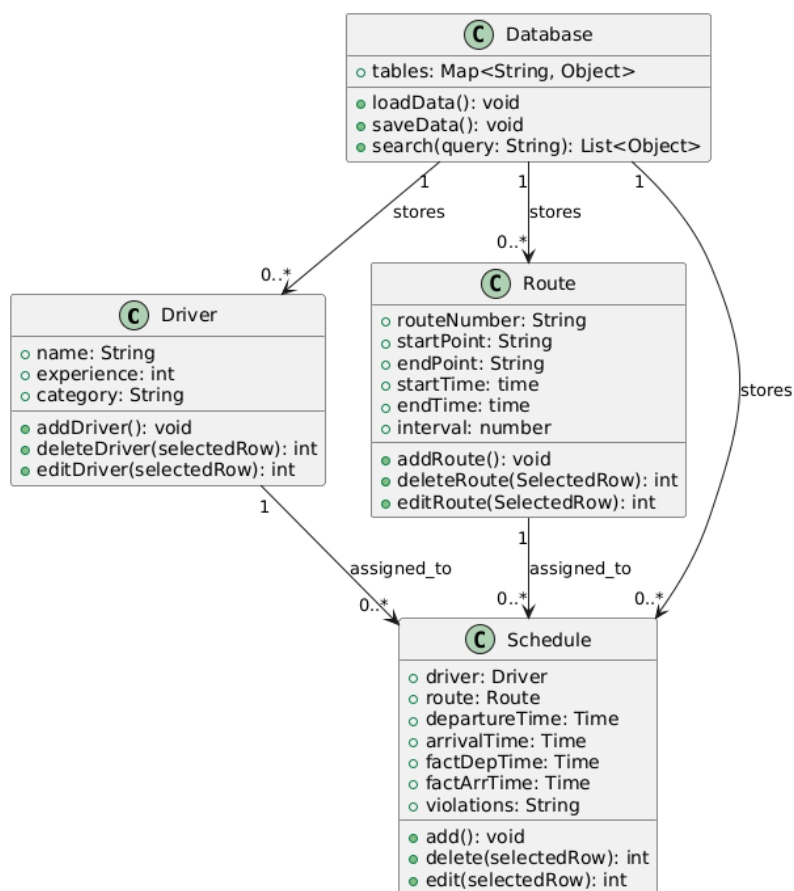


Рис. 2.6 – Диаграмма сущностей

Таблица 2.2

Сущность	Имя операции	Параметры операции			Тип возвращаемого значения	Назначение операции
		Вид	Название	Тип		
Водитель	Добавить	Вх.	ФИО, стаж, класс	String, int, string	Void	Добавление нового водителя
	Редактировать	Вх.	ID водителя, обновленные данные	int, string, int, string	Void	Редактирование существующего водителя
	Удалить	Вх.	ID водителя	int	void	Удаление водителя по ID
Маршрут	Добавить	Вх.	Номер маршрута, начальная остановка, конечная остановка, время начала, время окончания, интервал	Int, string, string, time, time, int	Void	Добавление нового маршрута
	Редактировать	Вх.	ID маршрута, номер маршрута, начальная остановка, конечная остановка, время начала, время окончания, интервал	Int, Int, string, string, time, time, int	Void	Редактирование существующего маршрута
	Удалить	Вх.	ID маршрута	int	void	Удаление маршрута
Расписание	Добавить	Вх.	ID маршрута, ID водителя, время отправления, время	Int, int, time, time,	Void	Добавление нового расписания

			прибытия, факт. Время отправления, факт. Время прибытия, нарушения	time, time, string		
	Редактировать	Вх.	ID расписания, ID маршрута, ID водителя, время отправления, время прибытия, факт. Время отправления, факт. Время прибытия, нарушения	int, int, int, time, time, time, time, string	Void	Редактирование существующего расписания
	Удалить	Вх.	ID расписания	int	void	Удаление расписания
База данных	Сохранить	Вх	Данные (список водителей, маршрутов, графиков)	List<Driver>, List<Route>, List<Schedule>	void	Сохранение данных в XML
	Загрузить	Вх.	Путь к XML	String	void	Загрузка данных из XML
	Искать	Вх.	Поле поиска, значение	String, String	List<Object>	Выполнение поиска по заданным критериям



## 2.4 Построение диаграммы программных классов

Диаграмма классов, представленная в этой UML-модели, иллюстрирует спецификации будущих программных классов и их взаимодействие в системе управления расписанием водителей, маршрутов и графиков. В основе диаграммы лежит объектная модель, где классы представляют сущности, такие как водители, маршруты, графики и взаимодействие с базой данных. Каждый класс включает три основных раздела: имя класса, атрибуты и методы.

Имя класса указывается в верхней части прямоугольника и представляет собой название сущности. В данном случае имена классов соответствуют их назначениям, например, Driver, Route, Schedule и другие.

Атрибуты описывают внутренние переменные класса, которые хранят данные. В UML-диаграмме атрибуты расположены во втором отделении прямоугольника и содержат информацию о типах данных, таких как String, int, Map, и других.

Диаграмма классов представлена на рис. 2.7.

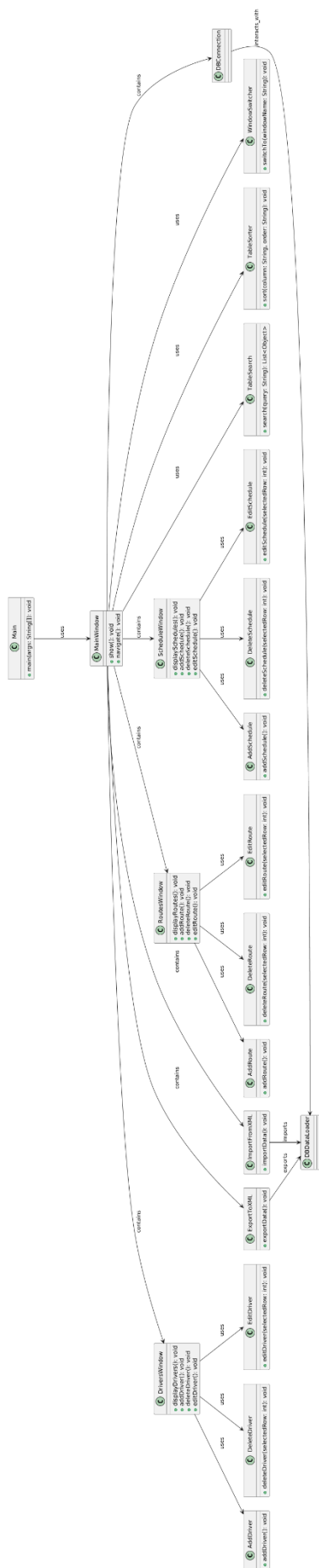


Рис. 2.7 – Диаграмма классов

## 2.5 Описание поведения ПК

Диаграмма последовательностей (sequence diagram) представляет собой схему, которая описывает взаимодействие между различными объектами системы в контексте выполнения определенного сценария. На диаграмме последовательностей показан порядок сообщений (или запросов) между объектами, а также их последовательность во времени. Эти диаграммы используются для представления динамического поведения системы, помогая визуализировать, как именно выполняется тот или иной процесс.

В данном случае, на диаграмме последовательностей показан процесс редактирования расписания в системе. Диаграмма отображает взаимодействие различных компонентов системы, таких как пользователь (User), интерфейс ScheduleWindow, класс EditSchedule и другие вспомогательные классы.

Описание действий:

### *1. Инициация действия:*

Пользователь нажимает кнопку "Редактировать" в интерфейсе ScheduleWindow. Это действие вызывает метод show() объекта EditSchedule.

### *2. Загрузка текущих данных:*

Метод show() загружает текущие данные выбранной строки из таблицы ScheduleWindow, такие как номер маршрута, ФИО водителя, номер автобуса, время отправления и др.

Для загрузки доступных маршрутов и водителей вызываются методы getRouteNumbers() и getDriverNames(), которые получают списки данных через взаимодействие с базой данных.

### *3. Отображение формы редактирования:*

Метод show() отображает форму редактирования с текущими данными расписания. Пользователь вводит новые значения или изменяет существующие.

#### 4. Валидация данных:

Метод `validate()` проверяет корректность введенных данных, таких как формат времени (HH:mm). Если данные некорректны, выбрасывается исключение, и пользователю отображается сообщение об ошибке.

#### 5. Обновление данных в базе:

Если данные корректны, метод `edit()` отправляет SQL-запрос через объект `DBConnection` для обновления данных расписания в базе.

#### 6. Обновление интерфейса:

После успешного выполнения запроса вызывается метод `loadSchedules()` для обновления данных в таблице интерфейса `ScheduleWindow` с использованием класса `DBDataLoader`.

#### 7. Отображение сообщения об успешном редактировании:

Пользователю отображается сообщение "Расписание успешно обновлено", завершая процесс.

Диаграмма последовательностей представлена на рис. 2.8

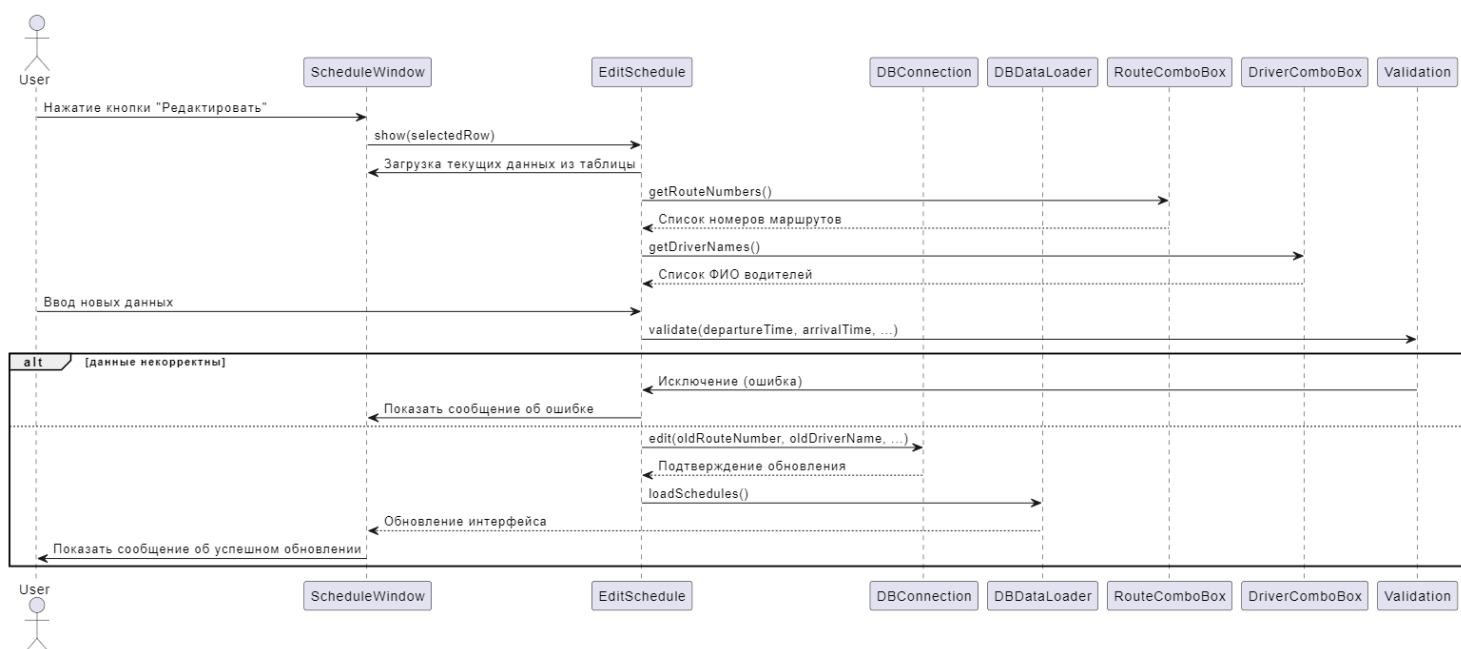


Рис. 2.8 – Диаграмма последовательностей для редактирования расписания

## 2.6 Построение диаграммы действий

Диаграмма действий (activity diagram) в данном случае предназначена для отображения шагов в процессе редактирования расписания в системе. Она помогает визуализировать последовательность действий и проверки, которые происходят при редактировании данных расписания. Диаграмма деятельности отображает как последовательные, так и возможные параллельные процессы, что позволяет лучше понять алгоритм работы системы.

В этой диаграмме действия представлены в виде прямоугольников, внутри которых указано, какое действие выполняется на каждом этапе, а переходы между ними — это стрелки, указывающие на порядок выполнения операций. Условные блоки (if) используются для отображения ветвлений, где проверяются условия, такие как корректность введенных данных или наличие необходимой информации в базе данных.

В этом примере представлена диаграмма деятельности для процесса добавления и удаления записи водителя из таблицы.

Диаграмма действий представлена на рис. 2.9.



Рис. 2.9 – Диаграмма действий

## 2.7 Реализация хранения данных многие-ко-многим

Связь многие-ко-многим — это связь, при которой одной записи из таблицы (А) могут соответствовать несколько записей из таблицы (В), и наоборот. В рамках проекта такая связь существует между таблицами Schedule, Routes и Drivers. Каждый маршрут может быть связан с несколькими записями в расписании, а каждый водитель может обслуживать несколько маршрутов.

В базе данных для реализации этой связи используется таблица Schedule, в которой хранятся внешние ключи для таблиц Routes и Drivers. Эти ключи ссылаются на уникальные идентификаторы маршрутов и водителей. Таким образом, таблица расписаний связывает два других компонента (маршрут и водителя) через их уникальные идентификаторы (ID). Это позволяет системе гибко управлять расписаниями, где один маршрут может быть связан с несколькими водителями, и один водитель может обслуживать несколько маршрутов (см. рис. 2.10).

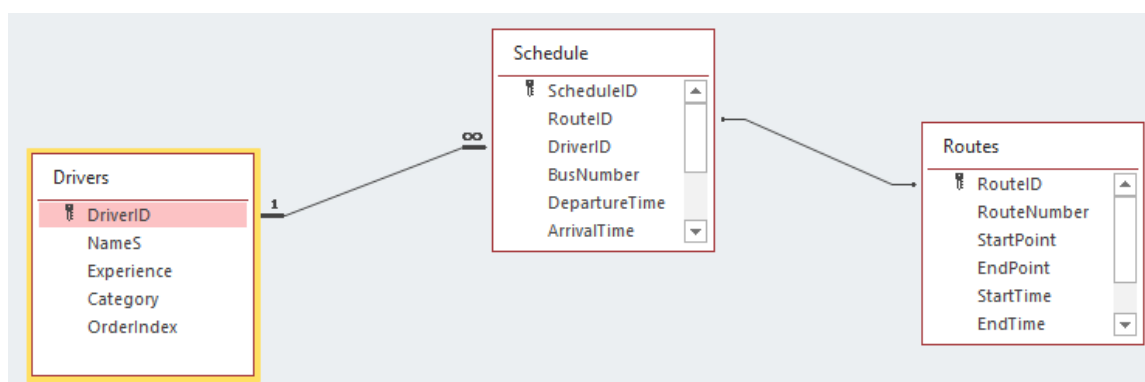


Рис. 2.10 – Реализация хранения данных многие-ко-многим

### **3. Руководство оператора**

#### **3.1 Назначение программы**

Программный комплекс «Диспетчеризация автобусного парка» предназначен для автоматизации процессов управления информацией о водителях, маршрутах и графиком движения. Программа реализована как настольное приложение с графическим интерфейсом на языке программирования Java, разработанным с использованием библиотеки Swing и взаимодействующим с локальной базой данных Access.

В рамках ПК «Диспетчеризация автобусного парка» пользователь может:

1. Просматривать подробную информацию о водителях, маршрутах и графиках движения;
2. Добавлять, редактировать и удалять данные о водителях, маршрутах и графиках движения;
3. Сортировать по возрастанию и убыванию данные о водителях, маршрутах и графиках движения;
4. Сохранять данные в XML документ, загружать данные из него;
5. Выполнять поиск информации по водителям, маршрутам и графикам движения по любому из имеющихся полей;
6. Создавать отчет в PDF или HTML формате по графикам движения маршрутов;
7. Выбирать нужную для работы базу данных;
8. Переключаться между экранными формами с водителями, маршрутами и графиком движения.

Программа ориентирована на использование базы данных автобусного парка, обеспечивая удобный, простой и понятный пользователю интерфейс, отображающий необходимую информацию для работы с ней.

#### **3.2 Условия выполнения программы**

Программный комплекс предназначен для функционирования под операционной системой Windows при поддержке MS Access, с установленной Java Runtime Environment (JRE).

*Минимальные системные требования:*

1. Процессор: Intel Core i3 или аналогичный;
2. Видеокарта: интегрированная (IntelHD Graphics или аналогичный);
3. ОЗУ: 256 Мб;
4. ПЗУ: 256 Мб свободного пространства;
5. Монитор: с поддержкой разрешения 1366x768;
6. Мышь и клавиатура.

Программные средства:

1. Операционная система Windows 7 и выше;
2. Java Runtime Environment (JRE) версии 8 или выше.

### **3.3 Описание задачи**

В ПК должна храниться информация о водителях, маршрутах и графике движения автобусов. Диспетчер автобусного парка может добавлять, изменять и удалять эту информацию. Ему могут потребоваться следующие сведения:

1. список водителей, их стаж работы и класс;
2. когда начинается или заканчивается движение автобуса на всех или отдельных маршрутах;
3. справка о графике движения автобусов и отчет о его нарушениях.

Обязательными требованиями при разработке кода ПК являются использование следующих конструкций языка Java:

1. закрытые и открытые члены классов;
2. наследование;
3. конструкторы с параметрами;
4. абстрактные базовые классы;



5. виртуальные функции;
6. обработка исключительных ситуаций;
7. динамическое создание объектов.

Для выполнения поставленной задачи разработана модель данных, которая включает основные объекты системы и связи между ними. Основные объекты:

1. Список водителей – хранит данные о ФИО водителей, их стаже и классе;
2. Маршруты – хранит данные о номере маршрута, начальной и конечной остановке, время начала и окончания движения автобусов, а также интервал
3. График движения – хранит данные о номере маршрута, ФИО водителя, номер автобуса, плановое время отправления и прибытия, фактическое время отправления и прибытия, а также нарушения, совершенные в ходе рейса, связывается с таблицами «Список водителей» и «Маршруты» через соответствующие идентификаторы (ID).

Данные надежно хранятся в базе данных MS Access.

Требования к коду ПК учтены при создании программных классов и непосредственном написании программы.

### **3.4 Входные и выходные данные**

Входные данные для программного комплекса «Диспетчеризация автобусного парка» вводятся пользователем через графический интерфейс программы, а также могут быть загружены из XML документов. Исходные данные являются характеристиками (атрибутами) информационных объектов и вводятся через диалоговые окна и (или) выбираются из предложенных значений.

1. Список водителей:
  - ФИО водителя
  - Стаж работы (числовое значение)

- Класс
2. Маршруты:
- Номер маршрута (числовое значение)
  - Начальная остановка
  - Конечная остановка
  - Время начала (формат времени HH:mm)
  - Время окончания (формат времени HH:mm)
  - Интервал (числовое значение)

3. График движения

- Номер маршрута (выпадающий список с номерами маршрутов из таблицы «Маршруты»)
- ФИО водителя (выпадающий список с ФИО водителей из таблицы «Список водителей»)
- Номер автобуса
- Время отправления (формат времени HH:mm)
- Время прибытия (формат времени HH:mm)
- Фактическое время отправления (формат времени HH:mm)
- Фактическое время прибытия (формат времени HH:mm)
- Нарушения

Данные, которые не были предложены программой при помощи выпадающих списков, вводятся вручную. Также данные могут быть добавлены из XML документа, однако в таблице «График движения» новая запись будет добавлена, если существуют такие ФИО водителя и номер маршрута в соответствующих таблицах, в целях избегания ошибок при отсутствующих ФИО водителя и номере маршрута).

Выходные данные для программного комплекса «Диспетчеризация автобусного парка» представлены в табличной форме в графическом

интерфейсе. Каждая таблица отображает описание необходимых информационных объектов, выполненное посредством их характеристик:

1. Список водителей:

- ФИО водителя;
- Стаж работы;
- Класс.

2. Маршруты:

- Номер маршрута;
- Начальная остановка;
- Конечная остановка;
- Время начала;
- Время окончания ;
- Интервал.

3. График движения

- Номер маршрута (связано с таблицей «Маршруты» по ID);
- ФИО водителя (связано с таблицей «Список водителей» по ID);
- Номер автобуса;
- Время отправления;
- Время прибытия;
- Фактическое время отправления;
- Фактическое время прибытия;
- Нарушения.

### **3.5 Выполнение программы**

#### *3.5.1 Подготовка к запуску*

Для работы с программой требуется база данных MS Access. В исходном коде предусмотрен выбор необходимой базы данных, а также создание таблиц по загруженной базе данных.

### 3.5.2 Запуск программы

При запуске программы отображается экранная форма «Главное меню» (см. рис. 3.1). При первом запуске, пользователь должен выбрать базу данных для работы, нажав на кнопку «Выбор базы данных». При последующих запусках, программа будет автоматически подключаться к выбранной базе данных ранее. Сменить базу данных, с которой будет происходить работа, можно при помощи этой же кнопки.

Далее после выбора базы данных, если соединение будет успешно установлено, пользователь может открыть любую из трех таблиц: «Список водителей», «Маршруты» или «График движения».

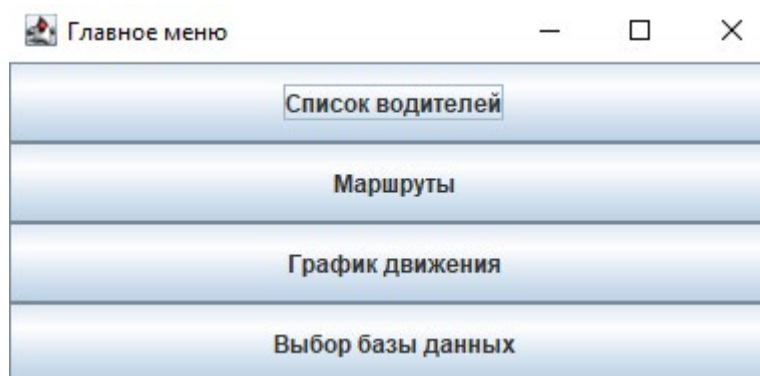


Рис. 3.1 - Главное окно

### 3.5.3 Выполнение основных функций

Рассмотрим таблицу «Список водителей» (рис 2.2). В ней отображается информация: ФИО водителей, их стаж и класс.

Рассмотрим основные функции:

#### 1. Добавление водителя:

При нажатии на кнопку «Добавить», появляется экранная форма для добавления водителя (см. рис. 3.2), в которую пользователь вводит данные о новом водителе. Если в поле «стаж» введено не число или такой водитель уже существует (см. рис. 3.3), пользователь получает сообщение об ошибке и запись

не добавляется. Если все данные корректны – запись добавляется в таблицу и базу данных.

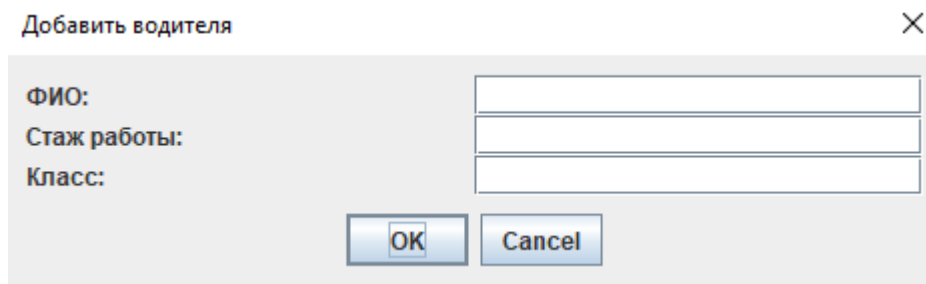


Рис 3.2 – Экранная форма для добавления водителя

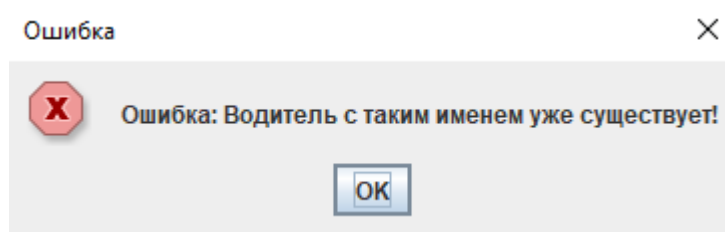


Рис 3.3 – Сообщение об ошибке при добавлении имеющегося водителя

## 2. Редактирование водителя:

Для начала нужно выбрать запись, которую нужно изменить, кликнув на нее. При нажатии на кнопку «Редактировать», появляется экранная форма для редактирования записи (см. рис. 3.4). Пользователь может изменить любое поле. При изменении ФИО водителя на имеющееся или если стаж не число, пользователь получит ошибку, как при добавлении водителя. Если все данные корректны – изменения вносятся в таблицу.

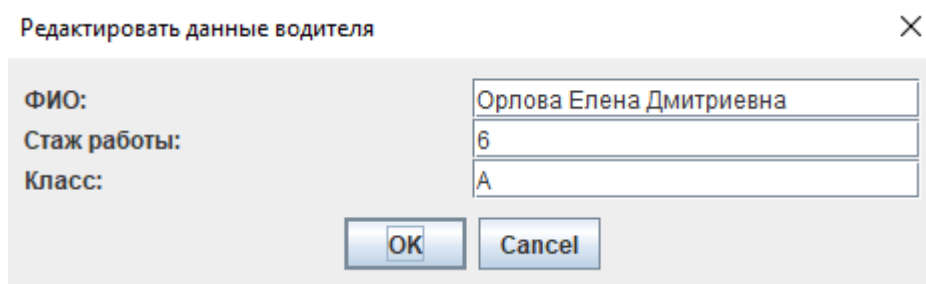


Рис. 3.4 – Экранная форма для редактирования водителя

## 3. Удаление водителя:

Аналогично редактированию, пользователь сначала выбирает нужную запись, кликнув на нее. Далее, при нажатии на кнопку «Удалить», появляется подтверждение об удалении (см. рис. 3.5). Запись удаляется после подтверждения пользователем.

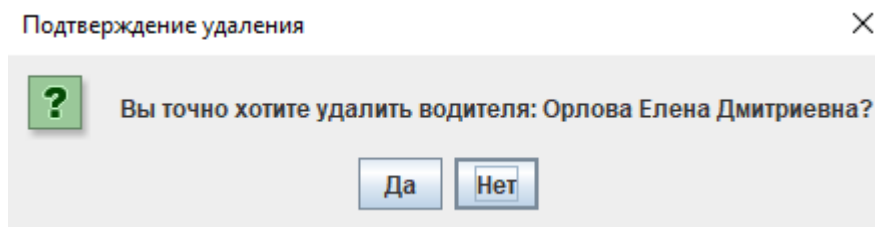


Рис 3.5 – Подтверждение удаления водителя

#### 4. Сортировка по возрастанию и убыванию:

Для сортировки пользователь выбирает поле, по которому нужно произвести сортировку (см. рис. 3.6). Далее нужно выбрать как будут отсортированы данные, по возрастанию или по убыванию и нажать на кнопку.

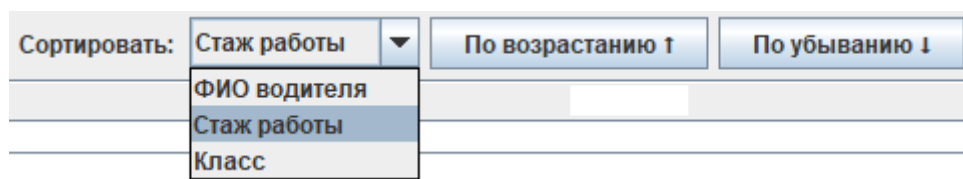


Рис. 3.6 – Выбор поля для сортировки

Результат сортировки по полю «стаж» по возрастанию на рис. 3.7.

Список водителей

Сохранить Загрузить Главное меню Таблица: Список водителей

Добавить Редактировать Удалить Сортировать: Стаж работы По возрастанию По убыванию

ФИО водителя	Стаж работы	Класс
Попова Дарья Викторовна	1	A
Лебедева Светлана Николаевна	2	B
Попова Дарья Викторовн	2	A
Соколова Анна Сергеевна	2	D
Смирнов Николай Александрович	3	B
Николаев Артем Валерьевич	4	D
Романов Алексей Андреевич	4	C
Федоров Андрей Сергеевич	5	D
Иванов Иван Романович	5	A
Иванов Иван Иванович	6	B
Орлова Елена Дмитриевна	6	A
Васильев Сергей Борисович	6	C
Шевченко Валентин Борисович	7	B
Кузнецова Екатерина Владимировна	8	C
Григорьева Татьяна Сергеевна	8	D
Михайлова Ольга Павловна	9	B
Соловьев Владислав Александрович	9	C
Ковалев Вадим Андреевич	10	A
Сидоров Алексей Петрович	10	C
Петрова Мария Ивановна	75	A

Критерий поиска: ФИО водителя Значение: Поиск Показать все записи

Рис. 3.7 – Таблица после сортировки по полю «стаж работы» по возрастанию

### 5. Поиск по критерию:

Для поиска пользователь выбирает поле, в котором будет производиться поиск(см рис. 3.8), заполняет поле «значение» и нажимает на кнопку поиск, после чего отобразятся все подошедшие под критерии записи.

Критерий поиска: ФИО водителя Значение:  Поиск Показать все записи

ФИО водителя  
Стаж работы  
Класс

Рис. 3.8 – Выбор поля для поиска

Результат поиска по полю «класс» со значением «А» на рис. 3.8.

Список водителей

Сохранить Загрузить Главное меню Таблица: Список водителей

Добавить Редактировать Удалить Сортировать: Стаж работы По возрастанию По убыванию

ФИО водителя	Стаж работы	Класс
Попова Дарья Викторовна	1	A
Иванов Иван Романович	5	A
Орлова Елена Дмитриевна	6	A
Ковалев Вадим Андреевич	10	A
Петрова Мария Ивановна	75	A

Критерий поиска: Класс Значение: A Поиск Показать все записи

Рис. 3.8 – Результат поиска

После работы с найденными значениями, пользователь может нажать на кнопку «Показать все записи», тем самым будут отображены все записи, имеющиеся в таблице.

#### *6. Сохранение данных:*

Пользователь может нажать на кнопку «Сохранить» для сохранения всех имеющихся данных в XML документ. При нажатии, открывается окно с выбором места сохранения и названия файла (см рис. 3.9). Файл сохраняется.

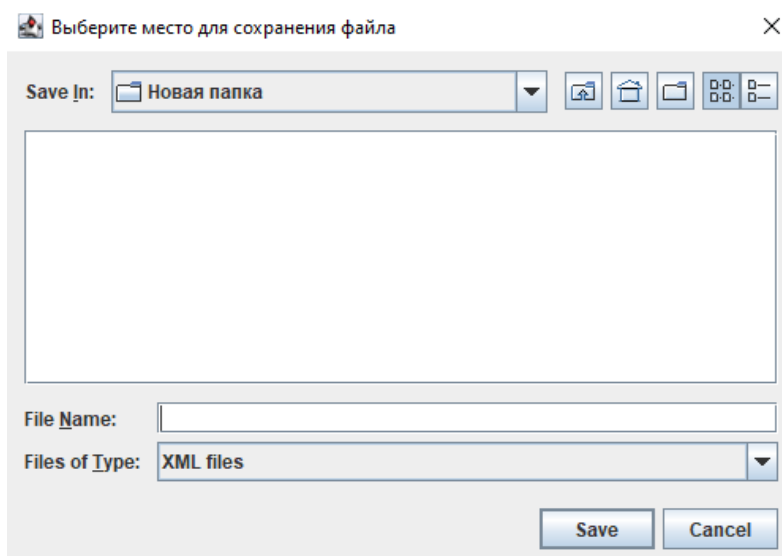


Рис. 3.9 – Сохранение XML документа

#### *7. Загрузка данных:*

Пользователь может загрузить данные из XML документа, нажав на кнопку «Загрузить». При нажатии, открывается окно для выбора файла, аналогичное, как в п.6 «Сохранение данных». Файл выбирается и данные загружаются в таблицу и базу данных.

#### *8. Смена таблицы:*

При нажатии на выпадающий список «Таблица», отображаются все три таблицы, с которыми может взаимодействовать пользователь: «Список водителей», «Маршруты» и «График движения» (см. рис. 3.10). После выбора нужной таблицы, пользователю открывается соответствующая таблица.



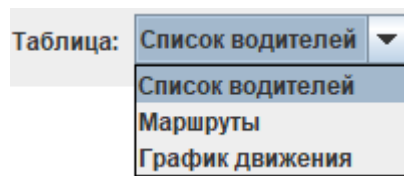


Рис. 3.10 – Выбор таблицы

#### 9. Переход в главное меню:

При нажатии на кнопку «Главное меню», таблица закрывается и отображается окно «Главное меню» (рис 3.1).

В таблицах «Маршруты» и «График движения» основные функции аналогичны функциям таблицы «Список водителей». Отличие только в функциях добавления и редактирования в таблице «График движения». Также там есть функция создания отчета. Рассмотрим их подробнее.

#### 10. Добавление расписания в таблицу «График движения»:

Выполнение аналогично добавлению водителю, однако здесь пользователь выбирает номер маршрута и ФИО водителя, имеющихся в таблицах «Список водителей» и «Маршруты» (см рис. 3.11).

Рис. 3.11 – Добавление расписания в таблице «График движения»

### 11. Редактирование записи в таблице «График движения»:

Происходит аналогично добавлению, пользователь также может менять все поля, только первые два поля – выпадающие списки, в которых нужно выбрать новые данные.

### 12. Создание отчета:

Пользователь может создать отчет в формате PDF или HTML формате (рис. 3.12), нажав на кнопку «Создать отчет». После выбора формата пользователю открывается окно с выбором места и названия для сохранения. Место и название выбираются, отчет сохраняется.

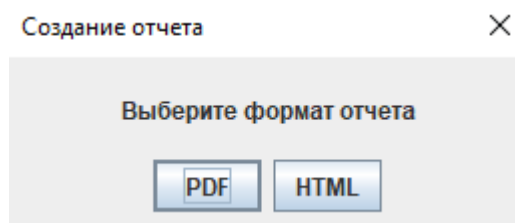
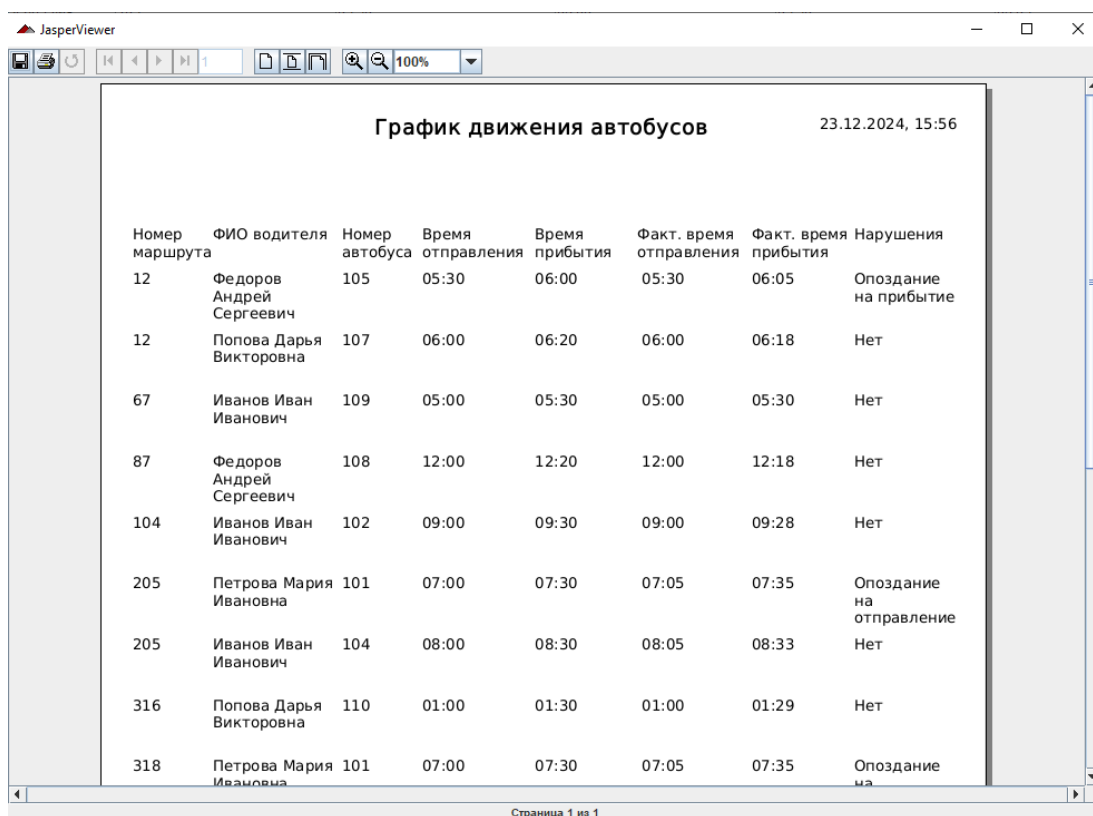


Рис. 3.12 – Окно выбора формата отчета

После сохранения отчета, он показывается в программе (см рис. 3.13).

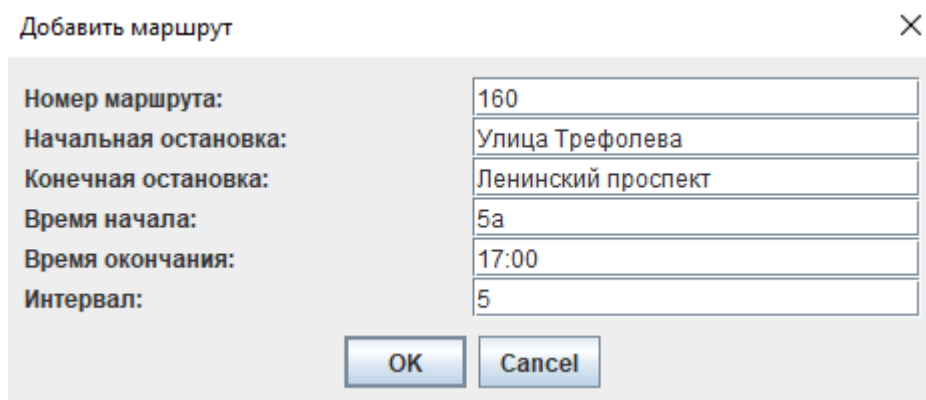


Номер маршрута	ФИО водителя	Номер автобуса	Время отправления	Время прибытия	Факт. время отправления	Факт. время прибытия	Нарушения
12	Федоров Андрей Сергеевич	105	05:30	06:00	05:30	06:05	Опоздание на прибытие
12	Попова Дарья Викторовна	107	06:00	06:20	06:00	06:18	Нет
67	Иванов Иван Иванович	109	05:00	05:30	05:00	05:30	Нет
87	Федоров Андрей Сергеевич	108	12:00	12:20	12:00	12:18	Нет
104	Иванов Иван Иванович	102	09:00	09:30	09:00	09:28	Нет
205	Петрова Мария Ивановна	101	07:00	07:30	07:05	07:35	Опоздание на отправление
205	Иванов Иван Иванович	104	08:00	08:30	08:05	08:33	Нет
316	Попова Дарья Викторовна	110	01:00	01:30	01:00	01:29	Нет
318	Петрова Мария Ивановна	101	07:00	07:30	07:05	07:35	Опоздание на отправление

Рис 3.13 – Автоматическое отображение отчета после создания

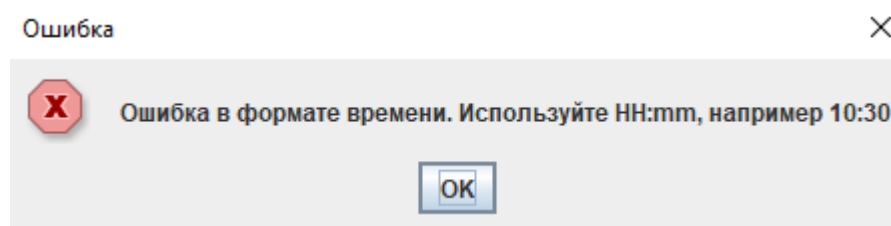
### 3.6 Проверка программы

Функциональность программы была протестирована на различных входных данных, в том числе некорректных. Например, при вводе неверного формата времени в таблице «Маршруты» (см. рис. 3.14), пользователь получает сообщение об ошибке (см. рис. 3.15), программа не добавляет некорректную запись и продолжает свою работу.



Добавить маршрут	✕
Номер маршрута:	160
Начальная остановка:	Улица Трефолева
Конечная остановка:	Ленинский проспект
Время начала:	5a
Время окончания:	17:00
Интервал:	5
<div>OK Cancel</div>	

Рис. 3.14 – добавление некорректной записи




Ошибка	✕
<div> Ошибка в формате времени. Используйте HH:mm, например 10:30</div>	
<div>OK</div>	

Рис. 3.15 – Предупреждение об ошибке

Таким же образом были проверены все функции всех трех таблиц.

Так же все функции были проверены в различных ситуациях, например, редактирование и удаление после сортировки таблицы и поиска значений должно происходить с соответствующими записями.

### 3.7 Сообщения оператору

1. При отсутствии базы данных по выбранному пути оператор получает сообщение об ошибке (см. рис. 3.16). В таком случае нужно корректно выбрать базу данных.

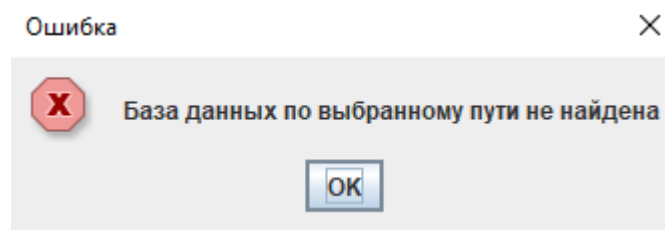


Рис. 3.16 – База данных не найдена

2. При вводе некорректных значений при добавлении или редактировании записей, а также в поле поиска, выводится сообщение об ошибке (см. рис. 3.17, 3.12, 3.15). В этой ситуации программа не завершает свою работу, а некорректная и (или) существующая запись не добавляется.

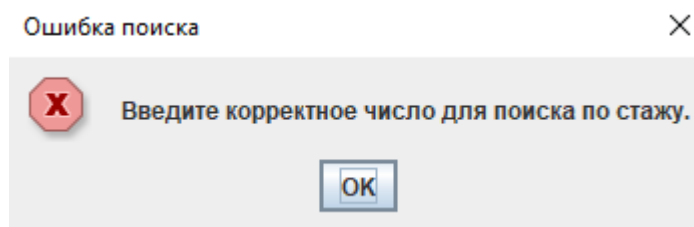


Рис. 3.17 – Ошибка при поиске

## **Заключение**

В результате проделанной работы разработан программный комплекс «Диспетчеризация автобусного парка», предназначенный для автоматизации управления данными о водителях, маршрутах и графиках движения автобусов. Также было создано руководство оператора, описывающее основные функции и порядок работы с системой.

В процессе проектирования выполнены следующие этапы: описание вариантов использования ПК, разработка прототипа пользовательского интерфейса, создание объектной модели данных, построение диаграммы классов и диаграммы действий, а также описание поведения системы.

В ходе реализации предусмотрены функциональные возможности добавления, редактирования, удаления, сортировки, поиска данных о водителях, маршрутах и графиках движения. Реализован экспорт и импорт данных при помощи XML документов, а также формирование отчетов в PDF и HTML форматах.

Курсовой проект удовлетворяет поставленным требованиям, обеспечивает удобство и стабильность работы программного комплекса по управлению автобусным парком.