

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РФ

Федеральное государственное бюджетное образовательное учреждение
высшего образования
«ПЕНЗЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»
ПОЛИТЕХНИЧЕСКИЙ ИНСТИТУТ

Факультет
вычислительной техники

Кафедра
МОиПЭВМ

Направление подготовки 09.03.02 «Информационные системы и технологии»

Направленность Информационные системы и технологии

БАКАЛАВРСКАЯ РАБОТА

на тему

**Информационная система управления учебным
процессом автошколы**

Студент _____ Королев Алексей Сергеевич
(подпись, дата) (ФИО полностью)

Руководитель _____ Балашова И.Ю.
(подпись, дата) (фамилия, инициалы)

Нормоконтролер _____ Такташкин Д.В.
(подпись, дата) (фамилия, инициалы)

Работа допущена к защите (протокол заседания кафедры от 09.06.2025 г. № 12)

Заведующий кафедрой _____ Козлов А.Ю.
(подпись) (фамилия, инициалы)

Работа защищена с отметкой _____ (протокол заседания ГЭК от _____ № _____)

Секретарь ГЭК _____ Попова Н.А.
(подпись) (фамилия, инициалы)

Пенза, 2025

Федеральное государственное бюджетное образовательное учреждение
высшего образования
ПЕНЗЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ПОЛИТЕХНИЧЕСКИЙ ИНСТИТУТ

Факультет
Вычислительной техники

Кафедра
МО и ПЭВМ

«Утверждаю»
Заведующий кафедрой
А.Ю. Козлов
«20» декабря 2024 г

**ЗАДАНИЕ
НА ВЫПУСКНУЮ КВАЛИФИКАЦИОННУЮ РАБОТУ
БАКАЛАВРА**

1. Студент Королев Алексей Сергеевич гр. 21ВИ1 факультета ВТ
<Фамилия, имя, отчество полностью>

2. Тема работы Информационная система управления учебным процессом автошколы

Тема утверждена приказом ПГУ № _____ от " _____ " _____ 20 _____

3. Руководитель работы к.т.н., доцент, доцент кафедры «МО и ПЭВМ» Балашова И.Ю.

4. Задание на работу (назначение разработки, исходные данные и т.п.)

1) *Назначение: разработка информационной системы управления учебным процессом автошколы.*

2) *Среда выполнения: Windows 11.*

Языковые средства разработки: C#, HTML, CSS, JavaScript, SQL, UML.

Инструментальные средства: Visual Studio 2022, SQL Server Management Studio 20, Visual Paradigm 17.0.

Технологии разработки: ООП, многослойная архитектура.

3) *Основные функции приложения:*

а) авторизация с разграничением доступа;

б) управление расписанием занятий;

в) запись на занятия;

г) отмена записи на занятие;

д) просмотр информации об инструкторах;

е) просмотр информации о студентах;

ж) управление аккаунтами (создание, редактирование, удаление).

5. Перечень подлежащих разработке вопросов:

1) <i>Анализ предметной области и постановка задачи;</i>
2) <i>Проектирование;</i>
3) <i>Реализация;</i>
4) <i>Контроль качества программного обеспечения;</i>
5) <i>Оформление пояснительной записки.</i>

6. Календарный график выполнения работы

№ п/п	Наименование этапов работы	Объем работы	Срок выполнения	Подпись руководителя
1	Анализ предметной области и требований к системе	10%	08.04.2025 – 14.04.2025	
2	Проектирование программных средств	20%	15.04.2025 – 25.04.2025	
3	Реализация программных средств	30%	26.04.2025 – 12.05.2025	
4	Тестирование программных средств	20%	13.05.2025 – 22.05.2025	
5	Оформление пояснительной записки	20%	23.05.2025 – 03.06.2025	

Дата выдачи задания «20» декабря 2024г.

Руководитель бакалаврской работы _____
(подпись, дата)

Задание к исполнению принял студент _____
(подпись, дата)

Работу к защите допустить
Декан ФВТ _____ Л.Р. Фионова
(подпись, дата)

Реферат

Пояснительная записка содержит 130 листов, 65 рисунков, 22 таблицы, 22 использованных источника и 7 приложений.

ИНФОРМАЦИОННАЯ СИСТЕМА, ВЕБ-ПРИЛОЖЕНИЕ,
МНОГОСЛОЙНАЯ АРХИТЕКТУРА, REST API, UML, VISUAL STUDIO 2022,
C#, HTML, CSS, JS, БАЗА ДАННЫХ, SQL SERVER.

Объект исследования – информационные системы управления учебным процессом в автошколе.

Цель выпускной квалификационной работы – разработать информационную систему управления учебным процессом автошколы.

Технологии разработки – разработка всех средств информационной системы управления учебным процессом автошколы производилась в среде разработки Visual Studio 2022 с использованием фреймворка ASP.NET Core. Разработка клиентской части приложения проводилась на языках разметки HTML, CSS и языке программирования JavaScript. Разработка серверной части приложения производилась на языке программирования C#. Разработка базы данных проводилась в СУБД SQL Server Management Studio 20.

Среда исполнения – Windows 11, веб-браузер.

Результаты работы – в ходе выполнения выпускной квалификационной работы была спроектирована и реализована информационная система управления учебным процессом автошколы. Все требования к разработке системы были соблюдены.

					ПГУ 09.03.02 – 08BP211.15 ПЗ			
Изм.	Лист	№ докум.	Подпись	Дата				
Разраб.		Королев А.С.			Информационная система управления учебным процессом автошколы. Пояснительная записка	Лит.	Лист	Листов
Пров.		Балашова И.Ю.					4	130
Реценз.						Группа 21ВИ1		
Н. контр.		Такташкин Д.В						
Утв.		Козлов А.Ю.						

Содержание

Введение.....	7
1 Анализ предметной области и постановка задачи на разработку информационной системы управления учебным процессом автошколы	8
1.1 Исследование предметной области.....	8
1.2 Анализ информационных систем аналогичного назначения	9
1.2.1 Сервис для автоматизации и управления автошколой «Кенео».....	9
1.2.2 Онлайн-система организации практических занятий «SignMeUp»	10
1.2.3 Облачный сервис «Автошкола Контроль».....	11
1.2.4 Сравнение систем управления обучением в автошколах	12
1.3 Анализ функциональных требований.....	13
1.4 Анализ нефункциональных требований	20
1.4.1 Общие нефункциональные требования.....	20
1.4.2 Технические средства реализации требований	21
1.5 Планирование разработки и оценка бюджета.....	23
2 Проектирование информационной системы управления учебным процессом автошколы.....	27
2.1 Архитектура приложения.....	27
2.2 Проектирование структур данных	29
2.2.1 Концептуальная модель базы данных	29
2.2.2 Логическая модель базы данных.....	31
2.3 Проектирование программных средств.....	32
2.4 Проектирование пользовательского интерфейса	36
3 Реализация информационной системы управления учебным процессом автошколы.....	38
3.1 Реализация базы данных	38
3.2 Реализация серверной части информационной системы управления учебным процессом автошколы	40

3.3 Реализация клиентской части информационной системы управления учебным процессом автошколы	44
3.4 Реализация интерфейса информационной системы управления учебным процессом автошколы	46
3.4.1 Реализация интерфейса студента	48
3.4.2 Реализация интерфейса инструктора.....	57
3.4.3 Реализация интерфейса администратора	62
4 Контроль качества программного обеспечения.....	72
4.1 Выбор режима тестирования	72
4.2 Тестирование серверной части информационной системы управления учебным процессом автошколы	73
4.3 Тестирование клиентской части информационной системы управления учебным процессом автошколы	74
4.4 Метрики кода.....	79
Заключение	81
Список использованных источников	83
Приложение А (справочное) Глоссарий	86
Приложение Б (обязательное) Листинг серверной части системы.....	89
Приложение В (обязательное) Листинг клиентской части системы	99
Приложение Г (обязательное) Листинг модульных тестов.....	104
Приложение Д (рекомендуемое) Атрибуты сущностей базы данных.....	107
Приложение Е (рекомендуемое) Графическая часть	113
Приложение Ж (рекомендуемое) Результаты функционального тестирования.....	119

Введение

Информационные технологии прочно интегрировались в повседневную деятельность, они обеспечивают более эффективное взаимодействие, выполнение рабочих процессов и принятие решений [1]. Современные информационные технологии широко применяются во многих сферах, включая образование. Автошколы, как образовательные учреждения, нуждаются в удобных и эффективных инструментах для организации учебного процесса.

Актуальной задачей является создание информационной системы (ИС), которая упростит управление учебным процессом и будет понятной для всех пользователей. Разработка такой системы позволит не только повысить эффективность работы учреждения, но и улучшить качество предоставляемых услуг и автоматизировать основные процессы обучения в автошколе, что делает ее актуальной в современных условиях.

Предметом работы является автоматизация процесса управления учебным процессом автошколы. Целью выпускной квалификационной работы является разработка информационной системы управления учебным процессом автошколы.

Для достижения цели необходимо выполнить следующие задачи:

- проанализировать предметную область и определить требования к разрабатываемой системе;
- провести планирование разработки и расчет бюджета;
- разработать архитектуру и структуру базы данных;
- реализовать ИС с учетом поставленных требований;
- провести тестирование системы и оценить ее качество.

Реализация этих задач поможет создать удобную систему управления учебным процессом автошколы.

1 Анализ предметной области и постановка задачи на разработку информационной системы управления учебным процессом автошколы

1.1 Исследование предметной области

Автошкола – это учебное заведение, которое занимается подготовкой будущих водителей различных категорий. В процессе обучения участвуют студенты, инструкторы и сотрудники администрации. Учебный процесс включает теоретические занятия, практические уроки вождения, составление расписаний, учет посещаемости и оценок, а также взаимодействие между всеми участниками.

На практике автошколы часто сталкиваются с трудностями в организации обучения. Документы ведутся вручную, что требует много времени и может приводить к ошибкам. Планирование занятий усложняется, связь между студентами и инструкторами часто осуществляется неэффективно, а администрация тратит много ресурсов на рутинные задачи. Все это влияет на качество обучения и общую работу учреждения.

Применение современных информационных технологий помогает решить эти проблемы. Автоматизация позволяет упростить рутинные процессы, обеспечить быстрый обмен информацией и сделать взаимодействие между студентами, инструкторами и администрацией более удобным и понятным. Внедрение специальной информационной системы поможет хранить и обрабатывать данные в одном месте, быстро изменять расписание, автоматически распределять занятия и улучшить организацию учебного процесса [2].

Целью разработки является создание веб-приложения для автоматизации ключевых функций автошколы. Система должна включать регистрацию и авторизацию пользователей, управление расписанием, запись на занятия, учет данных и возможность обратной связи. Важно, чтобы она была простой в использовании, надежной, быстрой и готовой к дальнейшему расширению.

Таким образом, автоматизация учебного процесса в автошколе — это актуальное и полезное решение, которое помогает повысить эффективность работы, качество обучения и удовлетворенность всех участников процесса.

1.2 Анализ информационных систем аналогичного назначения

1.2.1 Сервис для автоматизации и управления автошколой «Кенео»

Кенео представляет собой облачную систему, ориентированную на полную автоматизацию всех направлений деятельности автошколы — от регистрации учеников и преподавателей до формирования расписания и финансовой отчетности. Интерфейс системы представлен на рисунке 1.

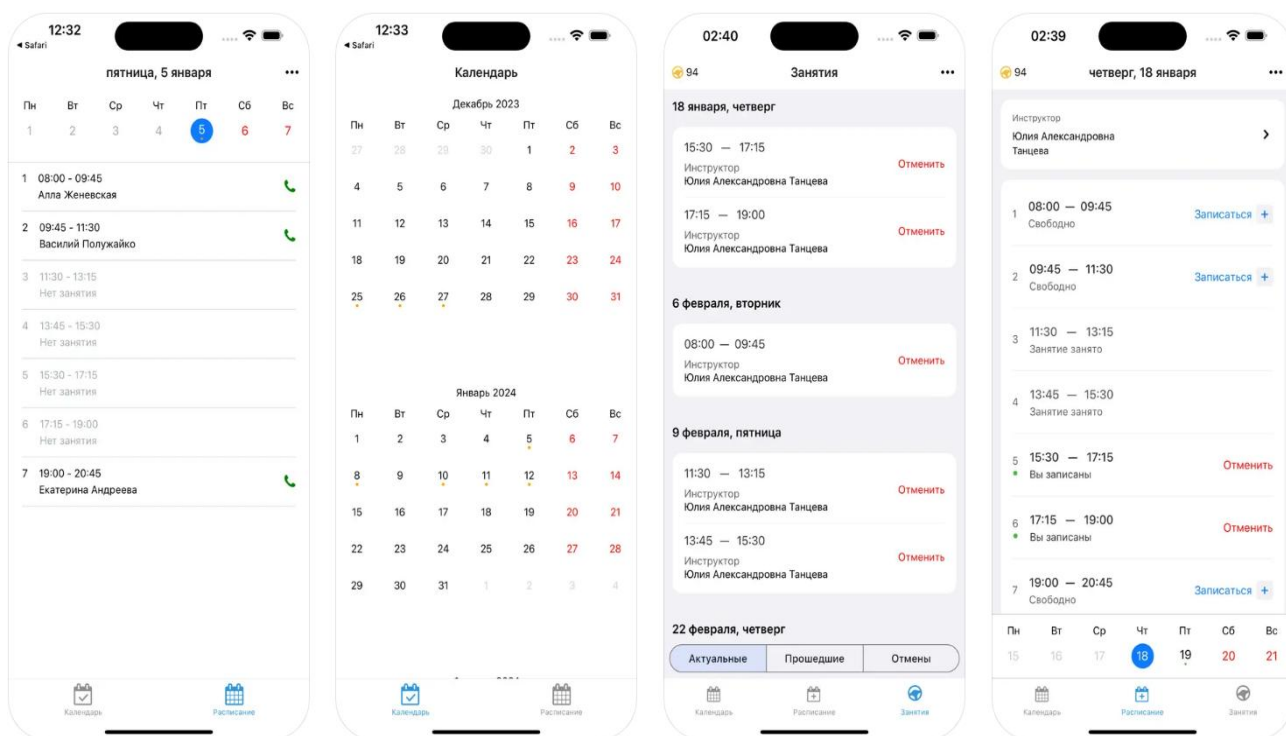


Рисунок 1 – Интерфейс приложения «Кенео»

Среди ее сильных сторон стоит отметить комплексность и продуманную интеграцию всех ключевых процессов внутри одной платформы. Система включает встроенную CRM, возможности онлайн-записи, автоматические уведомления, а также личные кабинеты для разных категорий пользователей.

Однако, при всех достоинствах, данное решение имеет и ряд ограничений. Во-первых, модель платной подписки может быть обременительной для небольших или начинающих автошкол. Во-вторых, система не очень гибкая, потому что рассчитана на типовые процессы и плохо подходит для индивидуальных настроек. Кроме того, отсутствие открытого API затрудняет подключение сторонних сервисов.

1.2.2 Онлайн-система организации практических занятий «SignMeUp»

«SignMeUp» – это специализированная система, фокусирующаяся на автоматизации процесса записи на практические занятия с инструктором. Подходит как дополнение к существующим системам учета в автошколах. Интерфейс приложения представлен на рисунке 2.

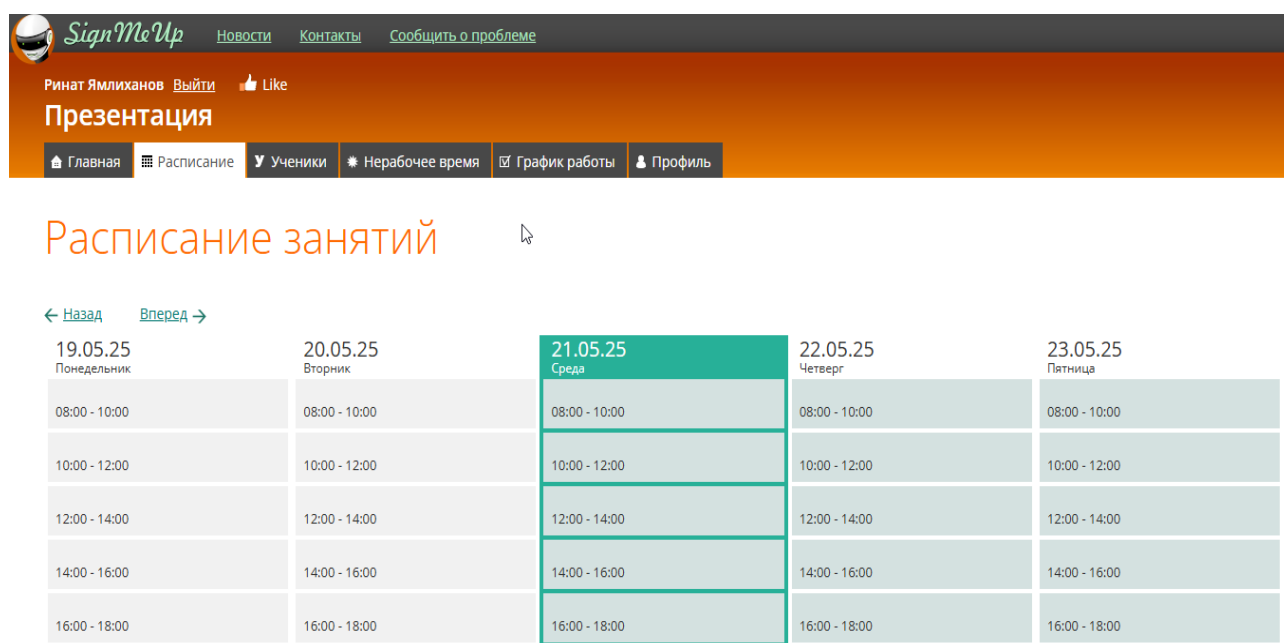


Рисунок 2 – Интерфейс приложения «SignMeUp»

Система имеет простой, интуитивно понятный интерфейс, подбор времени занятий и автоматическое оповещение. Это делает систему удобной в ежедневной работе инструкторов и учеников. Также стоит отметить наличие

разграничения прав доступа: отдельные интерфейсы предусмотрены для каждой категории пользователей.

Однако из-за ограниченного набора функций SignMeUp нельзя назвать универсальным решением. В системе нет возможности проводить теоретические занятия, оформлять документы, учитывать оплаты и следить за общим прогрессом обучения. Добавить недостающие функции тоже нельзя, поэтому использовать платформу как основную систему для всей автошколы не получится.

1.2.3 Облачный сервис «Автошкола Контроль»

«Автошкола Контроль» позиционируется как современное цифровое решение, акцентированное на мобильные устройства. Платформа предлагает все базовые функции: учет учеников, инструкторов, расписаний, оплат, электронного документооборота, а также встроенные тесты и модули аналитики. Основной особенностью является наличие брендируемого мобильного приложения, которое обеспечивает интерактивное взаимодействие между автошколой и учащимися. Интерфейс приложения представлен на рисунке 3.

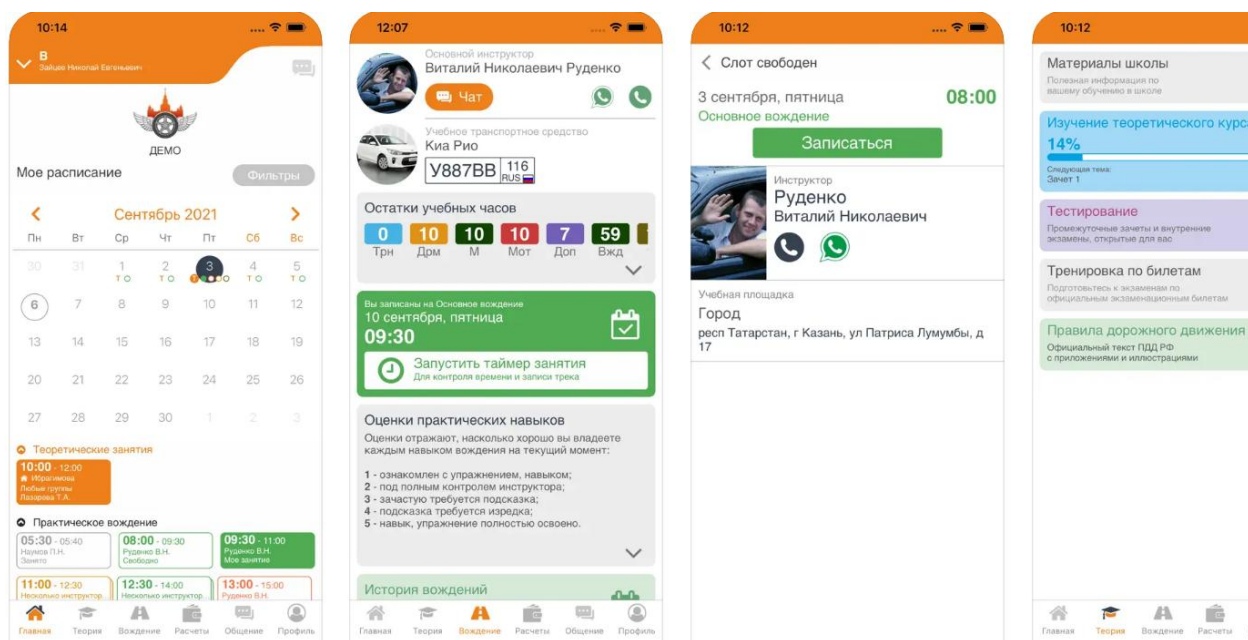


Рисунок 3 – Интерфейс приложения «Автошкола Контроль»

Тем не менее, столь сильная ориентация на мобильную платформу может обернуться определенными трудностями. В частности, взрослое поколение пользователей могут испытывать сложности при работе с гаджетами. Представители платформы в открытую утверждают, что приоритетом является работа с молодежью, тогда как удобство использования со стороны возрастных учеников отходит на второй план. Также наблюдается дефицит десктопных интерфейсов и отсутствие возможности работы в офлайн-режиме, что может ограничивать использование системы в некоторых условиях.

1.2.4 Сравнение систем управления обучением в автошколах

Для проведения сравнительного анализа информационных систем для автошкол был определен ряд критериев, на основе которых оценивались существующие решения. Система оценивалась по следующей шкале:

- 0 баллов: функция отсутствует или выполнена неудовлетворительно;
- 1 балл: функция реализована частично или со значительными ограничениями;
- 2 балла: функция реализована полноценно и качественно.

Результаты оценки представлены в таблице 1.

Таблица 1 – Сравнение систем аналогичного назначения

Критерий	Keneo	SignMeUp	Автошкола Контроль
Наличие всех основных функций для автошколы (управление расписанием, учет занятий, платежей, тестов, пользователей)	2	1	2
Организация и учет занятий по теории и практике (расписание, посещаемость, прогресс учеников)	2	1	2
Встроенная работа с документами	2	0	2
Личный кабинет ученика и инструктора	2	2	2

Продолжение таблицы 1

Критерий	Keneo	SignMeUp	Автошкола Контроль
Удобство использования на смартфонах	2	1	2
Гибкость и возможность адаптации под нужды конкретной школы	1	0	1
Простота и удобство интерфейса для пожилых пользователей (минимум действий, ПК-доступ)	1	0	0
Итого	12	5	11

Проведенный сравнительный анализ показал, что ни одно из существующих решений не сочетает в себе одновременно широкий функционал, гибкость настройки, возможность локального использования и адаптацию под разнородную аудиторию, поэтому разрабатываемая информационная система для автошкол должна быть гибкой, простой в использовании и недорогой. Кроме того, нужно предусмотреть возможность расширения системы в будущем.

1.3 Анализ функциональных требований

В процессе анализа предметной области и задач на разработку информационной системы управления учебным процессом автошколы были выделены следующие основные функциональные требования:

- авторизация пользователей (администратор, студент, инструктор);
- просмотр и редактирование профиля пользователя;
- просмотр студентами расписания занятий и возможность записаться на свободное занятие;
- просмотр инструкторами расписания занятий и возможность создавать или отменять занятия;
- просмотр студентами списка инструкторов;
- просмотр студентами своих забронированных занятий;
- просмотр инструкторами своих созданных занятий;

- просмотр инструкторами списка студентов;
- возможность пользователям (студентам и инструкторам) оставлять отзывы друг о друге;
- возможность администратора просматривать списки всех пользователей системы (студентов, инструкторов, администраторов);
- возможность администратора добавлять новых пользователей и управлять учетными записями;
- просмотр и управление списком автошкол (для администратора).

В соответствии с указанными требованиями были построены диаграммы вариантов использования UML, отражающие взаимодействие пользователей с системой.

Диаграмма вариантов использования – диаграмма, описывающая, какой функционал разрабатываемой программной системы доступен каждой группе пользователей. Диаграмма вариантов использования служит для описания функциональности и поведения проектируемой системы. На ней видно, кто взаимодействует с системой и какие действия может выполнить.

Актером является тот, кто использует систему. В качестве актера может выступать человек, внешняя сущность, класс или другая система. Актеры не связаны между собой напрямую.

Вариант использования – выполняемые системой действия, приводящие к наблюдаемым актерами результатам. Каждый вариант использования описывает конкретный сценарий – последовательность шагов, которые система выполняет, чтобы получить нужный результат [3].

Студент может просматривать расписание занятий, записываться на занятия, просматривать свои записи на занятия, отменять их, просматривать список инструкторов и оставлять на них отзывы, редактировать данные своего профиля. Диаграмма вариантов использования для студента представлена на рисунке 4.

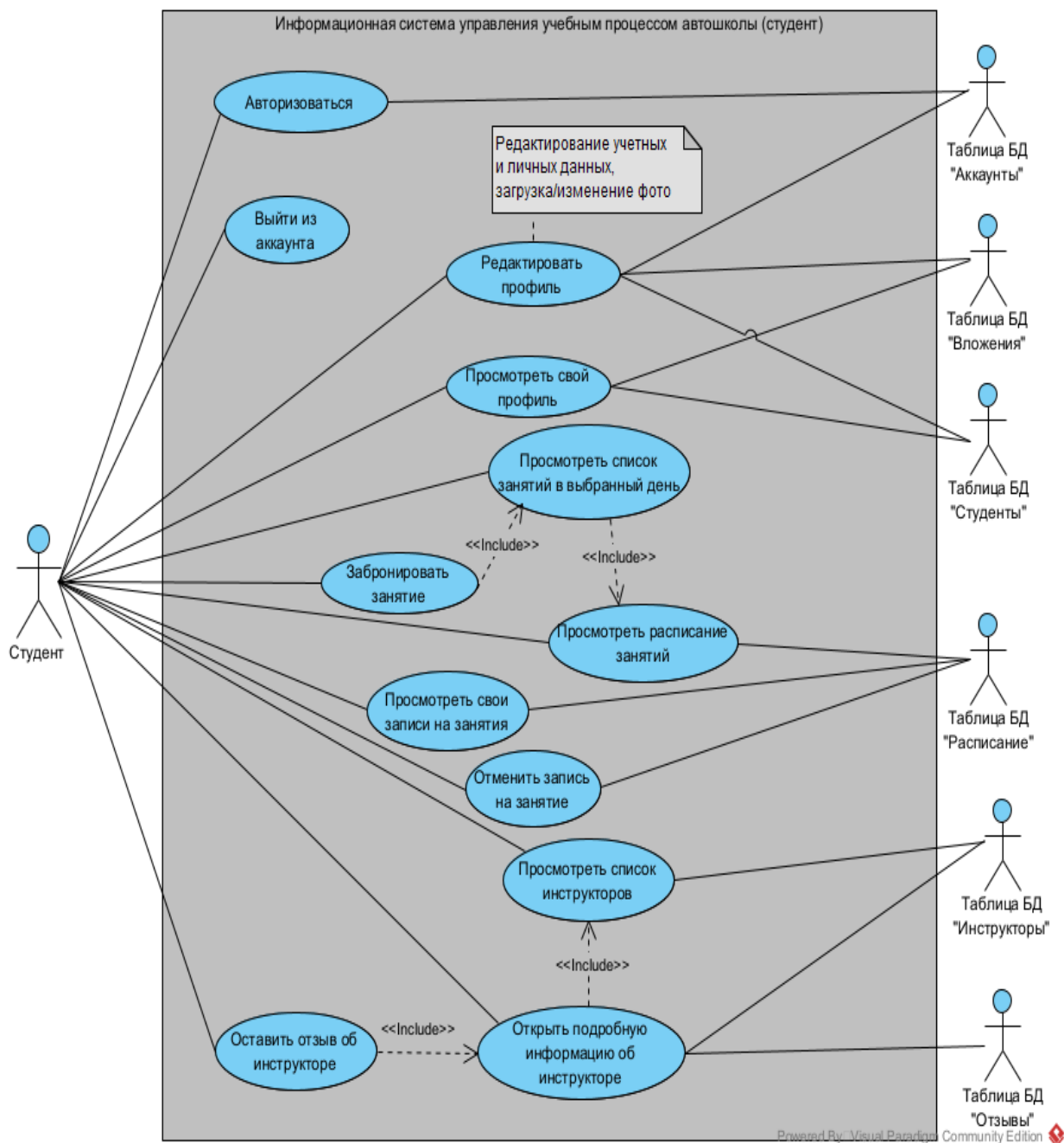


Рисунок 4 – Диаграмма вариантов использования (студент)

Инструктор может просматривать расписание занятий, добавлять занятия в расписание, просматривать свои занятия, отменять их, просматривать список студентов, просматривать подробную информацию о конкретном студенте и оставлять отзыв на него, просматривать список автошкол, просматривать и редактировать данные своего профиля. Диаграмма вариантов использования для инструктора представлена на рисунке 5.



Рисунок 5 – Диаграмма вариантов использования (инструктор)

Администратор работает с записями. Он может просматривать, добавлять, изменять и удалять данные студентов, инструкторов, аккаунтов, управлять записями автошкол, автомобилей. Так же, как и студент или инструктор, администратор может просматривать и редактировать информацию в своем профиле. Диаграмма вариантов использования для администратора представлена на рисунке 6.

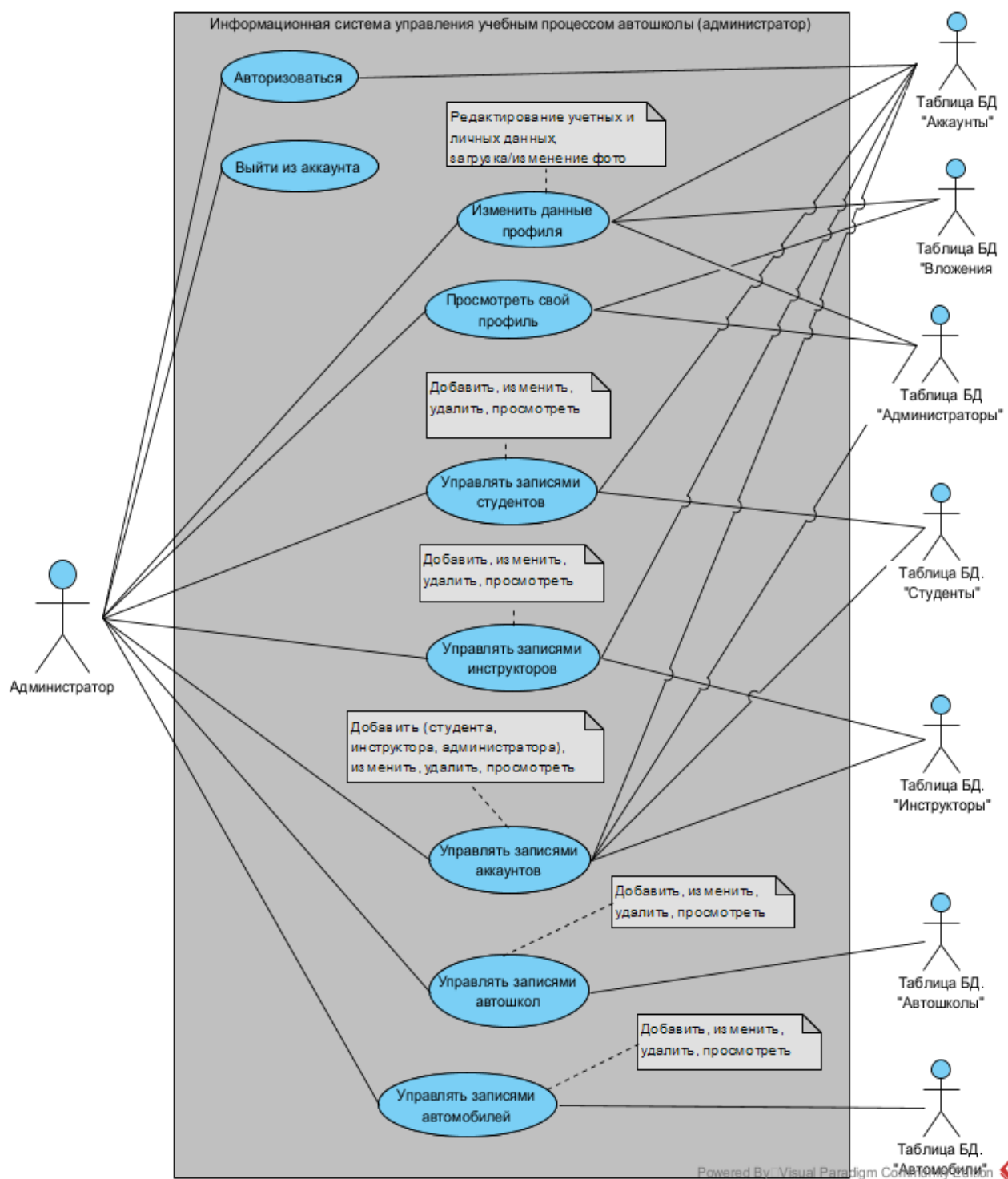


Рисунок 6 – Диаграмма вариантов использования (администратор)

Ниже приведено описание наиболее важных вариантов использования, связанных с основным функционалом записи на занятия.

Спецификация варианта использования «Авторизоваться» приведена на рисунке 7.

Прецедент: Авторизоваться
ID: 1
Краткое описание: Пользователь авторизуется в системе
Главные актеры: Пользователь
Второстепенные актеры: Нет
Предусловие: 1. Пользователь не авторизован 2. Открыта страница авторизации
Основной поток: 1. Пользователь вводит данные для входа (логин, пароль) 2. Пользователь нажимает кнопку «Войти»
Постусловия: 1. Пользователь авторизован 2. В системе отображаются доступные ему страницы

Рисунок 7 – Спецификация варианта использования «Авторизоваться»

Спецификация варианта использования «Добавить занятие в расписание» приведена на рисунке 8.

Прецедент: Добавить занятие в расписание
ID: 2
Краткое описание: Инструктор добавляет новую запись на занятие в расписание
Главные актеры: Пользователь (инструктор)
Второстепенные актеры: Нет
Предусловие: 1. Пользователь авторизован как инструктор 2. Открыта страница с расписанием
Основной поток: 1. Пользователь нажимает кнопку «Добавить» 2. Пользователь вводит дату и время 3. Пользователь нажимает кнопку «Создать»
Постусловия: 1. В расписании отображается созданное занятие

Рисунок 8 – Спецификация варианта использования «Добавить занятие в расписание»

Спецификация варианта использования «Управлять своими занятиями» приведена на рисунке 9.

Прецедент: Управлять своими занятиями
ID: 3
Краткое описание: Инструктор управляет своими записями (просматривает, отменяет)
Главные актеры: Пользователь (инструктор)
Второстепенные актеры: Нет
Предусловие: <ol style="list-style-type: none"> 1. Пользователь авторизован как инструктор 2. Открыта страница с записями инструктора
Основной поток: <ol style="list-style-type: none"> 1. Пользователь просматривает свои занятия Альтернативный поток: <ol style="list-style-type: none"> 1. Пользователь отменяет свои занятия

Рисунок 9 – Спецификация варианта использования «Управлять своими занятиями»

Спецификация варианта использования «Просмотреть список занятий в выбранный день» приведена на рисунке 10.

Прецедент: Просмотреть список занятий в выбранный день
ID: 4
Краткое описание: Студент просматривает информацию по занятиям на выбранный день
Главные актеры: Пользователь (студент)
Второстепенные актеры: Нет
Предусловие: <ol style="list-style-type: none"> 1. Пользователь авторизован как студент 2. Открыта страница с расписанием
Основной поток: <ol style="list-style-type: none"> 1. Пользователь нажимает на ячейку со свободным занятием
Постусловия: <ol style="list-style-type: none"> 1. Отображается страница с детальной информацией по занятиям на выбранный день 2. В карточке свободного занятия отображается кнопка «Забронировать»

Рисунок 10 – Спецификация варианта использования «Просмотреть список занятий в выбранный день»

Спецификация варианта использования «Забронировать занятие» приведена на рисунке 11.

Прецедент: Забронировать занятие
ID: 5
Краткое описание: Студент бронирует занятие в расписании
Главные актеры: Пользователь (студент)
Второстепенные актеры: Нет
Предусловие: <ol style="list-style-type: none"> 1. Пользователь авторизован как студент 2. Открыта страница с детальной информацией по занятиям на выбранный день 3. Имеются свободные записи
Основной поток: <ol style="list-style-type: none"> 1. Пользователь нажимает на кнопку «Забронировать» 2. В появившемся модальном окне с подтверждением бронирования занятия пользователь нажимает кнопку «Подтвердить»
Постусловия: <ol style="list-style-type: none"> 1. Пользователь записался на занятие 2. На странице с забронированными записями пользователя появляется новая запись

Рисунок 11 – Спецификация варианта использования «Забронировать занятие»

В ходе анализа функциональных требований разрабатываемой системы были выделены ключевые сценарии взаимодействия пользователей с системой, включая авторизацию управление расписанием занятий, . Для наиболее важных сценариев были приведены спецификации, которые отображают действия пользователя и системы.

1.4 Анализ нефункциональных требований

1.4.1 Общие нефункциональные требования

Нефункциональные требования определяют качество и характеристики работы системы, не связанные напрямую с конкретными функциями. Для

информационной системы управления учебным процессом автошколы можно выделить следующие нефункциональные требования: производительность, безопасность, удобство использования, модульность и расширяемость.

Производительность: система должна обеспечить быструю реакцию на действия пользователя, включая загрузку страниц, авторизацию, отображение расписания занятий и управления данными. Приложение необходимо разработать с учетом возможного роста числа пользователей. Для этого необходимо предусмотреть возможность масштабирования при увеличении нагрузки.

Безопасность: система должна обеспечить надежную авторизацию пользователей с учетом их ролей (студент, инструктор, администратор). Доступ к личным данным студентов и инструкторов должен быть ограничен в соответствии с их ролями.

Удобство использования: приложение должно обладать простым и интуитивно понятным интерфейсом, чтобы пользователь без подготовки мог легко им пользоваться. Навигация по системе должна быть интуитивно понятной, чтобы пользователи быстро находили нужную информацию и функции.

Модульность и расширяемость: архитектура системы должна позволять легко добавлять новые функции и модули без изменения уже реализованных функций. Логика работы с базой данных должна быть инкапсулирована, чтобы изменения структуры БД не влияли на остальную часть системы [4].

1.4.2 Технические средства реализации требований

Разрабатываемая информационная система управления учебным процессом в автошколе разрабатывается как веб-приложение с использованием современных технологий и архитектурных подходов.

Приложение будет построено с использованием многослойной модульной архитектуры: каждый модуль отвечает за взаимодействие с конкретной сущностью (например, студенты, инструкторы, автошколы и др.). Для каждого

модуля будут реализованы отдельные контроллеры, интерфейсы, сервисы и репозитории. Такой подход обеспечит модульность, упростит разработку, сопровождение и масштабирование системы.

Для хранения данных будет использован Microsoft SQL Server, проектирование и управление БД будет осуществляться в Microsoft SQL Server Management Studio. Это решение обеспечит высокую производительность, надежность и гибкую систему управления доступом [5].

Серверная часть будет реализована на языке программирования C# с использованием фреймворка ASP.NET Core в среде разработки Visual Studio 2022. Такой выбор позволит создать производительную, безопасную и масштабируемую серверную часть с поддержкой REST API [6].

Клиентская часть приложения будет создана с использованием HTML, CSS и JavaScript, что обеспечит совместимость с современными браузерами и гибкость интерфейса.

Для создания диаграмм, описывающих структуру и поведение системы, используется язык UML 2.0. Диаграммы разрабатываются в среде Visual Paradigm. Это удобный, гибкий инструмент для моделирования, который поддерживает графическое проектирование различных типов диаграмм и экспорт в различные форматы [7].

Веб-приложение должно быть оптимизировано для работы в популярных браузерах:

- Google Chrome (версия 84 и выше);
- Mozilla Firefox (версия 79 и выше);
- Microsoft Edge (версия 84 и выше);
- Safari (версия 16 и выше).

Преимущества выбранных инструментов и подходов:

- многослойная архитектура позволит создать простое, масштабируемое решение;

- фреймворк ASP.NET Core гарантирует высокую производительность и безопасность;
- MS SQL Server – надежная СУБД с широкими возможностями настройки;
- Visual Paradigm облегчит визуальное моделирование систем;
- клиент на HTML/CSS/JS – обеспечит совместимость с различными устройствами и браузерами.

1.5 Планирование разработки и оценка бюджета

Планирование разработки и оценка бюджета являются важными этапами при создании информационной системы управления учебным процессом автошколы. Чтобы наглядно представить сроки выполнения всех работ и рассчитать их стоимость, была построена диаграмма Ганта с использованием программы ProjectLibre.

Разработка проекта была разделена на четыре основных этапа:

- анализ предметной области и постановка задачи – включает изучение специфики работы автошколы, анализ существующих аналогов и определение требований к системе;
- проектирование информационной системы – включает разработку архитектуры, структуры базы данных, а также проектирование интерфейса и логики работы приложения;
- реализация информационной системы – охватывает создание базы данных, разработку серверной и клиентской частей системы;
- контроль качества информационной системы – включает выбор подхода к тестированию и проведение проверки работоспособности всех компонентов системы.

При планировании разработки было выделено 15 задач. За дату начала работ было взято 8 апреля 2025 года, окончание работ по проекту запланировано на 3 июня 2025 года. На рисунке 12 приведено описание задач для каждого этапа.

Название	Продолжитель...	Начало	Окончание
Начало проекта	0 дней	08.04.2025, 08:00	08.04.2025, 08:00
Анализ предметной области и постановка задачи	5 дней	08.04.2025, 08:00	14.04.2025, 17:00
Исследование предметной области	0,4 дней	08.04.2025, 08:00	08.04.2025, 11:12
Анализ информационных систем аналогичного назначения	0,6 дней	08.04.2025, 11:12	08.04.2025, 17:00
Анализ функциональных требований	2 дней	09.04.2025, 08:00	10.04.2025, 17:00
Анализ нефункциональных требований	1 день	11.04.2025, 08:00	11.04.2025, 17:00
Определение технических средств реализации требований	1 день	14.04.2025, 08:00	14.04.2025, 17:00
Проектирование информационной системы	11 дней	15.04.2025, 08:00	29.04.2025, 17:00
Определение архитектуры приложения	2,5 дней	15.04.2025, 08:00	17.04.2025, 13:00
Проектирование структур данных	2 дней	17.04.2025, 13:00	21.04.2025, 13:00
Проектирование программных средств	3 дней	21.04.2025, 13:00	24.04.2025, 13:00
Проектирование пользовательского интерфейса	3 дней	24.04.2025, 13:00	29.04.2025, 17:00
Реализация информационной системы	13 дней	30.04.2025, 08:00	22.05.2025, 17:00
Реализация базы данных	2 дней	30.04.2025, 08:00	05.05.2025, 17:00
Реализация серверной части	5 дней	06.05.2025, 08:00	14.05.2025, 17:00
Реализация клиентской части	6 дней	15.05.2025, 08:00	22.05.2025, 17:00
Контроль качества информационной системы	7 дней	23.05.2025, 08:00	02.06.2025, 17:00
Выбор режима тестирования	1 день	23.05.2025, 08:00	23.05.2025, 17:00
Тестирование серверной части	2 дней	26.05.2025, 08:00	27.05.2025, 17:00
Тестирование клиентской части	3,5 дней	28.05.2025, 08:00	02.06.2025, 17:00
Конец проекта	0 дней	03.06.2025, 10:00	03.06.2025, 10:00

Рисунок 12 – Описание этапов разработки системы

На рисунке 13 представлена диаграмма Ганта, которая показывает последовательность выполнения всех задач и продолжительность каждого этапа.

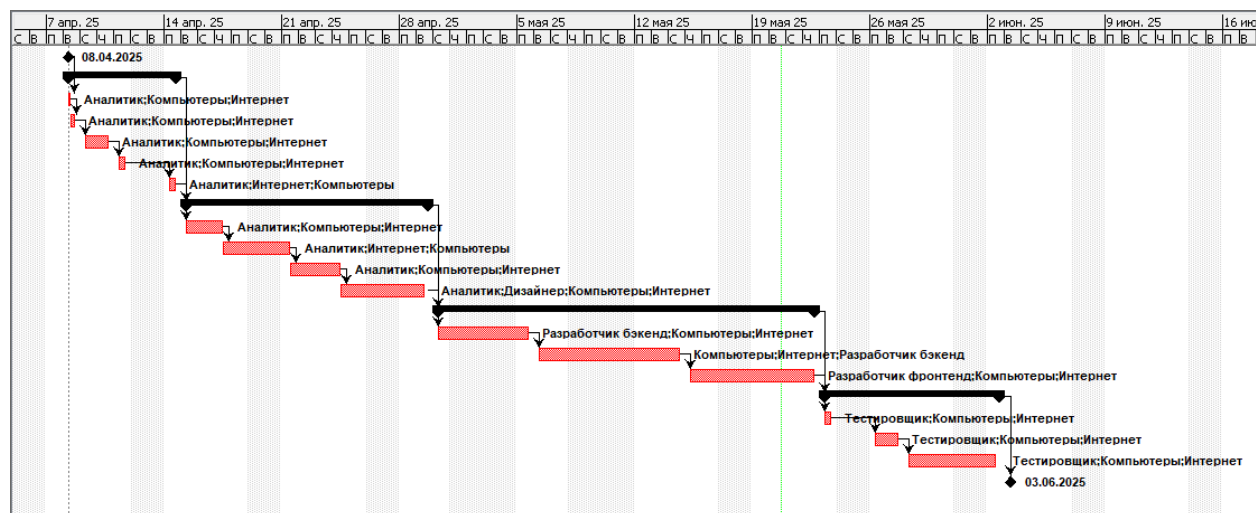


Рисунок 13 – Диаграмма Ганта с этапами разработки системы

Общая продолжительность работы над проектом составила 37 рабочих дней. Каждый этап состоит из нескольких задач, выполнение которых необходимо для перехода к следующему этапу.

На рисунке 14 приведена стоимость рабочего времени задействованных ресурсов.

		Название	Стандартная ставка
1		Аналитик	450/час
2		Дизайнер	400/час
3		Разработчик бэкенд	550/час
4		Разработчик фронтенд	500/час
5		Тестировщик	450/час
6		Компьютеры	0
7		Интернет	0

Рисунок 14 – Стоимость ресурсов

Общие затраты проекта с учетом продолжительности задач представлены на рисунке 15.

Название	Стоимость
Начало проекта	0,00 Р
☒ Анализ предметной области и постановка задачи	18000,00 Р
Исследование предметной области	1440,00 Р
Анализ информационных систем аналогичного назначения	2160,00 Р
Анализ функциональных требований	7200,00 Р
Анализ нефункциональных требований	3600,00 Р
Определение технических средств реализации требований	3600,00 Р
☒ Проектирование информационной системы	45600,00 Р
Определение архитектуры приложения	7200,00 Р
Проектирование структур данных	7200,00 Р
Проектирование программных средств	10800,00 Р
Проектирование пользовательского интерфейса	20400,00 Р
☒ Реализация информационной системы	63600,00 Р
Реализация базы данных	17600,00 Р
Реализация серверной части	22000,00 Р
Реализация клиентской части	24000,00 Р
☒ Контроль качества информационной системы	23400,00 Р
Выбор режима тестирования	3600,00 Р
Тестирование серверной части	7200,00 Р
Тестирование клиентской части	12600,00 Р
Конец проекта	0,00 Р
Итого: 150600 руб.	

Рисунок 15 – Общие затраты проекта

Итоговая стоимость проекта составила 150 600 рублей. Распределение затрат по этапам выглядит следующим образом:

- анализ предметной области и постановка задачи – 18 000 руб;
- проектирование информационной системы – 45 600 руб;
- реализация информационной системы – 63 600 руб;
- контроль качества информационной системы – 23 400 руб.

Проведенное планирование позволит заранее определить все основные этапы работы, правильно распределить ресурсы и обеспечит выполнение проекта в запланированные сроки.

2 Проектирование информационной системы управления учебным процессом автошколы

2.1 Архитектура приложения

Основной целью проектирования является создание гибкой, расширяемой и поддерживаемой информационной системы, способной эффективно управлять процессами автошколы.

Система будет построена по принципу клиент-серверной архитектуры, где часть клиента представляет собой интерфейс пользователя, а серверная часть отвечает за обработку данных и бизнес-логику. Такой подход позволяет эффективно распределять нагрузку, упрощает масштабирование системы, повышает ее надежность [8].

Для проектирования системы выбрана многослойная архитектура, что обеспечивает четкое разделение ответственности между различными слоями и компонентами системы. Основные слои системы: слой представления, бизнес-логика, слой доступа к данным, база данных.

Слой представления: отвечает за взаимодействие с пользователем и отображение информации.

Слой бизнес-логики: реализует бизнес-правила и логику обработки данных. Этот слой обрабатывает запросы от слоя представления, выполняет необходимые вычисления и манипуляции с данными.

Слой доступа к данным: отвечает за взаимодействие с базой данных. Этот слой обеспечивает доступ к данным, их извлечение и манипуляции.

База данных: отвечает за хранение данных.

Клиентская часть будет разработана с использованием языков разметки HTML, CSS и языка программирования Java Script в среде разработки Visual Studio 2022. Интерфейс пользователя должен быть интуитивно понятным, обеспечивая доступ ко всем функциям системы через удобные элементы управления.

Серверная часть будет реализована на языке C# с использованием фреймворка ASP.NET Core, что обеспечит высокую производительность и возможность масштабирования системы. Для хранения данных будет использоваться база данных Microsoft SQL Server.

Основные компоненты системы:

- модуль авторизации, который отвечает за вход пользователей в систему с разграничением ролей (студенты, инструкторы и администраторы);
- модуль управления расписанием будет обеспечивать создание, редактирование и просмотр расписания занятий как для студентов, так и для инструкторов;
- модуль управления пользователями включает функциональность для управления учетными записями студентов, инструкторов и администраторов;
- модуль взаимодействия с базой данных в виде репозитория реализует методы для доступа к данным, хранящимся в базе данных, и поддерживает выполнение операций создания, чтения и изменения данных.

На рисунке 16 представлена диаграмма развертывания системы, отображающая архитектуру ИС управления учебным процессом автошколы.

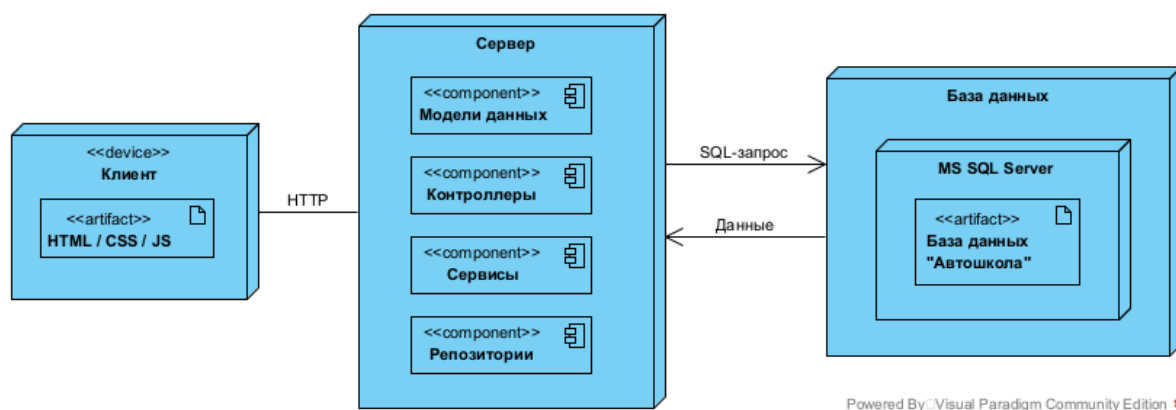


Рисунок 16 – Диаграмма развертывания ИС управления учебным процессом автошколы

Для визуализации взаимодействия компонентов была создана диаграмма компонентов. Диаграмма компонентов отображена на рисунке 17.

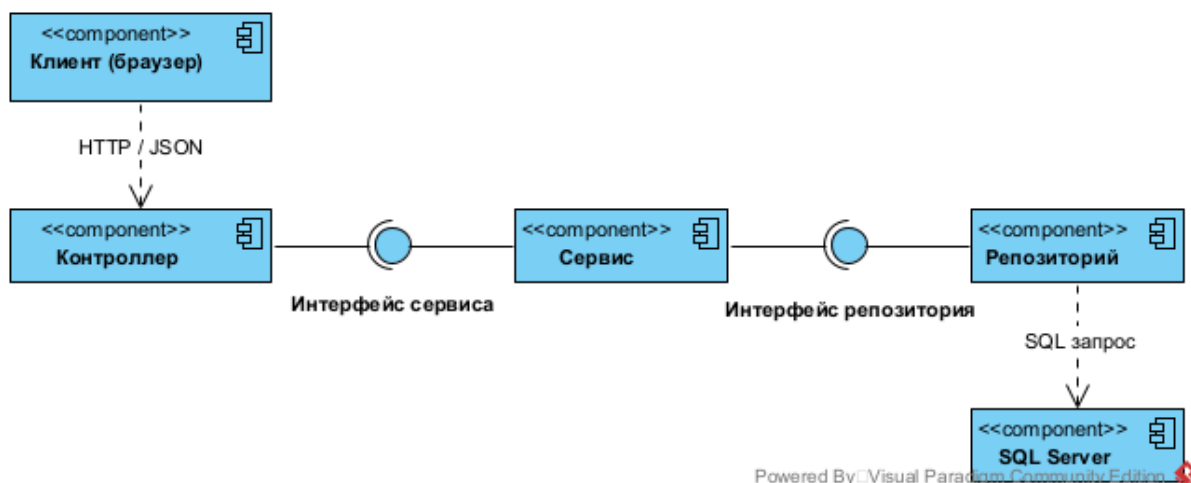


Рисунок 17 – Диаграмма компонентов серверной части системы

Таким образом было выполнено высокоуровневое проектирование информационной системы управления учебным процессом автошколы.

2.2 Проектирование структур данных

2.2.1 Концептуальная модель базы данных

Концептуальная модель – это упрощенное описание предметной области, которое показывает, какие объекты существуют, какими свойствами они обладают и как связаны между собой. Она не зависит от конкретных технологий или способов хранения данных и служит основой для дальнейшего проектирования информационной системы [9].

Для составления концептуальной модели данных требуется выделить список всех основных сущностей предметной области разрабатываемой системы. Описание сущностей предметной области представлено в таблице 2.

Таблица 2 – Сущности БД

Название сущности	Идентификатор сущности	Назначение сущности
Студент	Student	Содержит сведения о студентах
Инструктор	Instructor	Содержит сведения об инструкторах
Администратор	Admin	Содержит сведения об администраторах

Продолжение таблицы 2

Название сущности	Идентификатор сущности	Назначение сущности
Аккаунт	Account	Содержит информацию об учетных данных
Расписание	Schedule	Содержит сведения о расписании занятий
Автошкола	Driving_School	Содержит сведения об автошколах
Город	City	Содержит сведения о городах
Автомобиль	Car	Содержит информацию об автомобилях инструкторов
Отзыв	Review	Содержит сведения об отзывах студентов и инструкторов
Вложение	Attachment	Содержит ссылки на файлы

Графическое представление концептуальной модели базы данных представлено на рисунке 18.

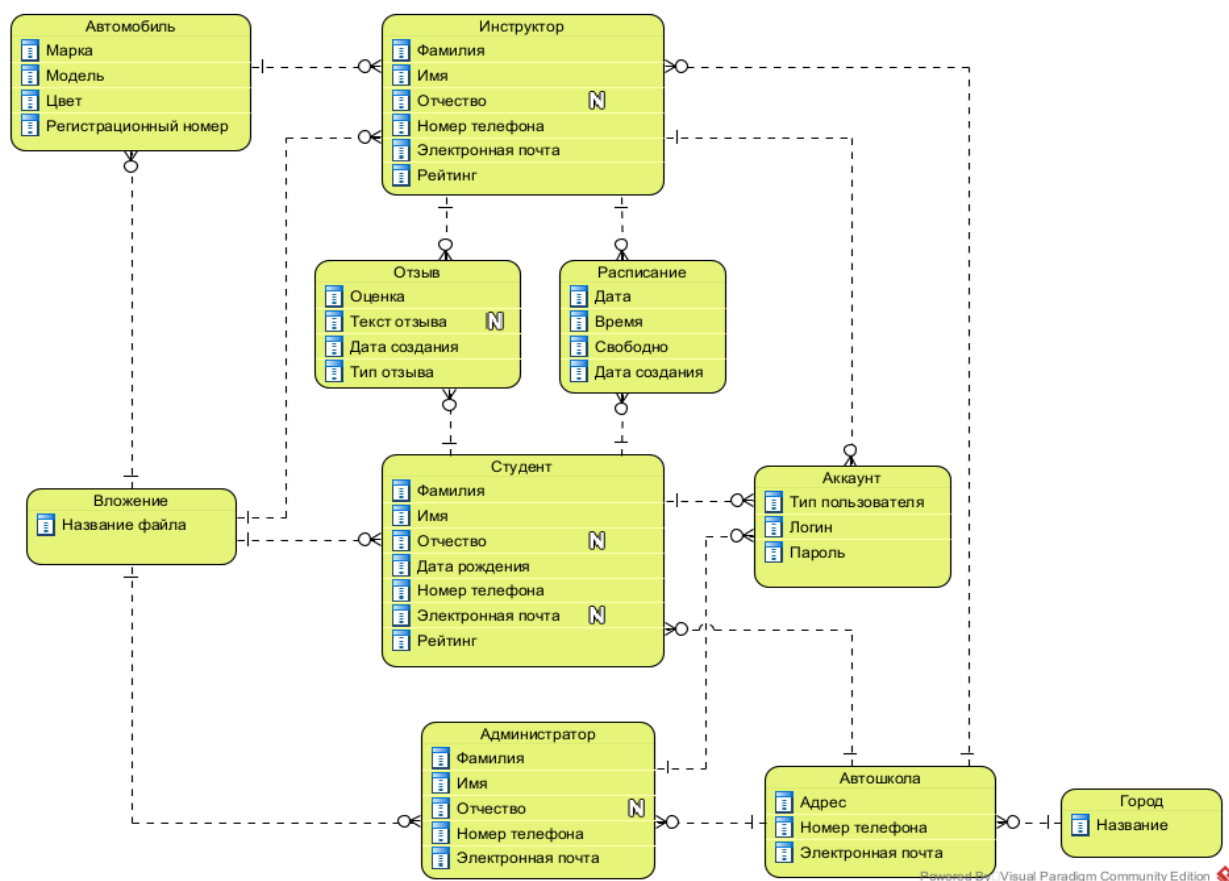


Рисунок 18 – Концептуальная модель базы данных ИС управления учебным процессом автошколы

Таким образом была разработана концептуальная модель разрабатываемой базы данных ИС управления учебным процессом автошколы.

2.2.2 Логическая модель базы данных

Логическая модель данных представляет собой более детальную и технически проработанную версию концептуальной модели. В ней осуществляется детальная проработка сущностей, их атрибутов и взаимосвязей, что создает основу для последующего перехода к физической реализации [10].

Логическая модель проектируемой базы данных представлена на рисунке 19.

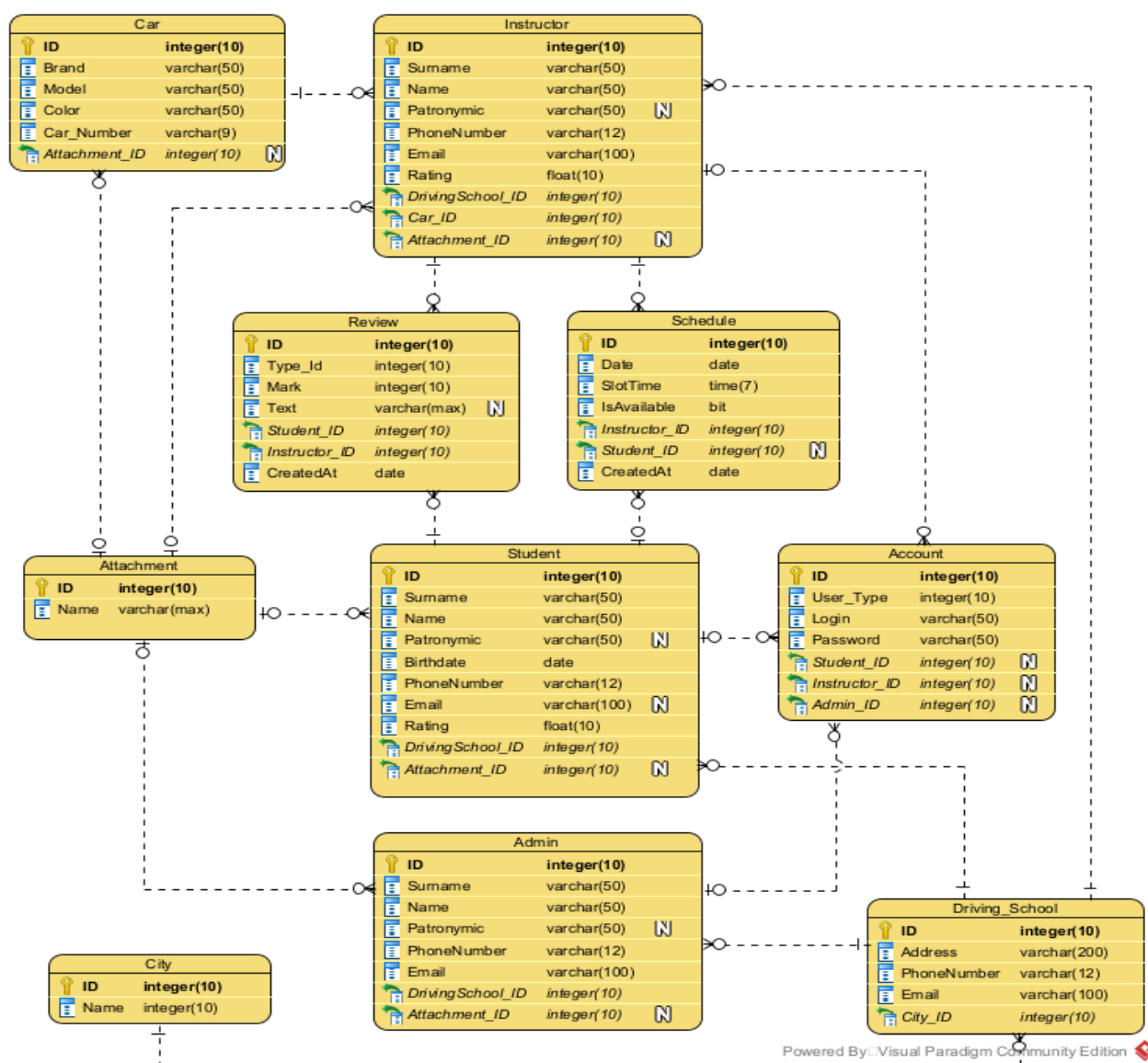


Рисунок 19 – Логическая модель базы данных системы

Таким образом, разработанные модели полностью отражают структуру предметной области и обеспечат основу для последующего перехода к физической реализации.

Наборы атрибутов для каждой сущности предметной области и их характеристики приведены в приложении Д.

2.3 Проектирование программных средств

На основании разработанной архитектуры для каждого из модулей были построены диаграммы классов. Диаграммы классов основных модулей представлены в приложении Е.

Серверная часть системы реализуется с использованием фреймворка ASP.NETCore, который обеспечит модульную, кроссплатформенную и производительную среду для разработки веб-приложений. Серверное приложение будет следовать многослойной архитектуре, разделяя логику на несколько уровней:

- слой контроллеров (Controllers): обрабатывает HTTP-запросы, вызывает соответствующие методы бизнес-логики и возвращает результаты на клиентскую часть;
- слой бизнес-логики (Services): содержит основную логику информационной системы, включая обработку данных и выполнение бизнес-процессов;
- слой интерфейсов (Interfaces): содержит интерфейсы для сервисов и репозиториев, что позволит обеспечить абстракцию, гибкость и облегчить тестирование и расширяемость системы;
- слой доступа к данным (Repositories): взаимодействует с базой данных через Entity Framework Core;
- слой моделей (Models): определяет структуры данных, используемых в системе [11].

Краткая спецификация контроллеров представлена в таблице 3.

Таблица 3 – Контроллеры

Название класса	Назначение
AuthController	Обрабатывает запросы на авторизацию пользователей
CarController	Управляет данными об автомобилях, прикрепленных к инструкторам
Driving_SchoolController	Управляет данными об автошколах, в которых работают инструкторы и учатся студенты
InstructorController	Управляет личными и профессиональными данными инструкторов
ScheduleController	Обрабатывает запросы, связанные с расписанием занятий
StudentController	Управляет личными и учебными данными студентов
AdminController	Управляет данными администраторов системы
ReviewController	Обрабатывает отзывы студентов и инструкторов о занятиях и взаимодействии
AttachmentController	Управляет вложениями (фото и документы студентов, инструкторов, автомобилей)
CityController	Обрабатывает запросы, связанные с городами, в которых действуют автошколы

Каждый контроллер взаимодействует с соответствующим сервисом, который изолирует бизнес-логику. Сервисы обрабатывают данные, выполняют проверки и обеспечивают выполнение бизнес-процессов приложения. Спецификация сервисов приведена в таблице 4.

Таблица 4 – Сервисы

Название класса	Назначение
AuthService	Реализует логику авторизации пользователей
CarService	Реализует бизнес-логику управления данными автомобилей
Driving_SchoolService	Управляет логикой обработки данных об автошколах
InstructorService	Обрабатывает логику управления инструкторами

Продолжение таблицы 4

Название класса	Назначение
ScheduleService	Управляет логикой создания, редактирования и удаления расписаний
StudentService	Реализует бизнес-логику по управлению данными студентов
AdminService	Управляет логикой обработки данных администраторов
ReviewService	Обрабатывает отзывы, включая создание, модерацию и отображение
AttachmentService	Управляет логикой работы с вложениями
CityService	Управляет справочником городов

Каждый сервис имеет свой интерфейс, который определяет методы для взаимодействия с другими компонентами системы.

Описание интерфейсов сервисов представлено в таблице 5.

Таблица 5 – Интерфейсы сервисов

Название интерфейса	Назначение
IAuthService	Интерфейс для авторизации пользователей
ICarService	Определяет методы работы с автомобилями
IDriving_SchoolService	Интерфейс управления данными об автошколах
IInstructorService	Интерфейс взаимодействия с данными инструкторов
IScheduleService	Интерфейс работы с расписанием занятий
IStudentService	Определяет методы управления данными студентов
IAdminService	Интерфейс обработки данных администраторов
IReviewService	Интерфейс для работы с отзывами
IAttachmentService	Интерфейс взаимодействия с вложениями
ICityService	Интерфейс управления справочником городов

Репозитории обеспечивают доступ к данным, хранящимся в базе данных. Каждый репозиторий реализует методы для выполнения операций чтения, записи, обновления и удаления.

Перечень используемых репозиторий приведен в таблице 6.

Таблица 6 – Репозитории

Название класса	Назначение
AuthRepository	Управляет данными, связанными с авторизацией
CarRepository	Обеспечивает доступ к данным автомобилей
Driving_SchoolRepository	Выполняет операции с данными автошкол
InstructorRepository	Осуществляет доступ к данным инструкторов
ScheduleRepository	Выполняет операции с расписанием занятий
StudentRepository	Обрабатывает запросы к данным студентов
AdminRepository	Управляет данными администраторов
ReviewRepository	Хранит и извлекает данные отзывов
AttachmentRepository	Работает с вложениями (фото студентов, инструкторов, автомобилей)
CityRepository	Обрабатывает данные о городах

Для каждого репозитория также создан интерфейс, который определяет набор операций, доступных для работы с данными.

Интерфейсы репозиторий представлены в таблице 7.

Таблица 7 – Интерфейсы репозиторий

Название интерфейса	Назначение
IAuthRepository	Интерфейс доступа к данным авторизации
ICarRepository	Интерфейс работы с данными автомобилей
IDriving_SchoolRepository	Интерфейс управления данными автошкол
IInstructorRepository	Интерфейс доступа к данным инструкторов
IScheduleRepository	Интерфейс операций с расписанием
IStudentRepository	Интерфейс доступа к данным студентов
IAdminRepository	Интерфейс доступа к данным администраторов
IReviewRepository	Интерфейс управления отзывами
IAttachmentRepository	Интерфейс доступа к вложениям
ICityRepository	Интерфейс управления данными о городах

Диаграммы классов, сопровождающие данное описание, наглядно иллюстрируют взаимодействие между слоями и демонстрируют структурированную архитектуру системы.

2.4 Проектирование пользовательского интерфейса

Пользовательский интерфейс разрабатываемой ИС управления учебным процессом автошколы представляет собой веб-приложение, работающее в браузере. Интерфейс будет создан с использованием HTML, CSS и JavaScript, что обеспечит его кроссбраузерность и доступность с различных устройств [12].

Целью проектирования интерфейса является создание простого, понятного и удобного средства взаимодействия с системой для всех категорий пользователей: студентов, инструкторов и администраторов. Интерфейс должен быть интуитивно понятным, чтобы пользователь без специальной подготовки мог быстро освоиться и начать работу с системой.

В системе предусмотрены следующие основные страницы и элементы интерфейса:

- a) главная публичная страница – содержит основную информацию о системе, доступную анонимному пользователю;
- b) страница авторизации – предназначена для входа в систему с учетом роли пользователя (студент, инструктор, администратор);
- c) профиль пользователя – отображает личные данные и предоставляет возможность их редактирования;
- d) расписание занятий:
 - 1) для студентов – отображение доступных занятий, возможность записаться или отменить запись;
 - 2) для инструкторов – просмотр своих занятий, возможность добавления и отмены занятий;
- e) список пользователей:
 - 1) для студентов – отображение списка инструкторов с возможностью оставить отзыв;

- 2) для инструкторов – просмотр списка студентов;
- 3) для администратора – доступ ко всем пользователям системы с возможностью управления записями (добавление, редактирование, удаление);
- f) управление автошколами и автомобилями – доступно только администратору для ведения справочной информации.

Каждая страница интерфейса должна взаимодействовать с серверной частью системы с помощью REST API. Запросы будут отправляться с использованием HTTP-протокола через встроенный в браузер API fetch. Все запросы обрабатываются на сервере, а ответы возвращаются в формате JSON и отображаются пользователю.

В интерфейсе должна быть реализована обработка ошибок:

- при вводе некорректных данных пользователю отображаются соответствующие уведомления;
- предусмотрены сообщения об успехе при выполнении операций (например, при успешной записи на занятие или редактировании профиля).

Навигация и основные элементы управления должны быть размещены логично и последовательно, чтобы упростить взаимодействие.

3 Реализация информационной системы управления учебным процессом автошколы

3.1 Реализация базы данных

На основе концептуальной и логической модели, была создана физическая модель данных, представленная на рисунке 20. Физическая модель данных – это точное представление структуры базы данных, предназначенное для конкретной СУБД. Она определяет, как логическая модель будет реализована на физическом уровне [13].

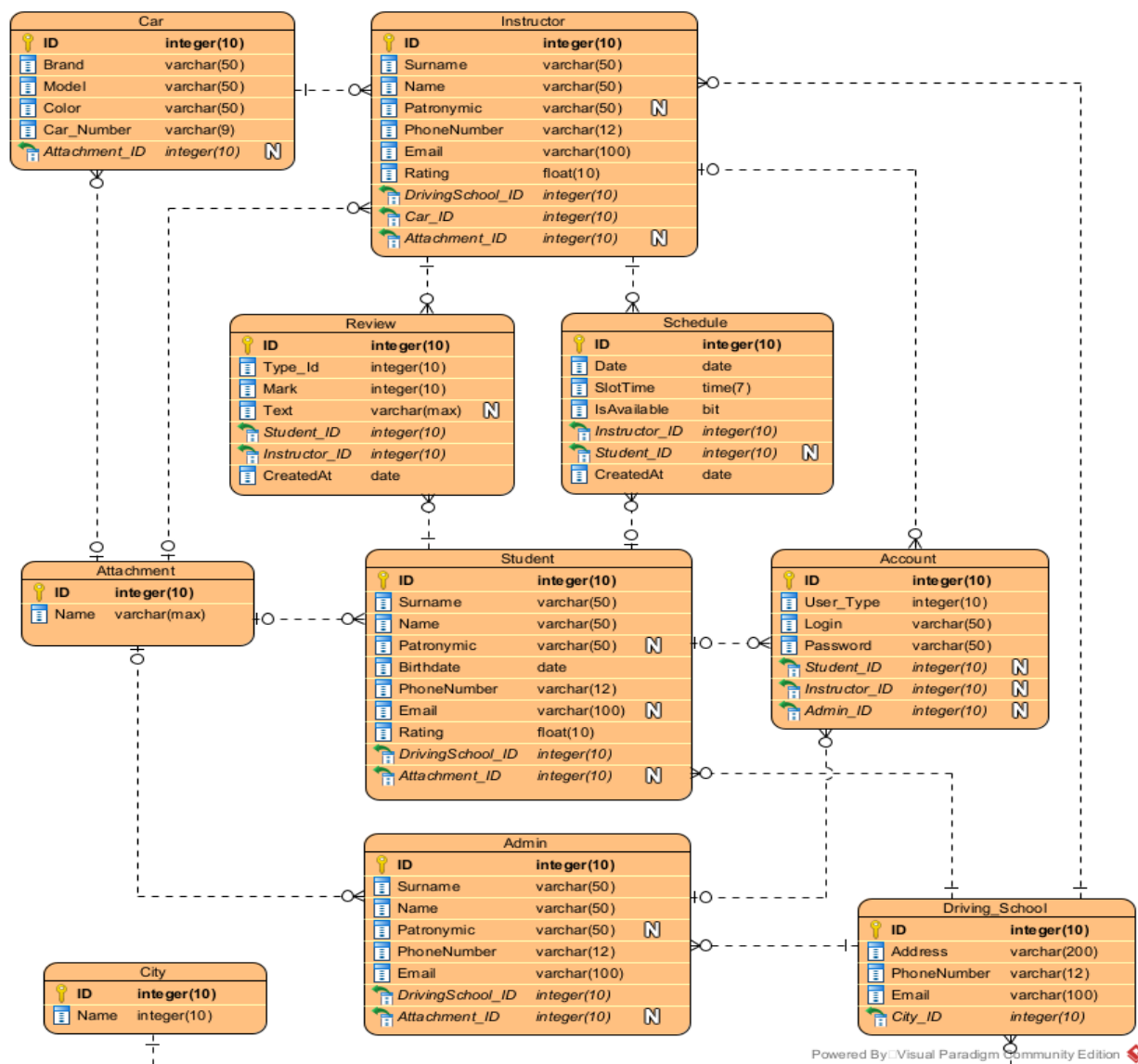


Рисунок 20 – Физическая модель базы данных ИС управления учебным процессом автошколы

Согласно физической модели с помощью СУБД MS SQL Server была развернута база данных ИС управления учебным процессом автошколы. Диаграмма развернутой базы данных представлена на рисунке 21.

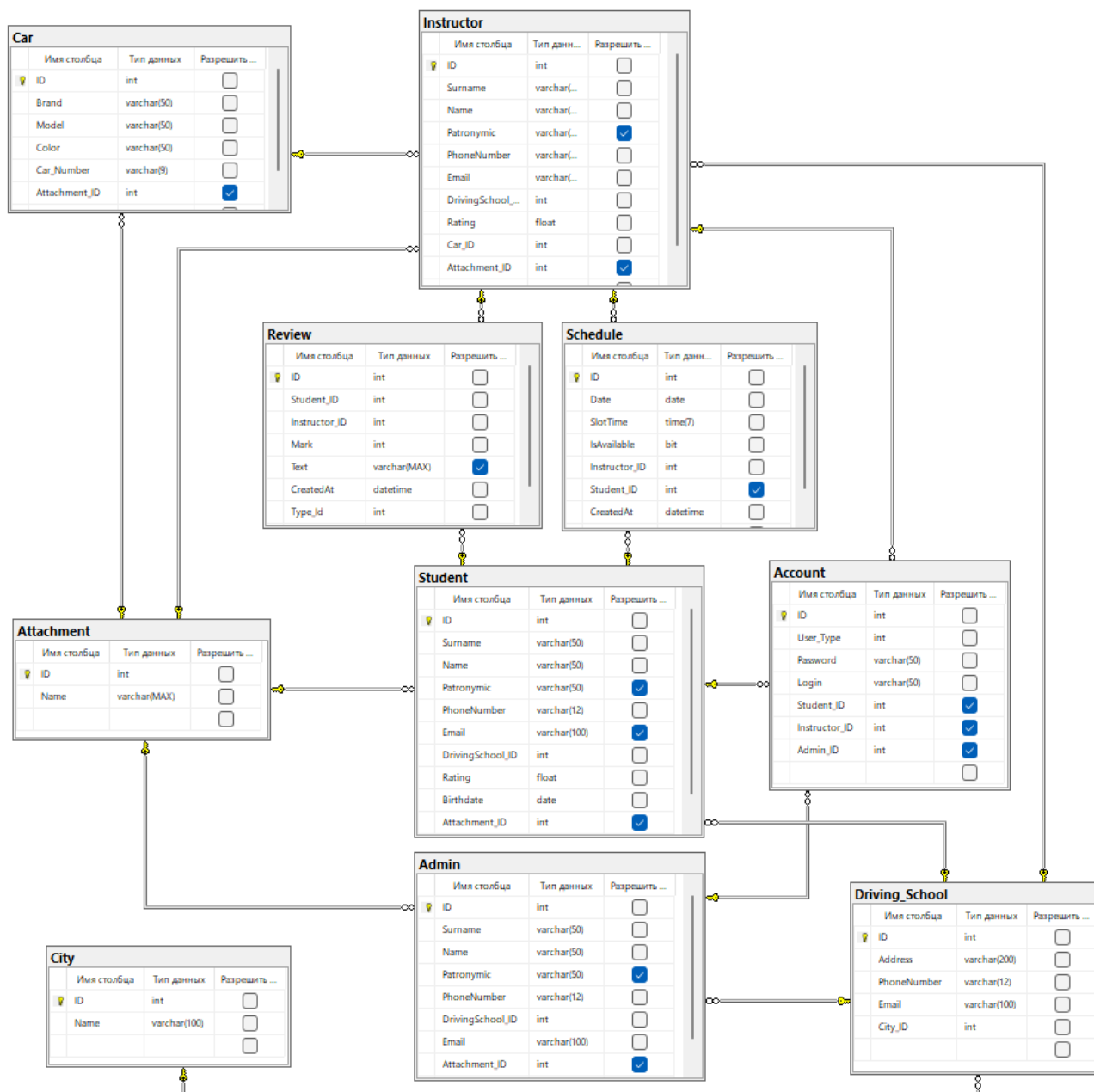


Рисунок 21 – Диаграмма базы данных ИС управления учебным процессом автошколы

Таким образом была реализована база данных системы управления учебным процессом автошколы, которая обеспечивает хранение, обработку и управление данными, необходимыми для функционирования информационной системы.

3.2 Реализация серверной части информационной системы управления учебным процессом автошколы

Серверная часть информационной системы управления учебным процессом автошколы разрабатывалась в среде разработки Visual Studio 2022, на языке программирования C# с использованием фреймворка ASP.NET Core. Фрагменты кода серверной части системы приведены в приложении Б. С полным кодом можно ознакомиться на GitHub [14].

Серверная часть построена по принципу многослойной архитектуры и включает в себя десять модулей, каждый из которых отвечает за отдельную функциональность системы. Для каждого модуля разработаны:

- модели, соответствующие структуре таблиц базы данных;
- интерфейсы сервисов, определяющие контракт взаимодействия;
- сервисы, содержащие бизнес-логику;
- репозитории, обеспечивающие доступ к данным;
- контроллеры, обрабатывающие HTTP-запросы.

Такой подход обеспечивает разделение ответственности и упрощает сопровождение и тестирование приложения [15].

Серверная часть взаимодействует с клиентской через REST API, реализованный в каждом модуле. Методы API поддерживают стандартные HTTP-операции: GET, POST, PUT, DELETE и обеспечивают полный цикл работы с сущностями системы.

Реализованные методы для модуля работы с администраторами описаны в таблице 8.

Таблица 8 – Методы API модуля «Администраторы»

URL запроса	HTTP метод	Описание
/admins	GET	Получить список всех администраторов
/admins	POST	Создать нового администратора
/admins/{id}	GET	Получить данные администратора по ID
/admins/{id}	PUT	Обновить данные администратора по ID
/admins/{id}	DELETE	Удалить администратора по ID

Реализованные методы для модуля работы со вложениями приведены в таблице 9.

Таблица 9 – Методы API модуля «Вложения»

URL запроса	HTTP метод	Описание
/attachment/{id}	GET	Получить вложение по ID
/attachment	POST	Добавить вложение
/attachment/{id}	DELETE	Удалить вложение по ID

Разработанные методы для модуля управления аккаунтами представлены в таблице 10.

Таблица 10 – Методы API модуля «Аккаунты»

URL запроса	HTTP метод	Описание
/accounts	GET	Получить список всех аккаунтов
/accounts	POST	Создать новый аккаунт
/accounts/{id}	GET	Получить аккаунт по ID
/accounts/{id}	DELETE	Удалить аккаунт по ID
/account/{id}	PUT	Обновить аккаунт по ID
/login	POST	Авторизация
/register/instructor	POST	Регистрация инструктора
/register/student	POST	Регистрация студента
/register/admin	POST	Регистрация администратора

Реализованные методы для модуля работы с автомобилями приведены в таблице 11.

Таблица 11 – Методы API модуля «Автомобили»

URL запроса	HTTP метод	Описание
/cars	GET	Получить список автомобилей
/cars	POST	Добавить новый автомобиль
/cars/{id}	GET	Получить автомобиль по ID
/cars/{id}	PUT	Обновить автомобиль по ID
/cars/{id}	DELETE	Удалить автомобиль по ID

Методы для модуля работы с городами отображены в таблице 12.

Таблица 12 – Методы API модуля «Города»

URL запроса	HTTP метод	Описание
/city	GET	Получить список городов
/city	POST	Добавить новый город

Продолжение таблицы 12

URL запроса	HTTP метод	Описание
/city/{id}	GET	Получить город по ID
/city/{id}	PUT	Обновить город по ID
/city/{id}	DELETE	Удалить город по ID

Методы для модуля управления автошколами приведены в таблице 13.

Таблица 13 – Методы API модуля «Автошколы»

URL запроса	HTTP метод	Описание
/driving_Schools	GET	Получить список автошкол
/driving_Schools/{id}	GET	Получить автошколу по ID
/driving_Schools	POST	Получить отфильтрованный список автошкол
/driving_Schools/{id}	PUT	Обновить автошколу по ID
/driving_Schools/{id}	DELETE	Удалить автошколу по ID
/driving_School	POST	Добавить автошколу

Реализованные методы для модуля управления инструкторами приведены в таблице 14.

Таблица 14 – Методы API модуля «Инструкторы»

URL запроса	HTTP метод	Описание
/instructors	GET	Получить список инструкторов
/instructors	POST	Получить отфильтрованный список инструкторов
/instructors/{id}	GET	Получить инструктора по ID
/instructors/{id}	PUT	Обновить инструктора по ID
/instructors/{id}	DELETE	Удалить инструктора по ID
/instructor	POST	Добавить инструктора

Реализованные методы для модуля работы с отзывами приведены в таблице 15.

Таблица 15 – Методы API модуля «Отзывы»

URL запроса	HTTP метод	Описание
/reviews	GET	Получить все отзывы
/reviews/{id}	GET	Получить отзыв по ID
/reviews/{id}	DELETE	Удалить отзыв по ID
/reviews/to-instructor/{instructorId}	GET	Получить отзывы, адресованные инструктору

Продолжение таблицы 15

URL запроса	HTTP метод	Описание
/reviews/to-student/{studentId}	GET	Получить отзывы, адресованные студенту
/reviews/from-student/{studentId}	GET	Получить отзывы, написанные студентом
/reviews/from-instructor/{instructorId}	GET	Получить отзывы, написанные инструктором
/review	POST	Добавить новый отзыв

Реализованные методы для модуля управления расписанием представлены в таблице 16.

Таблица 16 – Методы API модуля «Расписание»

URL запроса	HTTP метод	Описание
/schedules	GET	Получить все расписания
/schedule/{id}	GET	Получить расписание по ID
/schedules/instructor/{instructor_id}	GET	Получить расписание конкретного инструктора
/schedules/student/{student_id}	GET	Расписание конкретного студента
/schedule	POST	Создать расписание
/schedules/{id}	PUT	Обновить расписание по ID
/schedules/{id}	DELETE	Удалить расписание по ID

Разработанные методы для модуля работы со студентами приведены в таблице 17.

Таблица 17 – Методы API модуля «Студенты»

URL запроса	HTTP метод	Описание
/students	GET	Получить список студентов
/students	POST	Получить отфильтрованный список студентов
/students/{id}	GET	Получить студента по ID
/students/{id}	PUT	Обновить данные студента по ID
/students/{id}	DELETE	Удалить студента по ID
/student	POST	Добавить студента

Несмотря на то, что система реализована в виде монолита, она обладает высокой степенью модульности, что в дальнейшем позволит при необходимости

перейти к микросервисной архитектуре без значительных архитектурных изменений.

3.3 Реализация клиентской части информационной системы управления учебным процессом автошколы

Клиентская часть информационной системы управления учебным процессом автошколы отвечает за отображение пользовательского интерфейса, обработку действий пользователей и взаимодействие с серверной частью системы посредством REST API. Ее структура была спроектирована с учетом модульности и разделения функционала на отдельные компоненты.

Для клиентской части системы была разработана диаграмма компонентов, которая поможет представить общую картину архитектуры клиентской части [16]. На диаграмме отображены HTML-страницы, подключаемые скрипты, таблицы стилей и зависимости между ними. Диаграмма компонентов представлена на рисунке Е.5 в приложении Е.

В таблице 18 приведено описание всех компонентов клиентской части системы.

Таблица 18 – Описание компонентов клиентской части информационной системы управления учебным процессом автошколы

Компонент	Назначение
main.html	Главная публичная страница системы
main.css	Стилизация главной страницы
auth.html	Страница авторизации пользователя
auth.js	Логика авторизации и валидации формы
auth.css	Стили для страницы авторизации
profile.html	Страница профиля текущего пользователя
profile.js	Загрузка и отображение данных профиля
profile.css	Визуальное оформление профиля
accounts.html	Страница управления аккаунтами
accounts.js	Обработка данных аккаунтов
accounts.css	Визуальное оформление таблицы аккаунтов

Продолжение таблицы 18

Компонент	Назначение
instructors.html	Страница со списком инструкторов
instructors.js	Загрузка списка инструкторов
instructors.css	Стилизация таблицы инструкторов
instructor-details.html	Просмотр информации об инструкторе
instructor-details.js	Загрузка и отображение подробной информации
instructor-details.css	Оформление карточки инструктора
schedule.html	Страница с расписанием
schedule.js	Загрузка и отображение расписания
schedule.css	Визуальная стилизация расписания
schedule-date.html	Расписание занятий на конкретную дату
schedule-date.js	Запрос и отображение данных по дате
schedule-date.css	Стилизация отображения по дням
students.html	Список студентов
students.js	Получение и отображение данных студентов
students.css	Стилизация таблицы студентов
student-details.html	Подробная информация о студенте
student-details.js	Загрузка и отображение карточки студента
student-details.css	Визуальное оформление блока информации
student-schedules.html	Просмотр расписания студента
student-schedules.js	Страница с занятиями авторизованного студента
student-schedules.css	Визуальное оформление карточек занятий
instructor-schedules.html	Страница с занятиями инструктора
instructor-schedules.js	Загрузка расписания инструктора
instructor-schedules.css	Оформление карточек расписания
driving-schools.html	Страница со списком автошкол
driving-schools.js	Получение и отображение списка автошкол
driving-schools.css	Визуальное оформление элементов списка
cars.html	Страница со списком автомобилей
cars.js	Загрузка информации об автомобилях
cars.css	Стили для таблицы автомобилей
roles.js	Общий скрипт проверки ролей и доступа
styles.css	Общие стили для всех страниц

Клиентская часть информационной системы управления учебным процессом автошколы реализуется с использованием языков разметки HTML, CSS, языка программирования JavaScript в среде разработки Visual Studio 2022.

Взаимодействие с серверной частью осуществляется путем отправки HTTP-запросов к API. Полученные от сервера данные отображаются в интерфейсе с использованием JavaScript. Фрагменты кода клиентской части системы приведены в приложении В. С полным кодом можно ознакомиться на Github [14]

3.4 Реализация интерфейса информационной системы управления учебным процессом автошколы

Согласно требованиям к пользовательскому интерфейсу, были разработаны следующие страницы:

- главная публичная страница;
- страница авторизации;
- страница профиля;
- страница с расписанием;
- страница с информацией о занятиях на конкретный день;
- страница со списком студентов;
- страница с информацией о конкретном студенте;
- страница со списком инструкторов;
- страница с информацией о конкретном инструкторе;
- страница со списком автошкол;
- страница со списком автомобилей;
- страница со списком аккаунтов.

При открытии системы пользователь попадает на главную страницу, которая доступна анонимным пользователям (рисунок 22). Пользователь может ознакомиться с общей информацией о системе и доступных возможностях без необходимости входа в учетную запись.

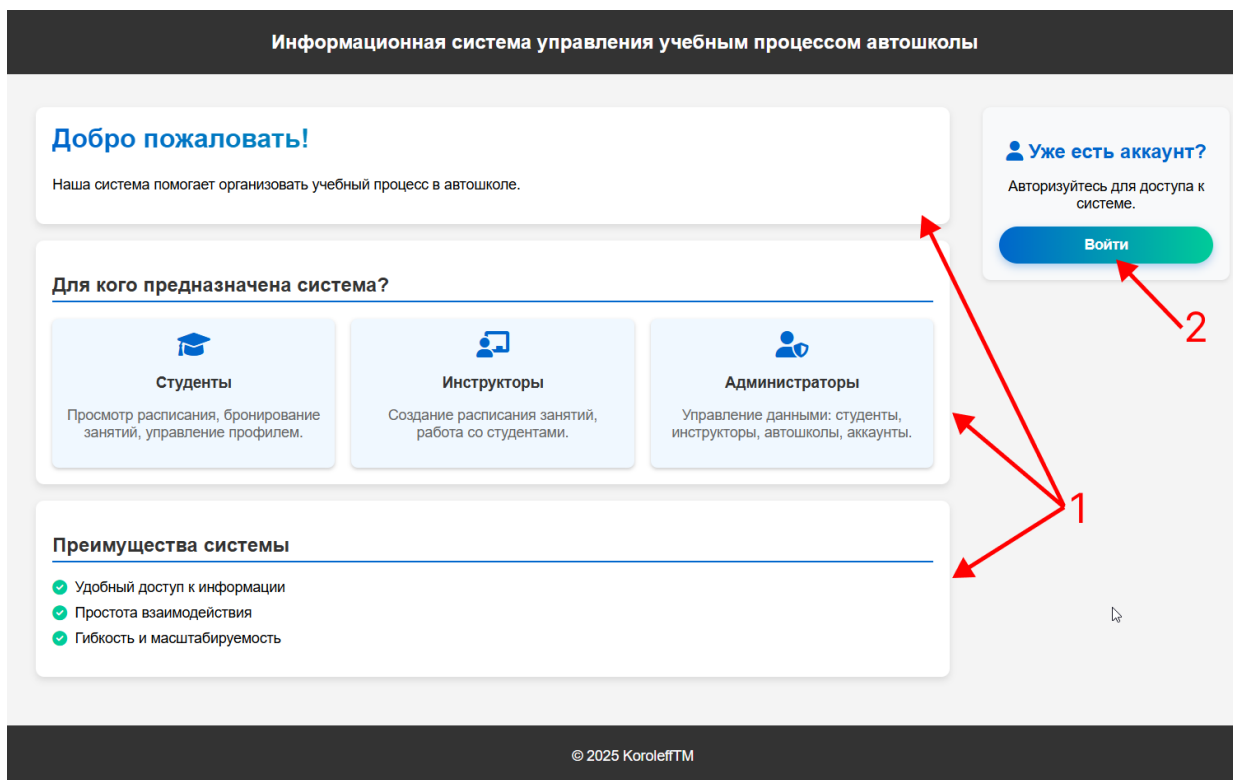


Рисунок 22 – Главная публичная страница

Основные элементы главной страницы:

1 – информация о системе;

2 – кнопка «Войти».

При нажатии на кнопку «Войти» откроется страница авторизации (рисунок 23).

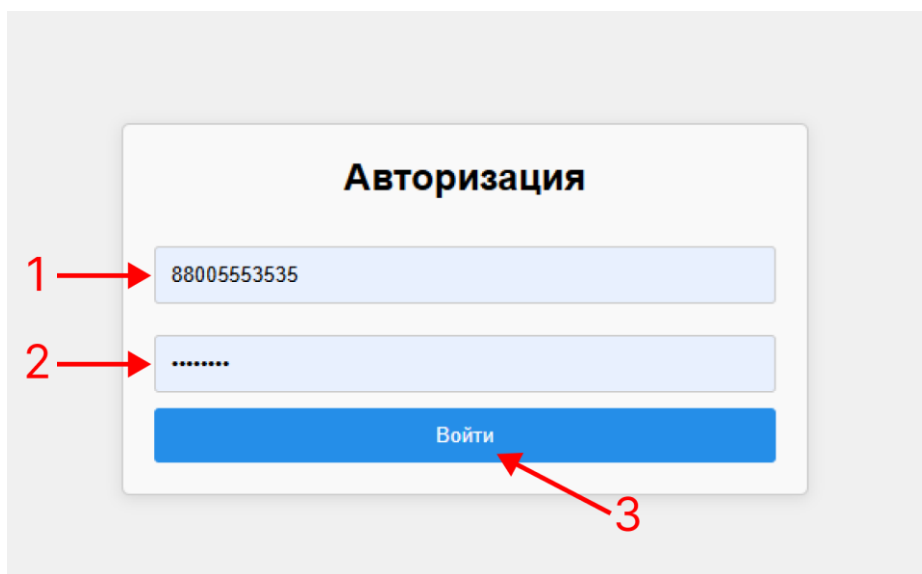


Рисунок 23 – Страница авторизации

Основные элементы страницы авторизации:

- 1 – поле ввода логина;
- 2 – поле ввода пароля;
- 3 – кнопка «Войти».

При нажатии на кнопку «Войти» система проверяет наличие зарегистрированного пользователя в системе. После корректной обработки данных пользователь, в зависимости от роли, попадает на страницу с расписанием, если пользователь инструктор или студент или на страницу с аккаунтами, если пользователь – администратор.

3.4.1 Реализация интерфейса студента

После авторизации студента (ученика) автошколы пользователь попадает на страницу с расписанием, представленную на рисунке 24.

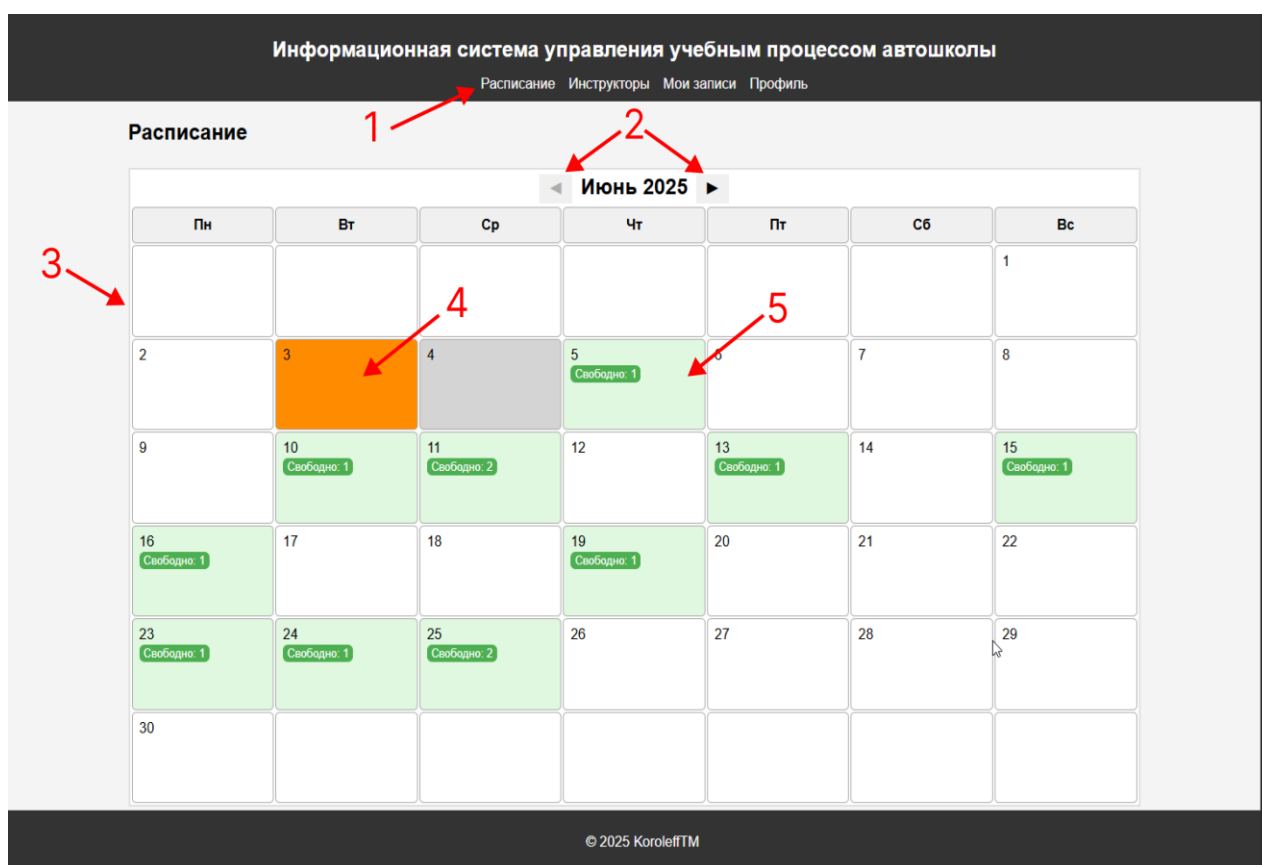


Рисунок 24 – Страница с расписанием (для студента)

Основные элементы страницы с расписанием (для студента):

- 1 – кнопки навигации по доступным страницам;
- 2 – кнопки переключения месяца;
- 3 – календарь с расписанием;
- 4 – текущая дата;
- 5 – ячейка со свободным занятием.

При нажатии на ячейку со свободным занятием откроется страница с карточками занятий на выбранный день (рисунок 25).

Основные элементы страницы с занятиями на выбранный день:

- 1 – карточка с занятием;
- 2 – время занятия;
- 3 – информация об инструкторе, проводящем занятие и его автомобиле;
- 4 – кнопка «Записаться».

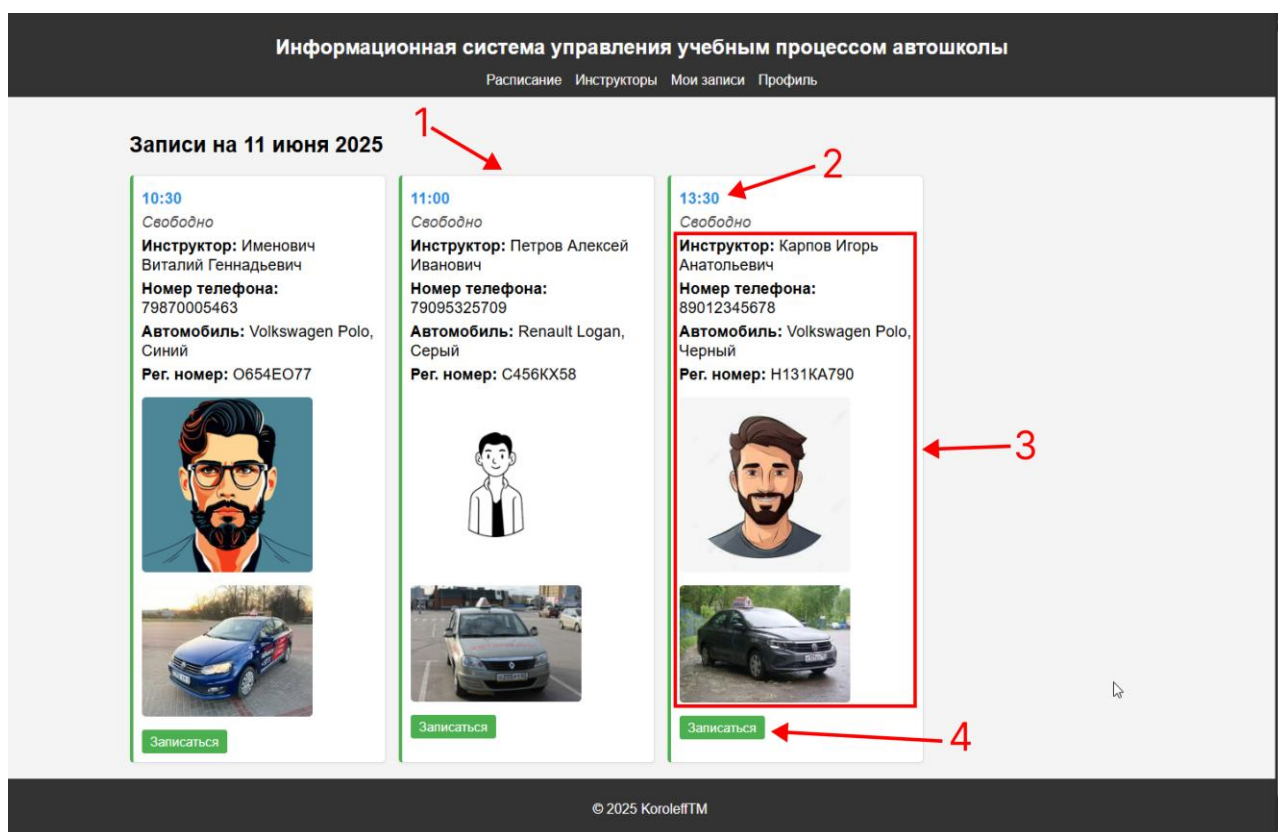


Рисунок 25 – Страница с занятиями на выбранный день

При нажатии на кнопку «Записаться» появляется модальное окно с подтверждением брони (рисунок 26).

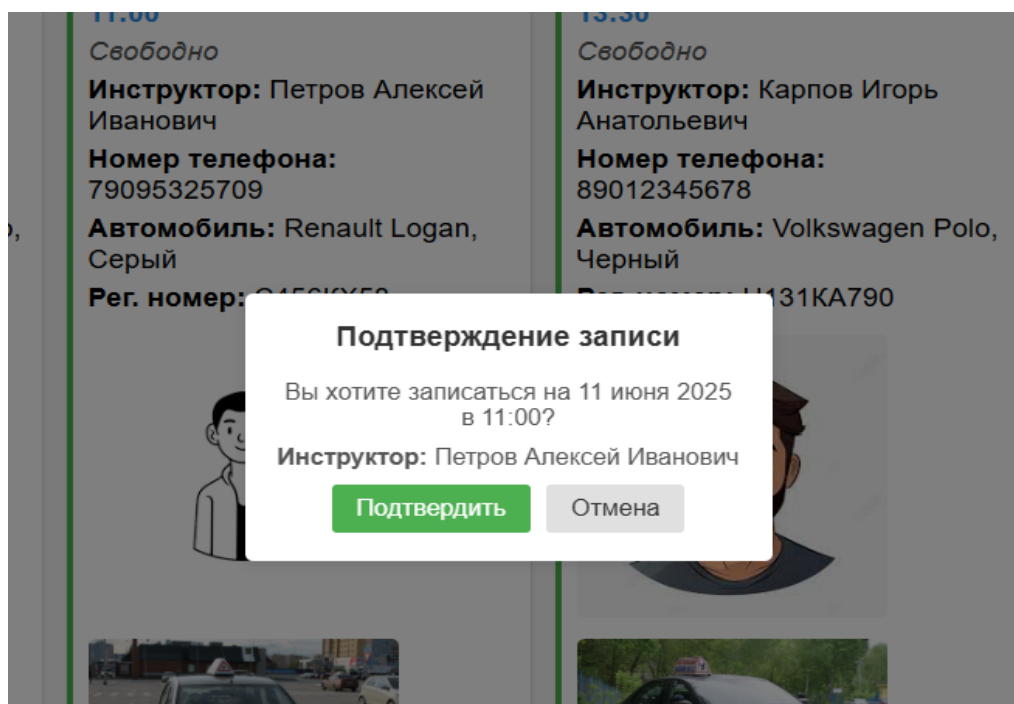


Рисунок 26 – Модальное окно подтверждения записи на занятие

При нажатии на кнопку «Подтвердить» появится уведомление об успешной записи (рисунок 27). При нажатии на кнопку «Отменить» модальное окно закроется.

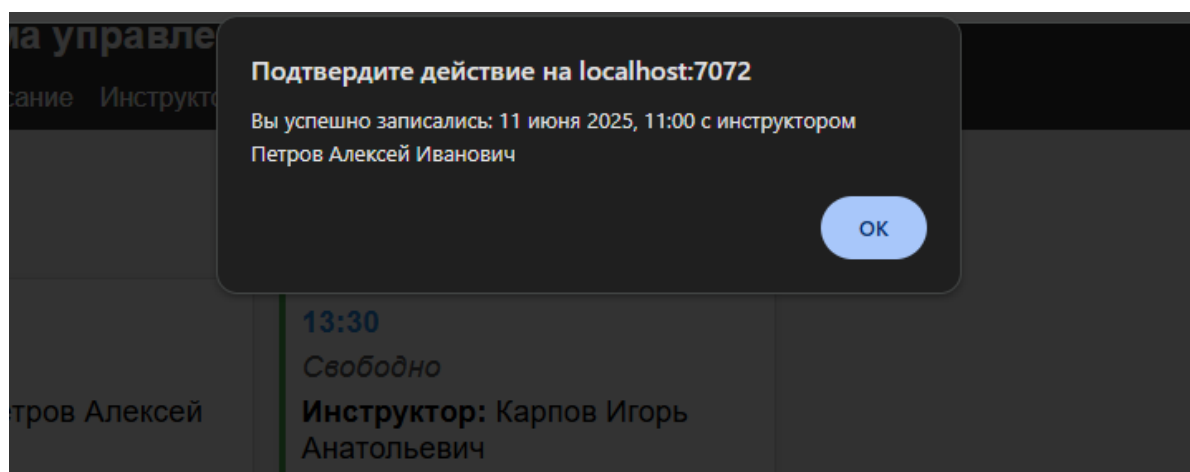


Рисунок 27 – Уведомление об успешной записи на занятие

При нажатии в уведомлении на кнопку «ОК» произойдет переход обратно к странице с расписанием.

При клике в шапке страницы на кнопку «Инструкторы» откроется страница со списком инструкторов (рисунок 28).

Информационная система управления учебным процессом автошколы
Расписание Инструкторы Мои записи Профиль

Список инструкторов

Фильтры

Фамилия:

Имя:

Отчество:

Телефон:

Email:

Город:

Автошкола:

ФИО	Телефон	Email	Автошкола	Автомобиль
Семенов Игорь Викторович	89374401202	siv@mail.ru	г. Пенза, ул. Пушкина, 44	Daewoo Nexia, Коричневый, P165XM58
Именович Виталий Геннадьевич	79870005463	sobaka@mail.ru	г. Пенза, ул. Германа-Титова, 38	Volkswagen Polo, Синий, O654EO77
Гойда Антон Романович	71242112512	goy@mail.ru	г. Пенза, ул. Германа-Титова, 38	Audi A4, Синий, P678OA58
Петров Алексей Иванович	79095325709	kas@mail.ru	г. Пенза, ул. Семенова, 24	Renault Logan, Серый, C456KX58
Никифоров Кирилл Петрович	79065432156	nkp@mail.ru	г. Пенза, ул. Пушкина, 44	Lada Largus, Серый, P892AK16
Фитилев Евгений Григорьевич	79941245121	feg@mail.ru	г. Пенза, ул. Германа-Титова, 38	Lada Granta, Белый, K987XM58
Кирсанов Виктор Павлович	79067845231	kvp@mail.ru	г. Пенза, ул. Пушкина, 44	Lada Kalina, Бордовый, Y412XM777
Симеонов Кирилл Петрович	79012345666	skp@mail.ru	г. Пенза, ул. Пушкина, 44	Lada Vesta, Коричневый, P225XM58
Сидоров Петр Петрович	89643567823	spp@mail.ru	г. Пенза, ул. Германа-Титова, 38	Skoda Fabia, Синий, P873TP77
Ивакин Игорь Викторович	85346782398	iva@gmail.com	г. Пенза, ул. Семенова, 24	Chevrolet Aveo, Серый, B478KP58

Кол-во на страницу:

Назад Страница 1 из 2 Вперед

© 2025 KoroleffTM

Рисунок 28 – Страница со списком инструкторов

Основные элементы страницы со списком инструкторов:

- 1 – таблица со списком инструкторов, содержащая информацию об инструкторах, контактные данные и привязку к автошколе;
- 2 – фильтры списка инструкторов;
- 3 – раскрывающийся список выбора количества отображаемых записей на страницу;
- 4 – кнопки переключения страниц.

На странице есть возможность отфильтровать список инструкторов по ФИО, номеру телефона, электронной почте, городу и по конкретной автошколе. При смене количества отображаемых записей на страницу происходит пересчет страниц для их пагинации. Пример фильтрации и изменения количества элементов на страницу показаны на рисунке 29.

Список инструкторов					
Фильтры	ФИО	Телефон	Email	Автошкола	Автомобиль
Фамилия: <input type="text" value="ива"/> Имя: <input type="text" value="Введите имя..."/> Отчество: <input type="text" value="Введите отчество..."/> Телефон: <input type="text" value="Введите номер телефона..."/> Email: <input type="text" value="@mail"/> Город: <input type="text" value="Любой"/> Автошкола: <input type="text" value="Любая"/> <input type="button" value="Применить"/> <input type="button" value="Сбросить фильтры"/>	Нива Кирилл Петрович	79065432156	nkp@mail.ru	г. Пенза, ул. Пушкина, 44	Lada Largus, Серый, P892AK16
	Иванов Кирилл	89765432389	ki@mail.com	г. Пенза, ул. Семенова, 24	Chevrolet Lacetti, Серый, T631BY77
	Ивакуров Артем Григорьевич	89003458967	kurov@mail.ru	г. Нижний Ломов, ул. Иванова, 52	Opel Vectra, Красный, O678EO58
	Криваев Антон Федорович	89067864512	fedor@mail.ru	г. Нижний Ломов, ул. Иванова, 52	Chevrolet Matiz, Зеленый, L234AK67
	Кучиваров Сергей	89456896723	sergo21@mail.com	г. Самара, ул. Шумахера, д. 5	Не указан

Кол-во на страницу: 5

Назад Страница 2 из 3 Вперед

Рисунок 29 – Отфильтрованный список инструкторов

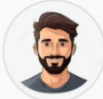
Если нажать на запись в таблице, откроется страница с подробной информацией по конкретному инструктору с данными инструктора и блоком с отзывами (рисунок 30).

Информационная система управления учебным процессом автошколы
Расписание Инструкторы Мои записи Профиль

1

ФИО: Карпов Игорь Анатольевич
Телефон: 89012345678
Email: karovsa87@gmail.com
Автошкола: г. Самара, ул. Шумахера, д. 5
Рейтинг: 5

Марка: Volkswagen
Модель: Polo
Цвет: Белый
Номер: O228AO58




2

Отзывы
Иванов Иван Иванович 31.05.2025
★★★★★
Отличный инструктор, доходчиво объясняет материал. Ездить с ним одно удовольствие!

3
Оставить отзыв
Введите текст отзыва...

4
Ваша оценка:
★★★★★

5
Оставить отзыв

© 2025 KorolefITM

Рисунок 30 – Страница информации об инструкторе

Основные элементы страницы с информацией об инструкторе:

1 – данные инструктора и его авто;

- 2 – отзывы на инструктора;
- 3 – текстовое поле для написания отзыва на инструктора;
- 4 – оценка инструктора («звездочки»);
- 5 – кнопка «Оставить отзыв».

На странице с данными инструктора можно просмотреть его рейтинг и отзывы от других студентов. В отзывах отображается ФИО автора, дата публикации, оценка («звездочки») и текст отзыва. Так же имеется возможность оставить отзыв авторизованному студенту, для этого нужно выбрать оценку, написать текст и нажать кнопку «Оставить отзыв».

При нажатии в шапке страницы кнопки «Мои записи» откроется страница с активными записями инструктора (рисунок 31).

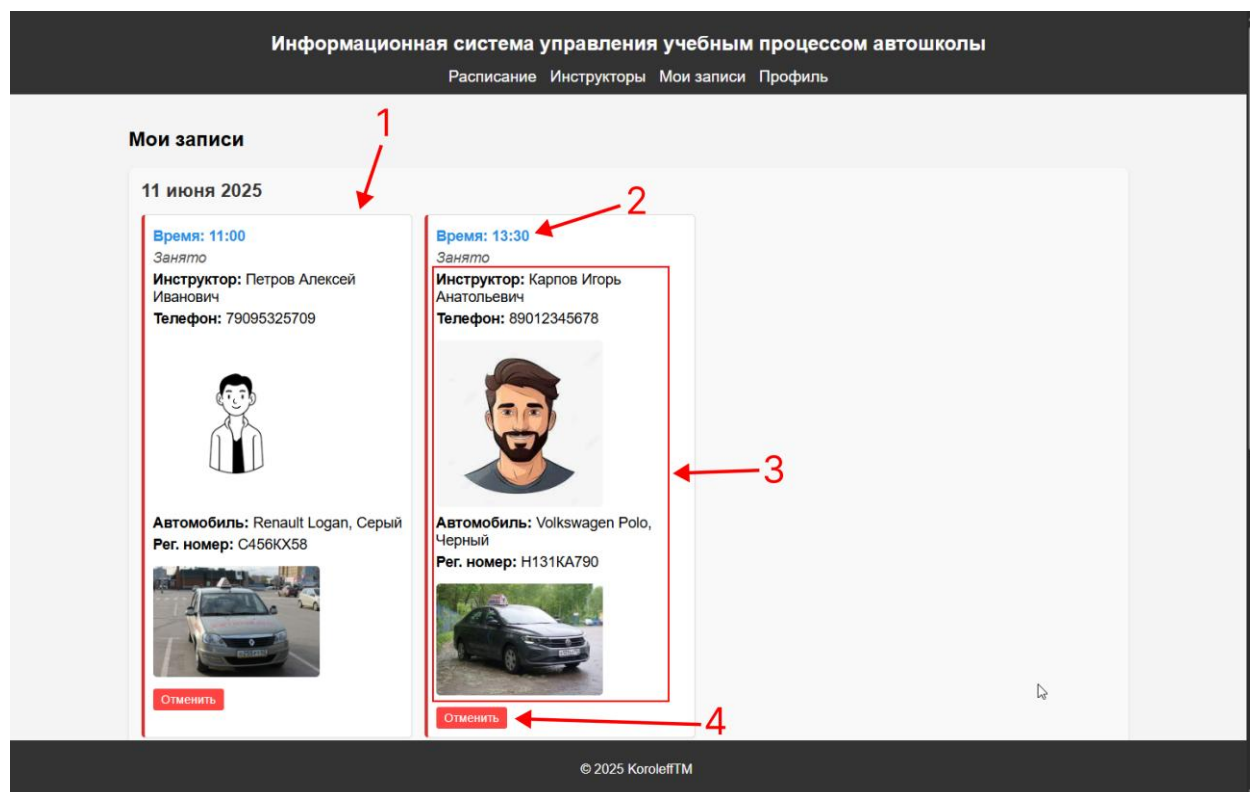


Рисунок 31 – Страница с записями на занятия (для студента)

Основные элементы страницы с забронированными занятиями студента:

- 1 – карточка с забронированным занятием;
- 2 – время занятия;
- 3 – информация об инструкторе, проводящем занятие и его автомобиле;

4 – кнопка «Отменить».

На странице «Мои записи» в отдельных компонентах (карточках) отображаются занятия, на которые записался авторизованный студент. В карточках отображается информация об авто и инструкторе, у которого будет проводиться занятие, фото инструктора и фото его автомобиля. Карточки отображаются по датам и отсортированы по времени. При нажатии в карточке с занятием на кнопку «Отменить» появляется модальное окно с подтверждением отмены записи на занятие (рисунок 32).

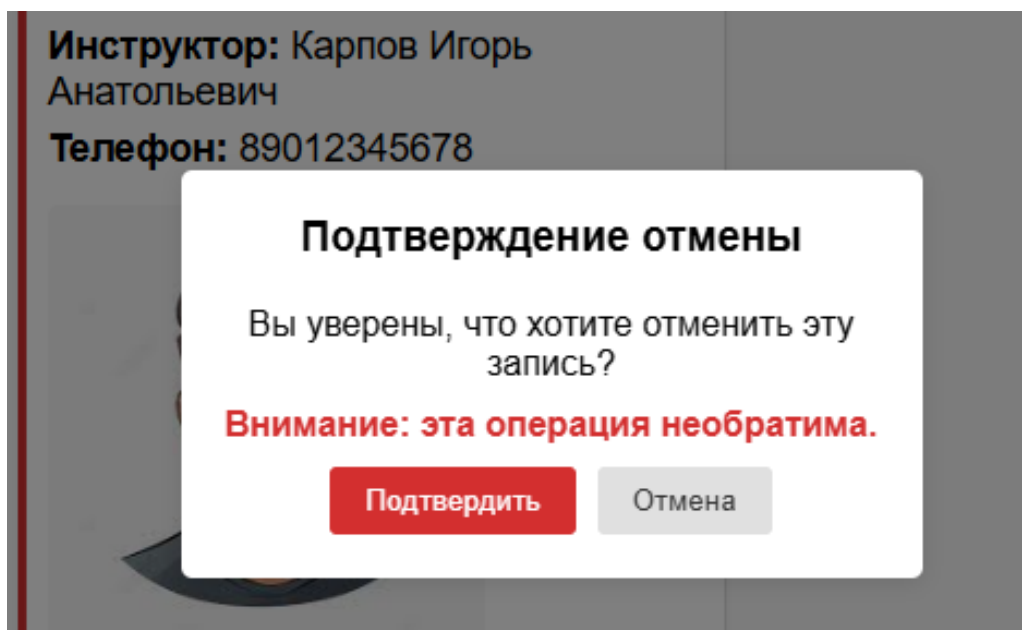


Рисунок 32 – Модальное окно подтверждения отмены записи на занятие

После подтверждения появляется уведомление, что запись на занятие отменена (рисунок 33).

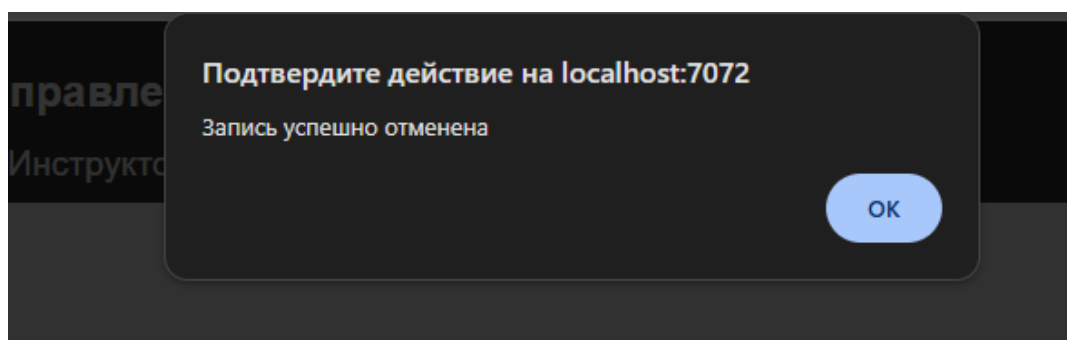


Рисунок 33 – Уведомление об отмене записи на занятие

При нажатии в шапке страницы на кнопку «Профиль» откроется страница с данными авторизованного студента (рисунок 34).

Основные элементы страницы профиля авторизованного студента:

- 1 – загруженное фото студента;
- 2 – кнопка изменения изображения профиля;
- 3 – данные авторизованного пользователя (студента);
- 4 – кнопка «Редактировать профиль»;
- 5 – кнопка «Выйти».

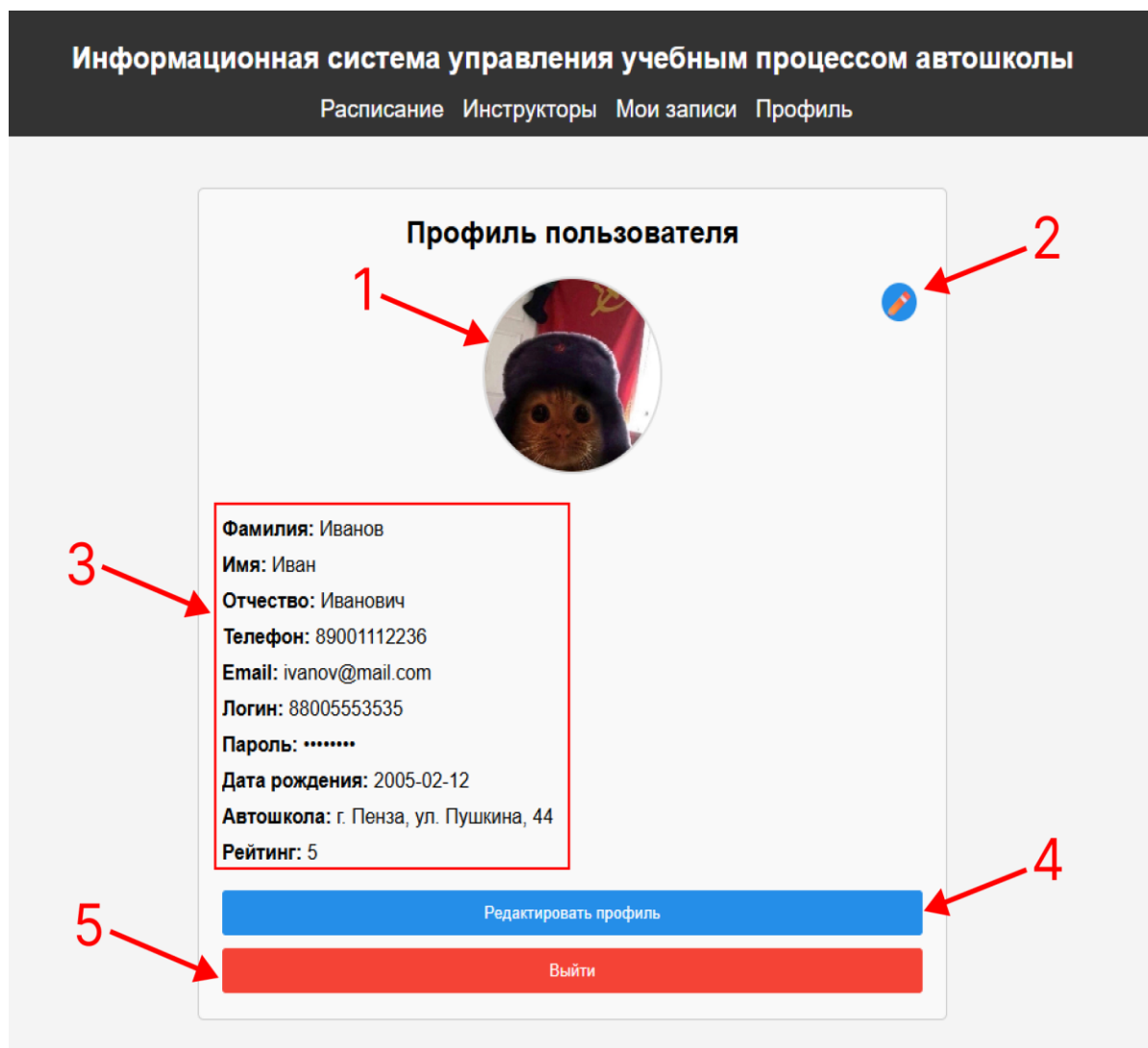


Рисунок 34 – Страница с профилем авторизованного студента

При нажатии на кнопку изменения изображения профиля (иконка карандаша) открывается системное окно, где можно выбрать фото, которое будет отображаться в профиле (рисунок 35).

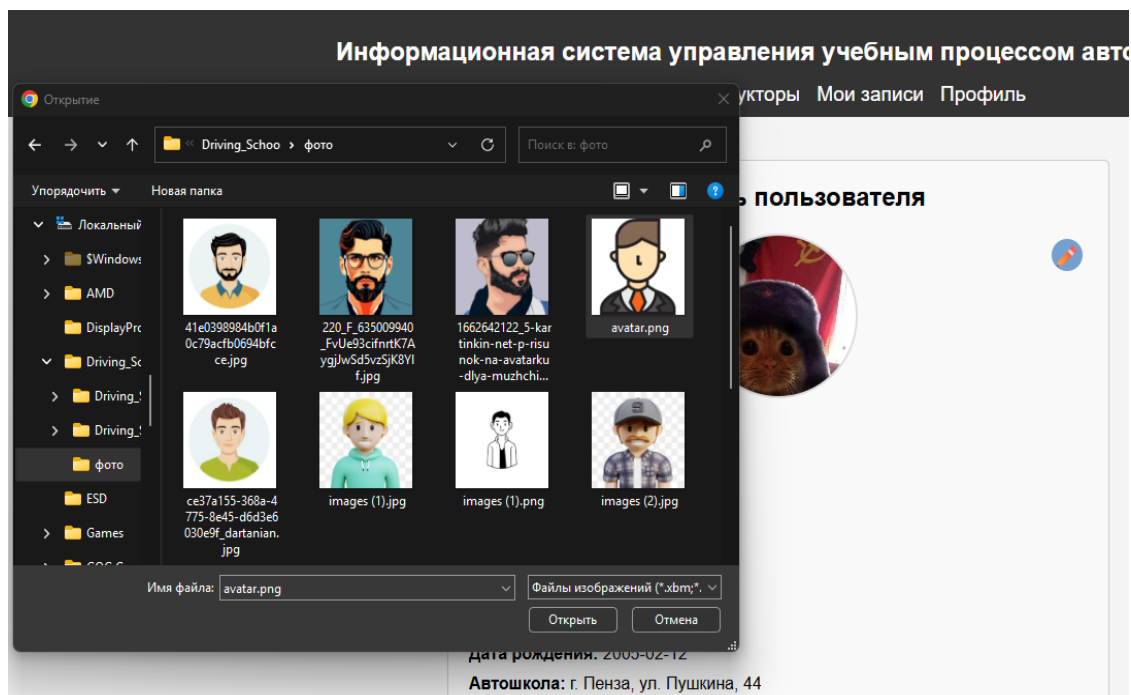


Рисунок 35 – Изменение изображения профиля

При нажатии на кнопку «Редактировать профиль» поля с данными становятся доступными для редактирования (рисунок 36).

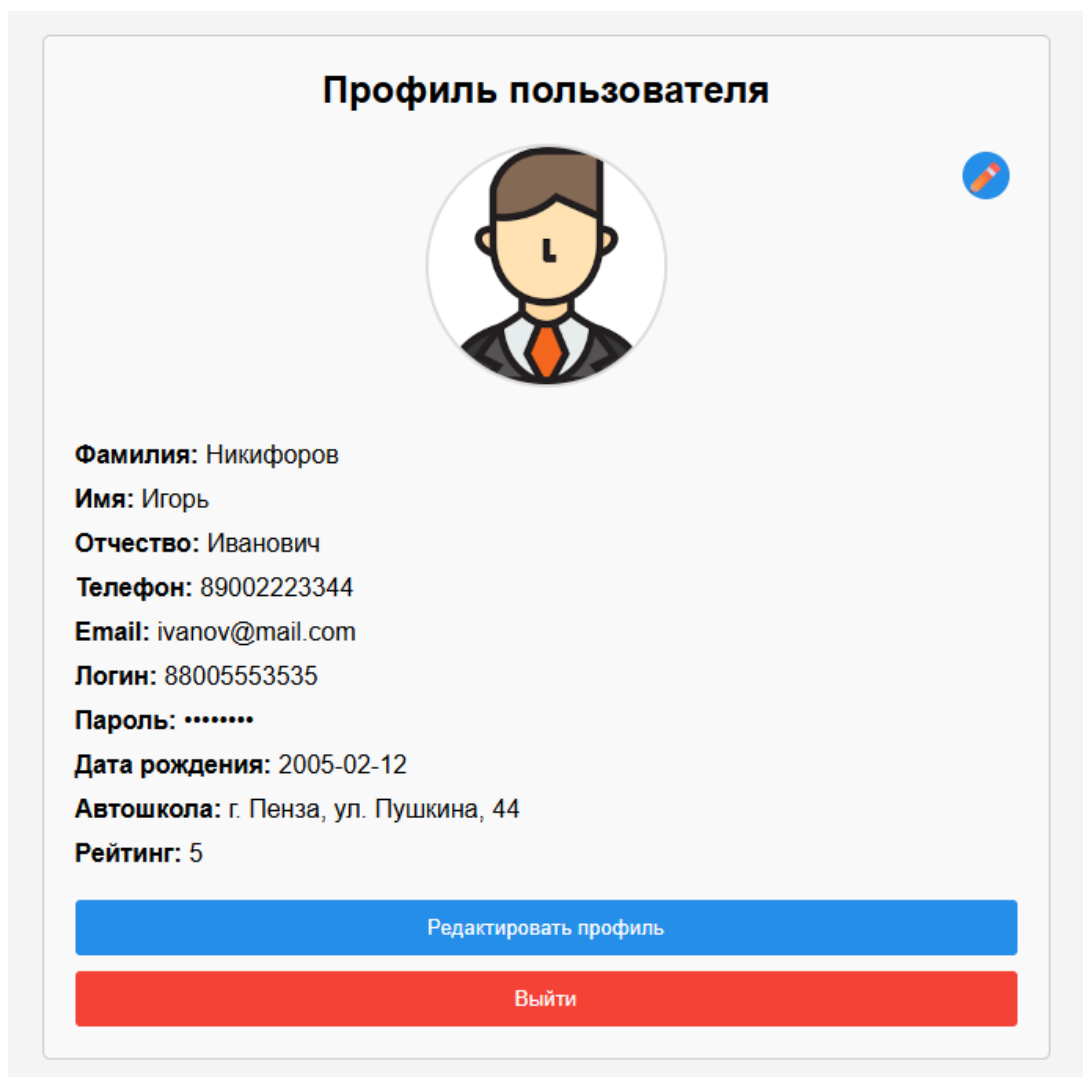
The screenshot shows a user profile form titled 'Профиль пользователя'. The form contains the following fields and values:

- Фамилия: Никифоров
- Имя: Игорь
- Отчество: Иванович
- Телефон: 89002223344
- Email: ivanov@mail.com
- Логин: 88005553535
- Пароль: Введите новый пароль
- Дата рождения: 2005-02-12
- Автошкола: г. Пенза, ул. Пушкина, 44
- Рейтинг: 5

At the bottom of the form, there are three buttons: 'Сохранить' (Save) in green, 'Отмена' (Cancel) in red, and 'Выйти' (Logout) in red.

Рисунок 36 – Редактирование профиля

После внесения изменений необходимо нажать кнопку «Сохранить» и в профиле отобразятся обновленные данные (рисунок 37). При нажатии на кнопку «Отмена» режим редактирования закрывается, данные не изменяются.



Профиль пользователя



Фамилия: Никифоров
Имя: Игорь
Отчество: Иванович
Телефон: 89002223344
Email: ivanov@mail.com
Логин: 88005553535
Пароль:
Дата рождения: 2005-02-12
Автошкола: г. Пенза, ул. Пушкина, 44
Рейтинг: 5

[Редактировать профиль](#)

[Выйти](#)

Рисунок 37 – Обновленный профиль

При нажатии на кнопку «Выйти» происходит выход из аккаунта, и пользователь попадает на главную публичную страницу системы.

3.4.2 Реализация интерфейса инструктора

При авторизации с данными инструктора пользователь попадает на страницу «Расписание». На странице отображается календарь с расписанием занятий и кнопка «Добавить» (рисунок 38). Функционал календаря на странице аналогичен тому, что и для студента.

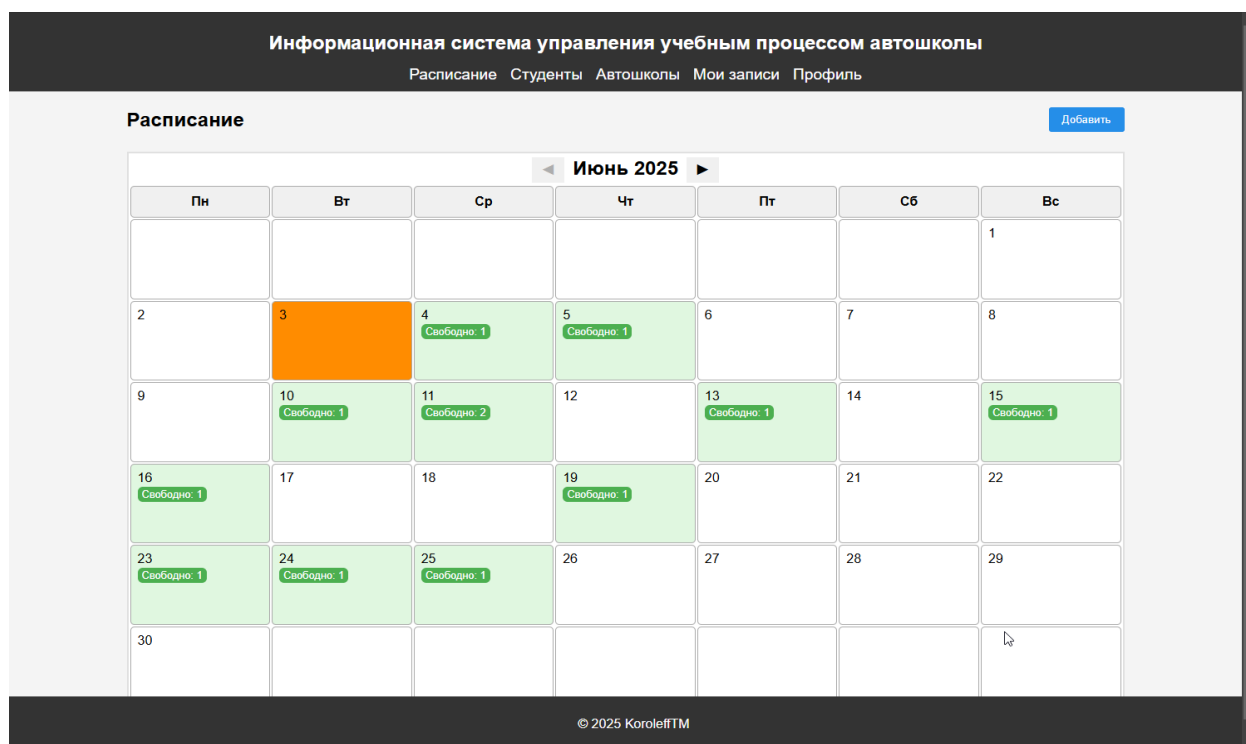


Рисунок 38 – Страница с расписанием занятий (для инструктора)

При нажатии на кнопку «Добавить» появляется модальное окно добавления нового занятия в расписание (рисунок 39).

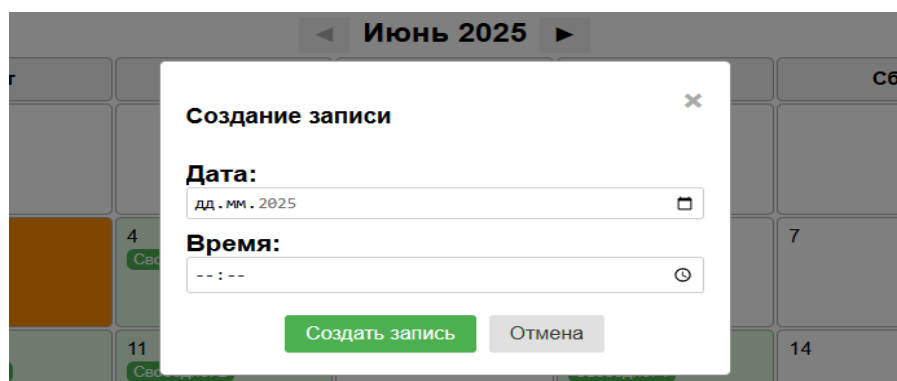


Рисунок 39 – Модальное окно добавления занятия в расписание

При нажатии на кнопку «Создать запись» происходит проверка введенных даты и времени и, если значения валидны (дата не в прошлом, время в промежутке с 8:00 до 20:00), занятие добавляется в расписание.

При нажатии в календаре на ячейку со свободным занятием открывается страница с занятиями на выбранный день. Помимо свободных занятий инструктору отображаются карточки забронированных занятий (рисунок 40).

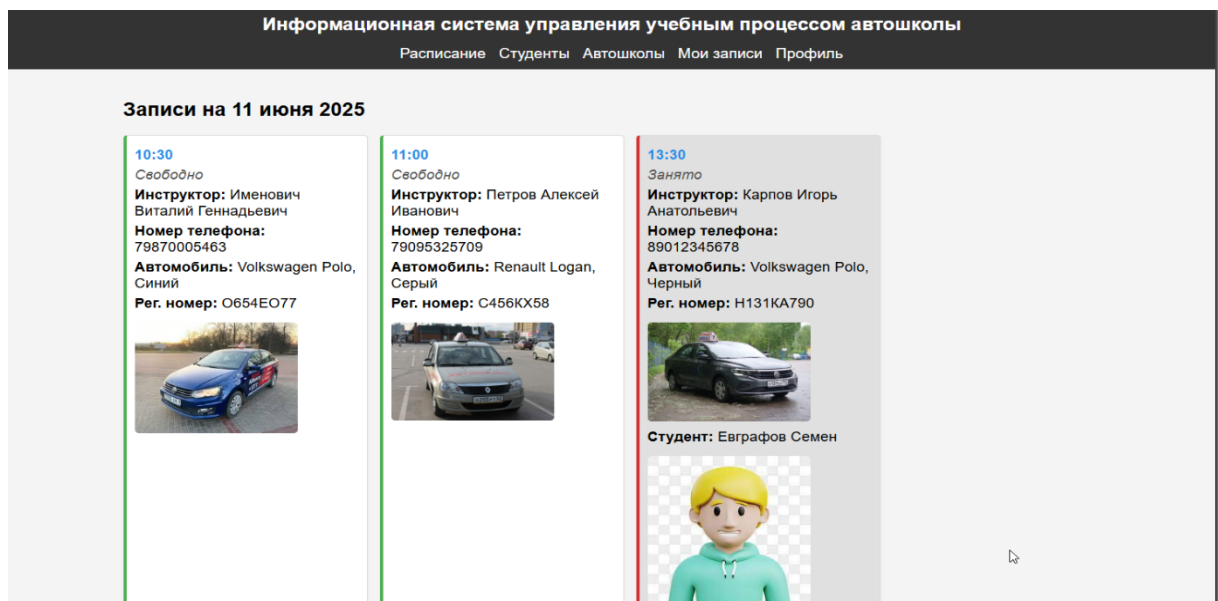


Рисунок 40 – Страница с занятиями на выбранный день (для инструктора)

При нажатии в шапке страницы на кнопку «Студенты» откроется страница со списком студентов (рисунок 41).



Рисунок 41 – Страница со списком студентов (для инструктора)

Функционал страницы аналогичен странице с инструкторами для студента (фильтрация, переключение страниц, выбор количества записей на страницу).

При нажатии на запись в таблице открывается страница с информацией студента и отзывами от инструкторов (рисунок 42). Функционал страницы аналогичен странице с информацией об инструкторе.

Информационная система управления учебным процессом автошколы
Расписание Студенты Автошколы Мои записи Профиль

ФИО: Никифоров Игорь Иванович
Телефон: 89002223344
Email: ivanov@mail.com
Дата рождения: 12.02.2005
Автошкола: г. Пенза, ул. Пушкина, 44
Рейтинг: 4.5



Отзывы

Карпов Игорь Анатольевич
08.05.2025
★★★★★
Отличный студент, всегда приходит на занятия полный энтузиазма.

Именович Виталий Геннадьевич
01.06.2025
★★★★☆
Часто опаздывает.

Оставить отзыв

Введите текст отзыва...

Ваша оценка:
★★★★☆

Оставить отзыв

Рисунок 42 – Страница с информацией о студенте

Если нажать в шапке страницы на кнопку «Автошколы», то произойдет переход на страницу со списком автошкол (рисунок 43).

Информационная система управления учебным процессом автошколы
Расписание Студенты Автошколы Мои записи Профиль

Список автошкол

Фильтры

Город:
Любой

Адрес:
Введите адрес...

Применить
Сбросить фильтры

Адрес	Город	Телефон	Email
ул. Пушкина, 44	Пенза	+78005553535	autoschoolpnz@mail.ru
ул. Германа-Титова, 38	Пенза	89001002030	user@example.com
ул. Иванова, 52	Нижний Ломов	88001002030	ivanova@mail.ru
ул. Курчанова, 73	Нижний Ломов	88007739401	latae@mail.ru
ул. Семенова, 24	Пенза	88007739405	semen@mail.ru
ул. Шумахера, д. 5	Самара	88413456789	samara@autoschool.ru
ул. Советская, д. 6	Саратов	89079056745	saratov-auto@gmail.com
ул. Комарова, д. 32	Рязань	88412456789	ryazan-school@mail.ru
ул. Московская, д. 71, помещение 1	Самара	88412333356	samaradrive@mail.ru

Кол-во на страницу: 10
Назад
Страница 1 из 1
Вперёд

Рисунок 43 – Страница со списком автошкол (для инструктора)

На странице отображается таблица с колонками «Адрес», «Город», «Телефон», «Email» и компонент с фильтрами по городам и адресам автошкол.

При нажатии на кнопку «Профиль» в шапке страницы откроется страница с данными авторизованного инструктора (рисунок 44).

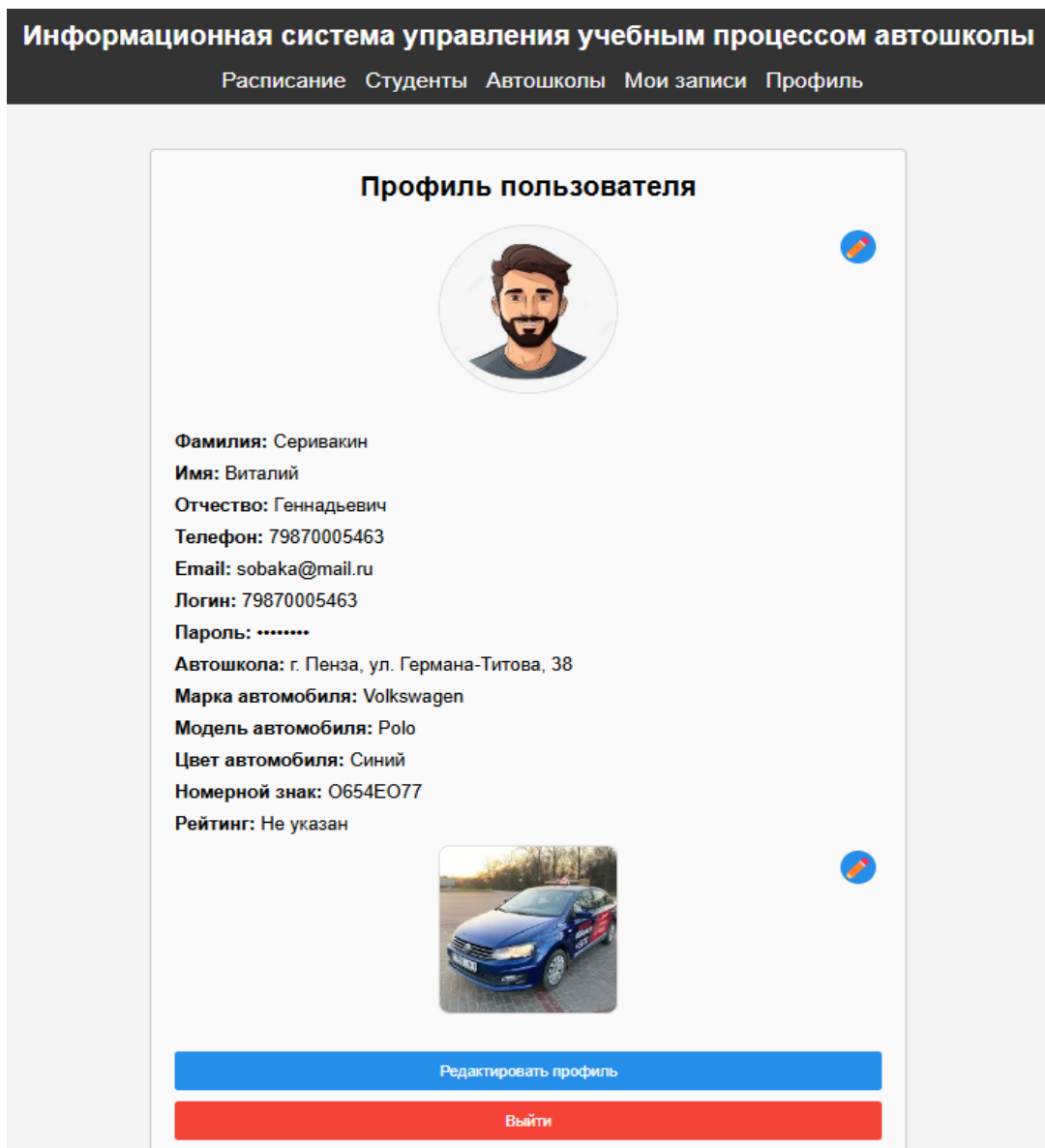


Рисунок 44 – Страница с профилем инструктора

Помимо информации об инструкторе на странице отображаются данные автомобиля инструктора. Функционал страницы аналогичен странице профиля студента (можно редактировать личные, учетные данные, а также данные авто). Имеется также возможность изменения фото профиля инструктора и изменение фото автомобиля.

При нажатии на кнопку «Выйти» происходит выход пользователя из аккаунта и переход на главную публичную страницу информационной системы.

3.4.3 Реализация интерфейса администратора

При авторизации с учетными данными администратора пользователь попадает на страницу «Аккаунты». На странице отображается список аккаунтов пользователей системы (рисунок 45).

Информационная система управления учебным процессом автошколы

Студенты Инструкторы Автошколы Автомобили Аккаунты Профиль

Список аккаунтов

Фильтры

Логин (номер телефона):
Введите логин...

Тип пользователя:
Любой

ФИО:
Введите ФИО...

Город:
Любой

Автошкола:
Любая

Применить Сбросить фильтры

Логин (номер телефона)	Пароль	Тип пользователя	ФИО	Город	Автошкола	Действия
88005553535	пароль	Студент	Никифоров Игорь Иванович	Пенза	ул. Пушкина_44	3 → ✎ 🗑
89001112234	пароль	Студент	Фамилия Имя Отчество	Пенза	ул. Германа-Титова_38	✎ 🗑
89012345678	пароль	Инструктор	Карпов Игорь Анатольевич	Самара	ул. Шумахера_д.5	✎ 🗑
89001112235	пароль	Студент	Ципролет Парацетамол Компливитович	Пенза	ул. Пушкина_44	✎ 🗑
89081112234	пароль	Студент	Евграфов Семен Игоревич	Пенза	ул. Пушкина_44	✎ 🗑
79870005463	пароль	Инструктор	Серивакин Виталий Геннадьевич	Пенза	ул. Германа-Титова_38	✎ 🗑
71242112512	пароль	Инструктор	Кривашеев Антон Романович	Пенза	ул. Германа-Титова_38	✎ 🗑
логин	пароль	Инструктор	Петров Алексей Иванович	Пенза	ул. Семенова_24	✎ 🗑
89001112233	пароль	Инструктор	Нива Кирилл Петрович	Пенза	ул. Пушкина_44	✎ 🗑
79074561723	пароль	Студент	Кротов Еж Витальевич	Пенза	ул. Германа-Титова_38	✎ 🗑

Записей на странице: 10

Назад Страница 1 из 3 Вперед

© 2025 KoroleffTM

Рисунок 45 – Страница со списком аккаунтов

Основные элементы страницы со списком аккаунтов для администратора:

- 1 – таблица со списком аккаунтов;
- 2 – кнопка «Добавить» для добавления нового аккаунта;
- 3 – колонка «Действия» с кнопками редактирования и удаления выбранной записи;
- 4 – блок с фильтрами;
- 5 – выбор количества отображаемых записей на страницу;
- 6 – переключатель страницы.

При наведении курсора на ячейку в колонке «Пароль» значение станет видимым для пользователя (рисунок 46).

Логин (номер телефона)	Пароль	Тип пользователя
88005553535	password	Студент
89001112234		Студент

Рисунок 46 – Наведенный курсор на ячейку с паролем

Если нажать на ячейку в колонке «Автошкола» отобразятся контакты данной автошколы (рисунок 47).

Город	Автошкола	Действия
Пенза	ул. Пушкин, 44 Телефон: +78005553535, Email: autoschoolpnz@mail.ru	 
Пенза	ул. Германа-Титова, 38 Телефон: 89001002030, Email: user@example.com	 
Самара	ул. Шумахера, д. 5	 

Рисунок 47 – Наведенный курсор на ячейку с паролем

При нажатии на кнопку редактирования записи (иконка карандаша) появится модальное окно редактирования данной записи (рисунок 48). В полях модального окна отобразятся значения логина и пароля для выбранного аккаунта, доступные для редактирования. После внесения изменений при нажатии на кнопку «Сохранить» запись обновится и в таблице отобразятся обновленные данные.

Рисунок 48 – Наведенный курсор на ячейку с паролем

При нажатии в колонке «Действия» на кнопку удаления (иконка корзины) появится модальное окно подтверждения удаления аккаунта с предупреждением (рисунок 49). При нажатии на «Подтвердить» запись удалится.

Рисунок 49 – Модальное окно подтверждения удаления аккаунта

При нажатии над таблицей на кнопку «Добавить» откроется модальное окно добавления нового аккаунта в систему. В модальном окне отображается раскрывающийся список выбора типа добавляемого пользователя (рисунок 50).

Рисунок 50 – Модальное окно добавления нового аккаунта

При выборе типа пользователя отобразятся соответствующие поля для заполнения (рисунок 51).

Добавление аккаунта

Тип пользователя: *

Студент

Фамилия: *

Введите фамилию

Имя: *

Введите имя

Отчество (если есть):

Введите отчество

Номер телефона (логин): *

Введите номер телефона

Пароль: *

.....

День рождения: *

дд . мм . гggг

Email:

Введите email

Город: *

Выберите город

Автошкола: *

Выберите автошколу

Сохранить **Отмена**

Рисунок 51 – Модальное окно добавления нового аккаунта студента

Для инструктора помимо полей для личной информации отображаются поля для добавления автомобиля: марка, модель, цвет и регистрационный номер (рисунок 52).

Добавление аккаунта

Отчество (если есть):

Введите отчество

Номер телефона (логин): *

Введите номер телефона

Пароль: *

.....

Email: *

Введите email

Город: *

Выберите город

Автошкола: *

Выберите автошколу

Марка автомобиля: *

Введите марку

Модель автомобиля: *

Введите модель

Цвет автомобиля: *

Введите цвет

Регистрационный номер: *

Введите номер

Сохранить **Отмена**

Рисунок 52 – Модальное окно добавления нового аккаунта инструктора

Поля модального окна добавления аккаунта администратора схожи с аккаунтом студента (рисунок 53).

Добавление аккаунта

Тип пользователя: *
Администратор

Фамилия: *
Введите фамилию

Имя: *
Введите имя

Отчество (если есть):
Введите отчество

Номер телефона (логин): *
Введите номер телефона

Пароль: *
.....

Email:
Введите email

Город: *
Выберите город

Автошкола: *
Выберите автошколу

Сохранить **Отмена**

Рисунок 53 – Модальное окно добавления нового аккаунта администратора

После нажатия на кнопку «Сохранить» система проверяет корректность введенных данных и создает новую запись.

При нажатии в шапке страницы на кнопку «Студенты» открывается страница со списком студентов (рисунок 54). На странице можно фильтровать, редактировать и удалять записи, как на странице «Аккаунты».

Информационная система управления учебным процессом автошколы						
Студенты Инструкторы Автошколы Автомобили Аккаунты Профиль						
Список студентов						
ФИО	Телефон	Email	Дата рождения	Автошкола	Действия	
Никифоров Игорь Иванович	89002223344	ivanov@mail.com	12.02.2005	г. Пенза, ул. Пушкина, 44		
Сидоров Иван Петрович	89085554545	user@example.com	22.08.1992	г. Пенза, ул. Пушкина, 44		
Фамилия Имя Отчество	89001112234	test@mail.ru	09.01.2001	г. Пенза, ул. Германа-Титова, 38		
Ципролет Парацетамол Компливитович	89001112235	tester@mail.ru	13.12.2007	г. Пенза, ул. Пушкина, 44		
Евграфов Семен Игоревич	89081112234	esi@mail.ru	24.04.1973	г. Пенза, ул. Пушкина, 44		
Кротов Еж Витальевич	79074561723	Не указан	16.03.1984	г. Пенза, ул. Германа-Титова, 38		
Петров Петр Иванович	89056734523	Не указан	10.12.2002	г. Пенза, ул. Пушкина, 44		
Семенов Иван Петрович	89054567823	Не указан	12.03.2000	г. Пенза, ул. Семенова, 24		
Курчатова Виктор Федорович	89374523489	kivi@gmail.com	17.02.1990	г. Пенза, ул. Германа-Титова, 38		
Никитина Анна Вячеславовна	79045681234	vya@mail.ru	30.04.2004	г. Пенза, ул. Германа-Титова, 38		

Рисунок 54 – Страница со списком студентов (для администратора)

При нажатии на запись откроется страница с информацией о студенте, (рисунок 55). Для администратора доступен функционал удаления отзывов.

ФИО: Никифоров Игорь Иванович
Телефон: 89002223344
Email: ivanov@mail.com
Дата рождения: 12.02.2005
Автошкола: г. Пенза, ул. Пушкина, 44
Рейтинг: 4.5

Отзывы

Карпов Игорь Анатольевич 08.05.2025
 ★★★★★
 Отличный студент, всегда приходит на занятия полный энтузиазма.
 Удалить

Серивакин Виталий Геннадьевич 01.06.2025
 ★★★★★☆
 Часто опаздывает.
 Удалить

Оставить отзыв

Введите текст отзыва...

Ваша оценка:
 ★★★★★

Оставить отзыв

Рисунок 55 – Страница с информацией о студенте (для администратора)

При нажатии в шапке страницы на кнопку «Инструкторы» откроется аналогичная страница со списком инструкторов и функционалом редактирования и удаления записей (рисунок 56).

Информационная система управления учебным процессом автошколы
 Студенты Инструкторы Автошколы Автомобили Аккаунты Профиль

Список инструкторов

Фильтры

Фамилия:
 Введите фамилию...

Имя:
 Введите имя...

Отчество:
 Введите отчество...

Телефон:
 Введите номер телефона...

Email:
 Введите email...

Город:
 Пенза

Автошкола:
 Любая

Применить Сбросить фильтры

ФИО	Телефон	Email	Автошкола	Автомобиль	Действия
Семенов Игорь Викторович	89374401202	siv@mail.ru	г. Пенза, ул. Пушкина, 44	Daewoo Nexia, Коричневый, P165XM58	✎ 🗑
Серивакин Виталий Геннадьевич	79870005463	sobaka@mail.ru	г. Пенза, ул. Германа-Титова, 38	Volkswagen Polo, Синий, O654EO77	✎ 🗑
Кривашев Антон Романович	71242112512	goy@mail.ru	г. Пенза, ул. Германа-Титова, 38	Audi A4, Синий, P678OA58	✎ 🗑
Петров Алексей Иванович	79095325709	kas@mail.ru	г. Пенза, ул. Семенова, 24	Renault Logan, Серый, C456KX58	✎ 🗑
Нива Кирилл Петрович	79065432156	nkp@mail.ru	г. Пенза, ул. Пушкина, 44	Lada Largus, Серый, P892AK16	✎ 🗑
Фитилев Евгений Григорьевич	79941245121	feg@mail.ru	г. Пенза, ул. Германа-Титова, 38	Lada Granta, Белый, K987XM58	✎ 🗑
Кирсанов Виктор Павлович	79067845231	kvp@mail.ru	г. Пенза, ул. Пушкина, 44	Lada Kalina, Бордовый, U412XM777	✎ 🗑
Симеонов Кирилл Петрович	79012345666	skp@mail.ru	г. Пенза, ул. Пушкина, 44	Lada Vesta, Коричневый, P225XM58	✎ 🗑
Сидоров Петр Петрович	89643567823	spp@mail.ru	г. Пенза, ул. Германа-Титова, 38	Skoda Fabia, Синий, P873TP77	✎ 🗑
Ивакин Игорь Викторович	85346782398	iva@gmail.com	г. Пенза, ул. Семенова, 24	Chevrolet Aveo, Серый, B478KP58	✎ 🗑

Кол-во на страницу: 10

Назад Страница 1 из 2 Вперед

Рисунок 56 – Страница со списком инструкторов (для администратора)

Функционал страницы с инструкторами аналогичен странице со студентами.

Если нажать на кнопку «Автошколы», откроется страница со списком автошкол. На странице отображается таблица, блок с фильтром по городу и адресу автошколы, а также кнопка «Добавить» (рисунок 57).

Информационная система управления учебным процессом автошколы
Студенты Инструкторы Автошколы Автомобили Аккаунты Профиль

Список автошкол Добавить

Фильтры

Город:
Любой ▼

Адрес:
Введите адрес...
Применить Сбросить фильтры

Адрес	Город	Телефон	Email	Действия
ул. Пушкина, 44	Пенза	+78005553535	autoschoolpnz@mail.ru	
ул. Германа-Титова, 38	Пенза	89001002030	user@example.com	
ул. Иванова, 52	Нижний Ломов	88001002030	ivanova@mail.ru	
ул. Курчанова, 73	Нижний Ломов	88007739401	lala@mail.ru	
ул. Семенова, 24	Пенза	88007739405	semen@mail.ru	
ул. Шумахера, д. 5	Самара	88413456789	samara@autoschool.ru	
ул. Советская, д.6	Саратов	89079056745	saratov-auto@gmail.com	
ул. Комарова, д.32	Рязань	88412456789	ryazan-school@mail.ru	
ул. Московская, д. 71, помещение 1	Самара	88412333356	samaradrive@mail.ru	

Кол-во на страницу: 10 ▼ Назад Страница 1 из 1 Вперед

Рисунок 57 – Страница со списком автошкол (для администратора)

При нажатии на кнопку «Добавить» откроется модальное окно «Добавить автошколу» (рисунок 58).

Добавить автошколу

Адрес: *
Введите адрес...

Телефон: *
Введите телефон...

Email: *
Введите email...

Город: *
Выберите город ▼ Добавить новый город

Сохранить Отмена

Рисунок 58 – Модальное окно «Добавить автошколу»

Новую автошколу можно привязать к городу, который уже имеется в системе. При нажатии на кнопку «Сохранить» система проверяет заполнение всех обязательных полей и при успешной валидации создает новую запись в таблице.

При нажатии на кнопку «Автомобили» в шапке страницы, откроется страница со списком автомобилей, в котором отображаются данные о марке, модели, цвете, регистрационном номере и городе нахождения автомобиля (рисунок 59).

Информационная система управления учебным процессом автошколы						
Студенты Инструкторы Автошколы Автомобили Аккаунты Профиль						
Список автомобилей						
Фильтры		Марка	Модель	Цвет	Регистрационный номер	Город
Город:						
Пенза						
Регистрационный номер:						
Введите номер...						
Применить						
Сбросить фильтры						
		Daewoo	Nexia	Коричневый	P165XM58	Пенза
		Volkswagen	Polo	Синий	O654EO77	Пенза
		Audi	A4	Синий	P678OA58	Пенза
		Renault	Logan	Серый	C456KX58	Пенза
		Lada	Largus	Серый	P892AK16	Пенза
		Lada	Granta	Белый	K987XM58	Пенза
		Lada	Kalina	Бордовый	Y412XM777	Пенза
		Lada	Vesta	Коричневый	P225XM58	Пенза
		Skoda	Fabia	Синий	P873TP77	Пенза
		Chevrolet	Aveo	Серый	B478KP58	Пенза
Кол-во на страницу: 10		Назад Страница 1 из 2				

Рисунок 59 – Страница со списком автомобилей

Функционал страницы «Автомобили» аналогичен странице с автошколами: можно фильтровать автомобили по городу и регистрационному номеру, редактировать и удалять записи.

При нажатии на кнопку «Профиль» открывается страница с информацией об авторизованном администраторе (рисунок 60). Функционал страницы аналогичен страницам профиля студента и инструктора, предоставляя возможность просмотра и редактирования персональных и учетных данных, таких как ФИО, контакты, логин и пароль.

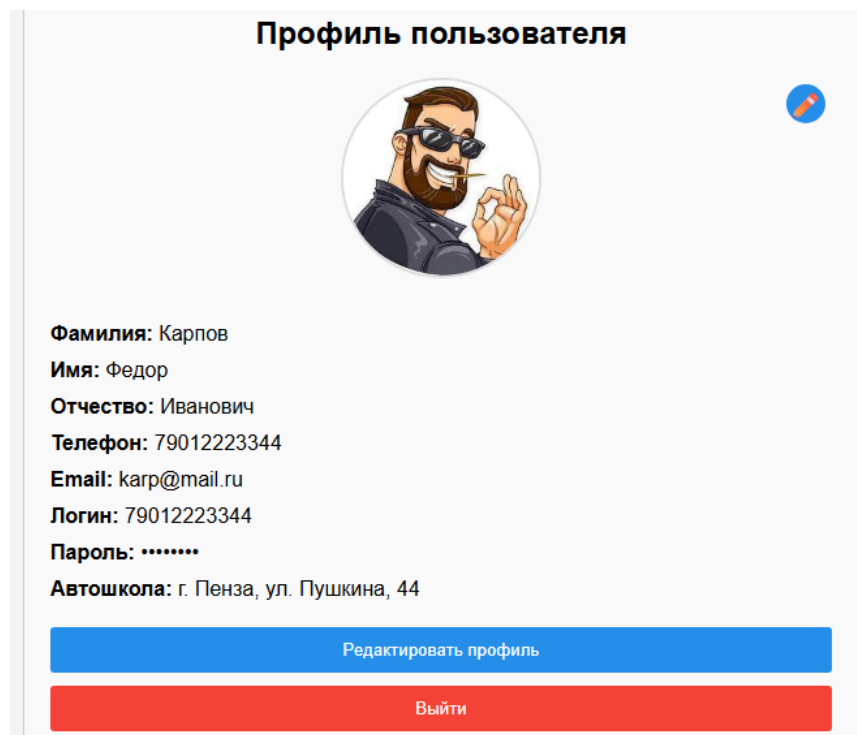


Рисунок 60 – Страница профиля администратора

Интерфейс системы разработан с учетом требований к удобству и функциональности. Используются технологии HTML, CSS и JavaScript, что обеспечивает кроссбраузерность, адаптивность и интерактивность страниц. Интерфейс предоставляет доступ ко всем ключевым функциям системы в соответствии с ролями пользователей, что способствует эффективному управлению учебным процессом автошколы.

4 Контроль качества программного обеспечения

4.1 Выбор режима тестирования

Тестирование – это важный этап разработки программного обеспечения. Оно позволяет убедиться, что система работает правильно, соответствует требованиям и удобна в использовании. Без тестирования невозможно гарантировать стабильность, надежность и корректную работу приложения в различных ситуациях [17].

Существует несколько основных видов тестирования: функциональное, модульное, интеграционное, производительности.

При функциональном тестировании проверяется, как система выполняет нужные функции. Тестируется, соответствует ли поведение программы ожиданиям пользователя и заданным требованиям. Модульное тестирование проверяет отдельные компоненты (например, функции, классы, сервисы) без запуска всей системы. Интеграционное тестирование проверяет взаимодействие между разными частями системы, обеспечивая их согласованную работу. При тестировании производительности идет проверка скорости и устойчивости системы под нагрузкой [18].

Для тестирования информационной системы управления учебным процессом автошколы был выбран комбинированный подход, включающий функциональное и модульное тестирование. Функциональное тестирование будет использоваться для проверки пользовательских сценариев на стороне клиента. Такой подход позволит проверить, как работают основные функции системы с точки зрения пользователя.

Модульное тестирование будет применяться для проверки логики отдельных компонентов на серверной части, например, контроллера занятий. Оно удобно тем, что каждый компонент можно протестировать отдельно от других. Это ускоряет процесс тестирования и позволяет легче находить и устранять ошибки.

Выбор этих двух видов тестирования объясняется особенностями архитектуры системы. Система построена по модульному принципу, а основные требования сосредоточены на корректной работе пользовательских функций. Поэтому выбранный подход обеспечит как проверку бизнес-логики, так и удобство использования системы на практике.

4.2 Тестирование серверной части информационной системы управления учебным процессом автошколы

Для проверки корректности работы серверной логики информационной системы было проведено модульное тестирование. Оно позволяет выявить ошибки на ранней стадии разработки и убедиться, что каждый компонент работает корректно в изоляции от остальных частей системы [19].

Для проведения модульного тестирования используются библиотеки xUnit (для написания тестов) и Moq (для проверки без подключения к реальной базе данных). В качестве объекта модульного тестирования выбран контроллер расписания занятий. Полный код модульных тестов приведен в приложении Г. Краткое описание модульных тестов приведено в таблице 19.

Таблица 19 – Модульные тесты контроллера занятий

Тест	Описание	Результат
CreateSchedule	Проверка корректности создания нового занятия в расписании	Пройден
UpdateSchedule_AddStudentId	Проверка записи студента на занятие (изменение статуса занятия на занятое)	Пройден
CancelScheduleByStudent	Проверка отмены записи студентом (изменение статуса занятия на свободное)	Пройден
DeleteSchedule	Проверка отмены занятия инструктором (удаление занятия из системы)	Пройден
FullLifecycle	Проверка полного жизненного цикла занятия (создание, запись студентом, отмена студентом и отмена инструктором)	Пройден

Результаты модульного тестирования контроллера занятий приведены на рисунке 61.

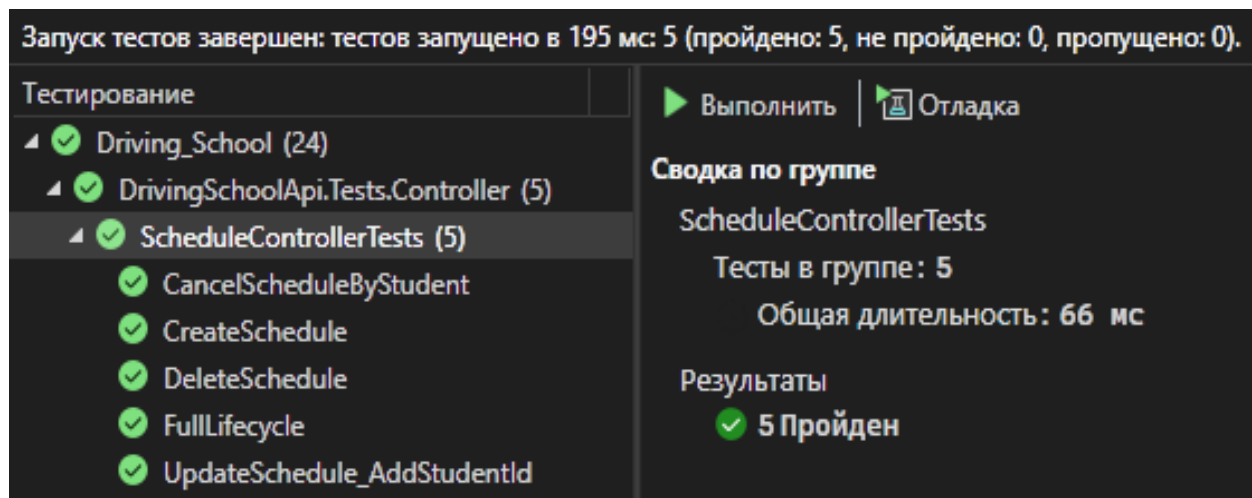


Рисунок 61 – Диаграмма состояний и переходов занятия

По результатам можно сделать вывод, что тесты пройдены успешно и контроллер занятий работает корректно. Таким образом система прошла этап модульного тестирования контроллера занятий.

4.3 Тестирование клиентской части информационной системы управления учебным процессом автошколы

После реализации системы было проведено функциональное тестирование. Результаты тестирования приведены в приложении Ж.

Для тестирования функционала занятий была разработана диаграмма состояний и переходов, представленная на рисунке 62. Она наглядно отражает поведение системы, показывая возможные состояния объекта и события, вызывающие переходы между ними. Диаграмма демонстрирует полный жизненный цикл записи на занятие.

Использование диаграммы состояний в процессе функционального тестирования помогает полноценно отследить возможные сценарии работы системы, проверить корректность переходов и реакцию системы на заданные действия. Это позволяет более тщательно проверить логику работы системы [20].

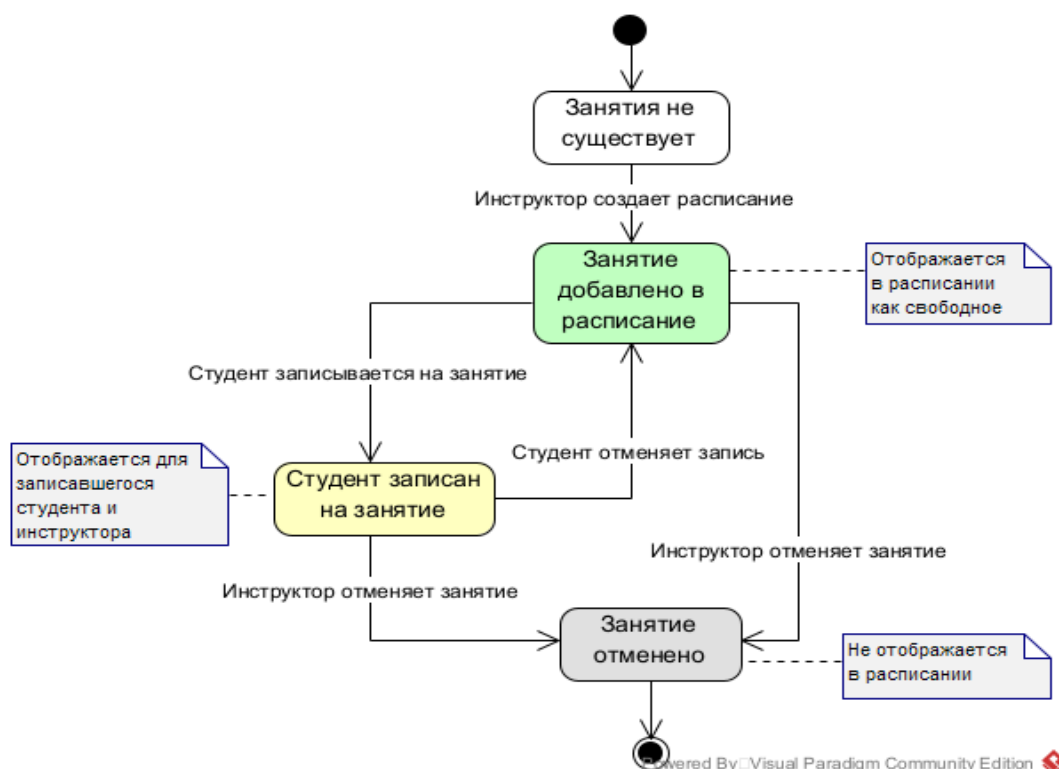


Рисунок 62 – Диаграмма состояний и переходов занятия

На основании диаграммы состояний и переходов занятия были созданы тестовые сценарии, описанные в таблице 20.

Таблица 20 – Тест-кейсы функционала занятий

Название	Шаги	Ожидаемый результат	Фактический результат
Создание занятия	<ol style="list-style-type: none"> 1. Авторизоваться как инструктор 2. Открыть страницу «Расписание» 3. Нажать кнопку «Добавить» 4. Выбрать валидные дату и время 5. Нажать кнопку «Создать» 	В расписании в ячейке с указанной датой появится свободное занятие.	Соответствует ожидаемому (рисунок Ж.1 – Ж.3)
Запись на занятие	<ol style="list-style-type: none"> 1. Авторизоваться как студент 2. Открыть страницу «Расписание» 3. Нажать на ячейку расписания со свободным занятием 4. Нажать кнопку «Записаться» 5. В появившемся модальном окне нажать кнопку «Подтвердить» 6. Перейти на страницу «Мои записи» 	На странице «Мои записи» отображается карточка с забронированным занятием	Соответствует ожидаемому (рисунок Ж.4 – Ж.6)

Продолжение таблицы 20

Название	Шаги	Ожидаемый результат	Фактический результат
Отмена записи на занятие студентом	<ol style="list-style-type: none"> 1. Авторизоваться как студент 2. Перейти на страницу «Мои записи» 3. Нажать на кнопку «Отменить» в карточке с забронированным занятием 4. В появившемся модальном окне нажать кнопку «Подтвердить» 5. Перейти на страницу «Расписание» 	На странице «Мои записи» карточка с забронированным занятием пропадает. В расписании в ячейке с датой отображается свободное занятие.	Соответствует ожидаемому (рисунок Ж.7, Ж.8)
Отмена занятия	<ol style="list-style-type: none"> 1. Авторизоваться как инструктор 2. Перейти на страницу «Мои записи» 3. Нажать на кнопку «Отменить» в карточке с занятием 4. В появившемся модальном окне нажать кнопку «Подтвердить» 5. Перейти на страницу «Расписание» 	На странице «Мои записи» пропадает карточка с занятием. В расписании в ячейке с датой не отображается свободное занятие	Соответствует ожидаемому (рисунок Ж.9, Ж.10)

Для описания состояний профиля пользователя была разработана диаграмма состояний и переходов, представленная на рисунке 63.

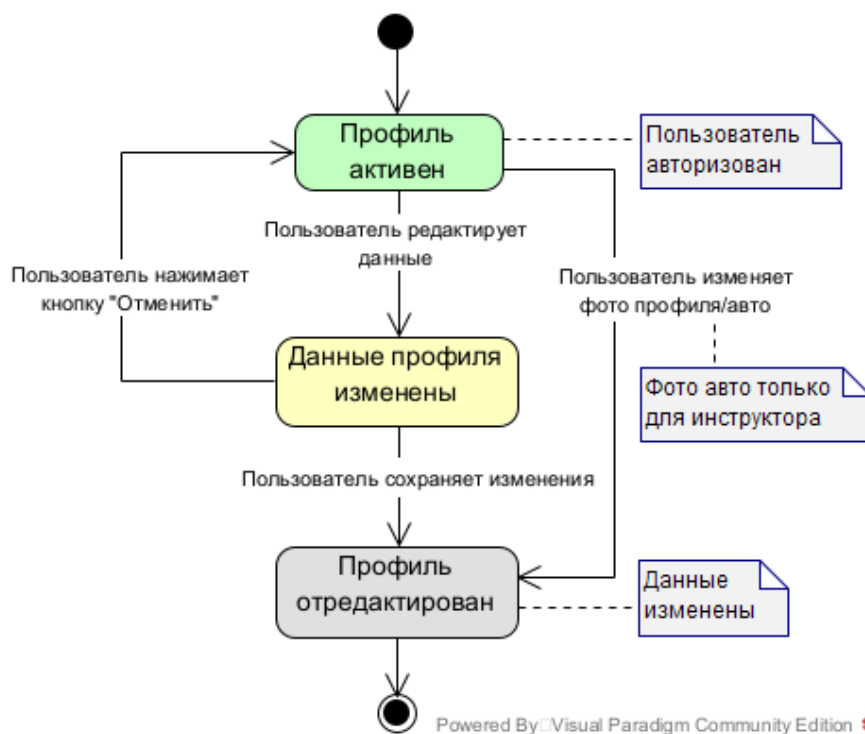


Рисунок 63 – Диаграмма состояний и переходов профиля

На основании диаграммы состояний и переходов профиля пользователя были написаны тестовые сценарии, представленные в таблице 21.

Таблица 21 – Тест-кейсы функционала профиля пользователя

Название	Шаги	Ожидаемый результат	Фактический результат
Переход в режим редактирования	<ol style="list-style-type: none"> 1. Авторизоваться 2. Открыть страницу «Профиль» 3. Нажать на кнопку «Редактировать» 	Поля с данными становятся доступными для редактирования	Соответствует ожидаемому (рисунок Ж.11, Ж.12)
Редактирование данных профиля	<ol style="list-style-type: none"> 1. Открыть страницу «Профиль» 2. Нажать на кнопку «Редактировать» 3. Отредактировать значения полей 4. Нажать на кнопку «Сохранить» 	Данные профиля изменены	Соответствует ожидаемому (рисунок Ж.13, Ж.14)
Пользователь нажимает кнопку «Отмена» при редактировании профиля	<ol style="list-style-type: none"> 1. Открыть страницу «Профиль» 2. Нажать на кнопку «Редактировать» 3. Отредактировать значения полей 4. Нажать на кнопку «Отмена» 	Выход из режима редактирования (данные не изменены)	Соответствует ожидаемому (рисунок Ж.15, Ж.16)
Изменить фото автомобиля	<ol style="list-style-type: none"> 1. Авторизоваться как инструктор 2. Нажать на кнопку редактирования фото (иконка карандаша) 3. Выбрать файл на устройстве (двойной клик по выбранному изображению) 	Фото автомобиля изменено	Соответствует ожидаемому (рисунок Ж.17 – Ж.19)

Для описания функционала управления аккаунтами была создана диаграмма состояний и переходов аккаунта, представленная на рисунке 64. Диаграмма иллюстрирует жизненный цикл учетной записи в системе, включая состояния «Не создан», «Активен», «Удален» и переходы между ними.

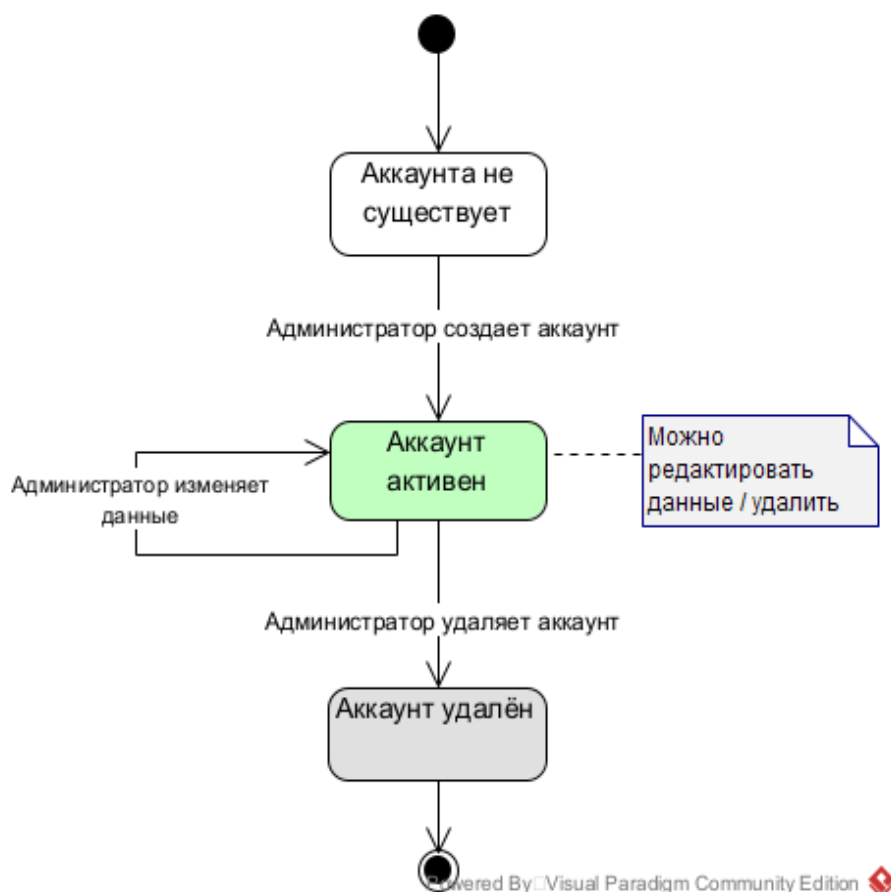


Рисунок 64 – Диаграмма состояний и переходов аккаунта

На основании диаграммы состояний и переходов аккаунта были написаны тестовые сценарии, представленные в таблице 22.

Таблица 22 – Тест-кейсы функционала аккаунтов

Название	Шаги	Ожидаемый результат	Фактический результат
Создание аккаунта	<ol style="list-style-type: none"> 1. Авторизоваться как администратор 2. Открыть страницу «Аккаунты» 3. Нажать кнопку «Добавить» 4. В открывшемся модальном окне выбрать тип создаваемого аккаунта 5. Заполнить данными обязательные поля 6. Нажать кнопку «Создать» 	В таблице с аккаунтами появится новая запись с введенными данными.	Соответствует ожидаемому (рисунок Ж.20 – Ж.22)

Продолжение таблицы 22

Название	Шаги	Ожидаемый результат	Фактический результат
Редактирование аккаунта	<ol style="list-style-type: none"> 1. Авторизоваться как администратор 2. Открыть страницу «Аккаунты» 3. Нажать на кнопку редактирования (иконка карандаша) в колонке «Действия» 4. В открывшемся модальном окне отредактировать значение в любом поле 5. Нажать кнопку «Сохранить» 	В таблице с аккаунтами учетные данные выбранной записи изменятся	Соответствует ожидаемому (рисунок Ж.23 – Ж.25)
Удаление аккаунта	<ol style="list-style-type: none"> 1. Авторизоваться как администратор 2. Перейти на страницу «Аккаунты» 3. Нажать на кнопку удаления (корзина) в колонке «Действия» 4. В появившемся модальном окне нажать кнопку «Удалить» 	Выбранная запись удалится из таблицы	Соответствует ожидаемому (рисунок Ж. 26 – Ж.28)

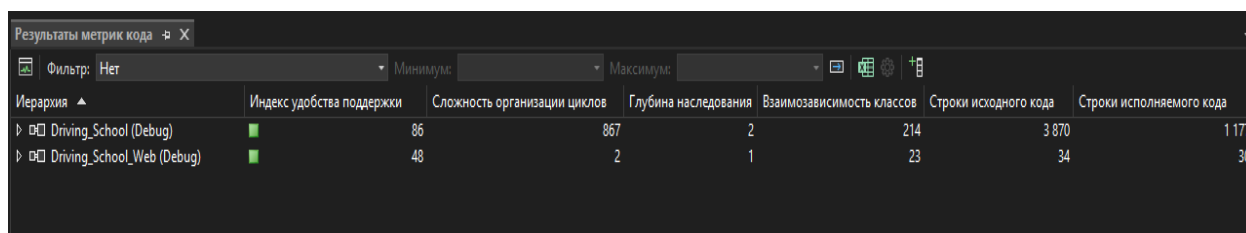
Таким образом было проведено функциональное тестирование клиентской части информационной системы управления учебным процессом автошколы. Тестирование показало, что система работает стабильно, ее, поведение соответствует заявленным функциональным требованиям.

4.4 Метрики кода

Метрики кода – это показатели, применяемые для оценки качества программного обеспечения. Они отражают такие характеристики, как читаемость, сложность, модульность и удобство сопровождения. Использование метрик помогает выявлять потенциальные проблемы на ранних этапах и упростить поддержку и развитие проекта [21].

На рисунке 65 представлены результаты метрик кода с использованием встроенных средств Microsoft Visual Studio, с использованием следующих критериев:

- индекс удобства поддержки – показывает, насколько код легко понимать и модифицировать;
- сложность организации циклов – отражает количество логических ветвлений в коде;
- глубина наследования – показывает количество уровней иерархии классов, где высокие значения могут усложнять понимание структуры;
- взаимозависимость классов – определяет степень зависимости между классами;
- строки исходного кода – обозначает общее количество строк в файлах, включая комментарии и пустые строки;
- строки исполняемого кода – указывает на количество строк, содержащих исполняемую бизнес-логику [22].



Иерархия	Индекс удобства поддержки	Сложность организации циклов	Глубина наследования	Взаимозависимость классов	Строки исходного кода	Строки исполняемого кода
Driving_School (Debug)	86	867	2	214	3 870	1 177
Driving_School_Web (Debug)	48	2	1	23	34	30

Рисунок 65 – Результаты метрик кода

На основании полученных метрик можно сделать выводы, что индекс удобства поддержки находится на высоком уровне как в клиентской, так и в серверной частях, что говорит о хорошем качестве архитектуры и соблюдении принципов модульности. Цикломатическая сложность у серверной части значительно выше, что связано с реализацией сложной логики бизнес-процессов и множеством проверок в API-контроллерах и сервисах.

Прочие показатели, такие как глубина наследования и связность классов, находятся в допустимых пределах, что свидетельствует об умеренной сложности архитектуры и поддерживаемости проекта.

Таким образом, по результатам анализа можно утверждать, что код системы пригоден для поддержки и расширения.

Заключение

В рамках выпускной квалификационной работы спроектирована и разработана информационная система управления учебным процессом автошколы.

На этапе анализа требований выполнено изучение предметной области, проведен сравнительный анализ аналогичных решений, выявлены основные бизнес-процессы, а также определены функциональные и нефункциональные требования к системе.

Для реализации была выбрана многослойная архитектура с модульным подходом, что обеспечивает масштабируемость и независимость модулей. Каждый модуль реализует контроллер, сервис и репозиторий.

Проектирование базы данных выполнено на основе концептуального моделирования, с дальнейшим построением логической и физической модели. Также разработаны основные UML-диаграммы, отражающие структуру и взаимодействие компонентов системы: диаграммы вариантов использования, классов, компонентов и развертывания.

Выполнено планирование и оценка стоимости разработки системы. Общая трудоемкость разработки составила 37 рабочих дней, а оценочная стоимость – 150600 рублей.

Разработка серверной части выполнена на языке C# с использованием ASP.NET Core в среде Visual Studio 2022. Клиентская часть реализована с применением HTML, CSS и JavaScript. Для взаимодействия между компонентами системы использовался REST API, обеспечивающий надежную передачу данных между клиентом и сервером.

Проведена оценка качества кода с использованием встроенных инструментов Visual Studio. Полученные метрики показали высокий уровень удобства поддержки и умеренную архитектурную сложность, за исключением цикломатической сложности в серверной части, что объясняется наличием большого количества условий в бизнес-логике.

Таким образом, поставленные цели выпускной квалификационной работы достигнуты. Разработанная информационная система обеспечивает автоматизацию ключевых процессов автошколы, обладает удобным пользовательским интерфейсом, достаточно гибкой архитектурой и отвечает требованиям к качеству. Система может быть использована в реальной практике и обладает потенциалом для дальнейшего расширения и доработки.

Список использованных источников

1. Роль информационных технологий в современном мире // Дзен [Электронный ресурс]. URL: https://dzen.ru/a/ZW80vJBMV1_JvMag (дата обращения: 15.05.2025)
2. Автоматизация бизнес-процессов в автошколах: решения компании ООО МКСИТИ // METRONEWS [Электронный ресурс]. URL: <https://www.metronews.ru/partners/novosti-partnerov-255/reviews/avtomatizaciya-biznes-processov-v-avtoshkolah-2133026/> (дата обращения: 15.05.2025)
3. Использование диаграммы вариантов использования UML при проектировании программного обеспечения // Хабр [Электронный ресурс]. URL: <https://habr.com/ru/articles/566218/> (дата обращения: 15.05.2025)
4. Что такое нефункциональные требования: примеры, определение, полное руководство // VISURESOLUTIONS [Электронный ресурс]. URL: <https://visuresolutions.com/ru/Блог/нефункциональные-требования/> (дата обращения: 16.05.2025)
5. Когда использовать базу данных MS SQL Server? // Группа компаний ДБ-сервис [Электронный ресурс]. URL: <https://www.dbserve.ru/blog/kogda-ispolzovat-bazu-dannyh-ms-sql-server> (дата обращения: 16.05.2025)
6. ASP.NET Core // WORKSPACE [Электронный ресурс]. URL: <https://workspace.ru/tools/framework/asp-dot-net-core/> (дата обращения: 16.05.2025)
7. Инструменты моделирования бизнес-процессов. Visual Paradigm // Генератор продаж [Электронный ресурс]. URL: <https://sales-generator.ru/blog/instrumenty-dlya-biznes-protsessov/#4> (дата обращения: 16.05.2025)
8. Клиент-серверная архитектура // SERVERGATE [Электронный ресурс]. URL: <https://servergate.ru/articles/klient-servernaya-arkhitektura/> (дата обращения: 16.05.2025)

9. Концептуальное моделирование (Conceptual Modeling): основы, методы и применение // Я зерокодер [Электронный ресурс]. URL: <https://ya.zerocoder.ru/pgt-konceptualnoe-modelirovanie-conceptual-modeling-osnovy-metody-i-primenenie/> (дата обращения: 16.05.2025)
10. Моделирование и проектирование данных // Проект DAMA DMBOK2: конспекты по управлению данными [Электронный ресурс]. URL: <https://dataliteracy.ru/dmbok-notes-5> (дата обращения: 17.05.2025)
11. Слоистая архитектура // Школа системного анализа и проектирования [Электронный ресурс]. URL: <https://systems.education/layered-architecture> (дата обращения: 17.05.2025)
12. Основы кроссбраузерной верстки // Skyprowiki [Электронный ресурс]. URL: <https://sky.pro/wiki/html/osnovy-krossbrauzernej-verstki/> (дата обращения: 25.05.2025)
13. Физическая модель данных // APPMASTER [Электронный ресурс]. URL: <https://appmaster.io/ru/glossary/fizicheskaja-model-dannykh> (дата обращения: 25.05.2025)
14. Репозиторий кода Driving_School: серверная и клиентская части // GITHUB [Электронный ресурс]. URL: https://github.com/zvezdniy10rd/Driving_School (дата обращения: 08.06.2025)
15. Архитектурный слой // Struchkov's Digital Garden [Электронный ресурс]. URL: <https://garden.struchkov.dev/ru/dev/architecture/Архитектурный-слой> (дата обращения: 25.05.2025)
16. UML: обзор основных типов диаграмм, диаграмма компонентов. Часть 2. // Хабр [Электронный ресурс]. URL: <https://habr.com/ru/articles/756552/> (дата обращения: 25.05.2025)
17. Тестирование и 7 основных этапов его проведения // NEIROS [Электронный ресурс]. URL: <https://neiros.ru/blog/code/testirovanie-i-7-osnovnykh-etapov-ego-provedeniya/> (дата обращения: 05.06.2025)
18. Типы тестирования программного обеспечения: различия и примеры // LOADVIEW [Электронный ресурс]. URL: <https://www.loadview->

testing.com/ru/blog/типы-тестирования-программного-обес/ (дата обращения: 05.06.2025)

19. Unit-тестирование // SKILLFACTORY MEDIA [Электронный ресурс]. URL: <https://blog.skillfactory.ru/glossary/unit-testirovanie/> (дата обращения: 05.06.2025)

20. Тестирование состояний и переходов (State Transition Testing) // TESTENGINEER [Электронный ресурс]. URL: <https://testengineer.ru/state-transition-testing/> (дата обращения: 05.06.2025)

21. Метрики в программировании: что это и зачем они нужны? // POLYGANT [Электронный ресурс]. URL: <https://polygant.net/ru/blog/metriki-v-programmirovanii-cto-eto-i-zachem-oni-nuzhny/> (дата обращения: 07.06.2025)

22. Роль качества кода: критические метрики и их влияние // IN-COM [Электронный ресурс]. URL: <https://www.in-com.com/ru/blog/the-role-of-code-quality-critical-metrics-and-their-impact/> (дата обращения: 07.06.2025)

Приложение А
(справочное)
Глоссарий

База данных (БД) – это организованная коллекция данных, которая позволяет удобно хранить, управлять и извлекать информацию.

Веб-браузер – прикладное программное обеспечение для поиска и просмотра веб-страниц, а также управления веб-приложениями.

Веб-приложение – это клиент-серверное программное решение, доступ к которому осуществляется через сетевой браузер посредством передачи данных по протоколу HTTP, размещается на удаленном сервере.

Информационная система (ИС) – это совокупность программного обеспечения, оборудования, данных, предназначенных для сбора, хранения, обработки и предоставления информации.

СУБД (Система управления базами данных) – это совокупность программных и лингвистических средств общего или специального назначения, обеспечивающих управление созданием и использованием баз данных.

Фреймворк – программная платформа, задающая архитектурные принципы и общие правила разработки приложений в определенной предметной области.

API (Application Programming Interface) – набор готовых классов, функций, процедур, констант и структур, предоставляемых приложением или операционной системой для использования во внешних программных продуктах.

JSON – JavaScript Object Notation, легковесный текстовый формат обмена данными, полностью независимый от языка реализации.

Microsoft SQL Server Management Studio – это интегрированная среда разработки и администрирования, предназначенная для работы с Microsoft SQL Server. Она предоставляет разработчикам и администраторам возможность управления базами данных, написания запросов, создания и редактирования объектов базы данных.

REST (Representational State Transfer) – это архитектурная концепция, определяющая набор ограничений для построения масштабируемых распределенных систем. Основными принципами REST является ресурсно-

ориентированная архитектура, использование стандартных HTTP-методов для манипуляции ресурсами, передача данных без сохранения состояния на сервере, унифицированный интерфейс доступа.

.NET – это платформа для разработчиков, состоящая из инструментов, языков программирования и библиотек для создания множества различных типов приложений. ASP.NET Core расширяет платформу разработчика .NET инструментами и библиотеками, специально предназначенными для создания веб-приложений.

Приложение Б
(обязательное)
Листинг серверной части системы

Листинг Б.1 – Schedule.cs

```
namespace Driving_School_API.Models.Schedule;

public class Schedule {
    public int Id { get; set; }
    public DateOnly Date { get; set; }
    public TimeOnly SlotTime { get; set; }
    public bool IsAvailable { get; set; }
    public int Instructor_ID { get; set; }
    public int? Student_ID { get; set; }
    public DateTime CreatedAt { get; set; }
}
```

Листинг Б.2 – ScheduleDTO.cs

```
public class ScheduleDto{
    [Required(ErrorMessage = "Дата не указана")]
    [DataType(DataType.Date, ErrorMessage = "Некорректный формат даты")]
    public DateOnly Date { get; set; }

    [Required(ErrorMessage = "Время не указано")]
    [DataType(DataType.Time, ErrorMessage = "Некорректный формат времени")]
    [CustomValidation(typeof(ScheduleDto),
        nameof(ValidateSlotTime))]
    [DefaultValue("08:00")]
    public TimeOnly SlotTime { get; set; }

    [Required(ErrorMessage = "Доступность слота не указана")]
    [DefaultValue(true)]
    public bool IsAvailable { get; set; }

    [Range(1, int.MaxValue, ErrorMessage = "ID инструктора не может быть < 1")]
    [Required(ErrorMessage = "ID инструктора не указан")]
    [DefaultValue(0)]
    public int Instructor_ID { get; set; }

    [DefaultValue("null")]
    public int? Student_ID { get; set; }

    public DateTime CreatedAt { get; private set; }

    // Метод для проверки корректности времени (время должно быть
    в интервале 08:00–20:00)
    public static ValidationResult? ValidateSlotTime(TimeOnly
        slotTime, ValidationContext context) {
        var start = new TimeOnly(8, 0); // 08:00
        var end = new TimeOnly(20, 0); // 20:00
```

```
        if (slotTime < start || slotTime > end) { return new
ValidationResult("Время должно быть в промежутке с 08:00 до
20:00"); }
```

```
        return ValidationResult.Success;
    }
}
```

```
public class ScheduleEditDto {
    [Required(ErrorMessage = "Доступность слота не указана")]
    [DefaultValue(true)]
    public bool IsAvailable { get; set; }
    [DefaultValue("null")]
    public int? Student_ID { get; set; }
}
```

Листинг Б.3 – ScheduleController.cs

```
[ApiController]
[Route("/schedules")]
public class SchedulesController : ControllerBase{
    private readonly IScheduleService _scheduleService;

    public SchedulesController(IScheduleService scheduleService) {
        _scheduleService = scheduleService; }

    [HttpGet]
    public async Task<IActionResult> GetAllSchedules([FromQuery]
int? month = null, [FromQuery] int? year = null, [FromQuery] int?
day = null){
        try {
            var schedules = await
_scheduleService.GetAllSchedulesAsync();

            // Фильтрация по месяцу и году, если параметры указаны
            if (month.HasValue && year.HasValue) {
                schedules = schedules.Where(s => s.Date.Month ==
month.Value && s.Date.Year == year.Value).ToList();

                // Дополнительная фильтрация по дню, если параметр
указан
                if (day.HasValue) { schedules = schedules.Where(s
=> s.Date.Day == day.Value).ToList(); }

                return Ok(schedules);
            }
            catch (Exception ex) {
                return StatusCode(500, new { Message = "Произошла
ошибка на сервере", Details = ex.Message });
            }
        }
    }
}
```

// Фрагмент

Листинг Б.4 – ScheduleService.cs

```
using Driving_School_API.Models.Schedule;

public class ScheduleService : IScheduleService {
    private readonly IScheduleRepository _scheduleRepository;

    public ScheduleService(IScheduleRepository scheduleRepository)
    {
        _scheduleRepository = scheduleRepository; }

    // получение данных всех расписаний
    public async Task<IEnumerable<Schedule>>
    GetAllSchedulesAsync() { return await
    _scheduleRepository.GetAllSchedulesAsync(); }

    // получить данные расписания по id
    public async Task<Schedule> GetScheduleByIdAsync(int id) {
    return await _scheduleRepository.GetScheduleByIdAsync(id); }
    // Фрагмент кода
```

Листинг Б.5 – ScheduleRepository.cs

```
using Driving_School_API.Models.Schedule;
using Microsoft.EntityFrameworkCore;

public class ScheduleRepository : IScheduleRepository {
    private readonly ApplicationDbContext _context;

    public ScheduleRepository(ApplicationDbContext context) {
        _context = context; }

    // получение данных всех расписаний
    public async Task<IEnumerable<Schedule>>
    GetAllSchedulesAsync() { return await
    _context.Schedule.ToListAsync(); }

    // получение данных расписания по id
    public async Task<Schedule> GetScheduleByIdAsync(int id) {
    return await _context.Schedule.FindAsync(id); }

    // получение занятий по id инструктора
    public async Task<List<Schedule>>
    GetScheduleByInstructorIdAsync(int instructor_id) { return await
    _context.Schedule.Where(s => s.Instructor_ID ==
    instructor_id).ToListAsync(); ; }

    // получение занятий по id студента
    public async Task<List<Schedule>>
    GetScheduleByStudentIdAsync(int student_id) { return await

    // Фрагмент
```

Листинг Б.6 – ApplicationDbContext.cs

```
using Microsoft.EntityFrameworkCore;
using Driving_School_API.Models.Instructor;
using Driving_School_API.Models.Student;
using Driving_School_API.Models.Review;
using Driving_School_API.Models.Car;
using Driving_School_API.Models.Schedule;
using Driving_School_API.Models;
using Driving_School_API.Models.Driving_School;
using Driving_School_API.Models.City;
using Driving_School_API.Models.Admin;

public class ApplicationDbContext : DbContext {
    public
    ApplicationDbContext(DbContextOptions<ApplicationDbContext>
options) : base(options) { }

    public DbSet<Student> Student { get; set; }
    public DbSet<Instructor> Instructor { get; set; }
    public DbSet<Review> Review { get; set; }
    public DbSet<Car> Car { get; set; }
    public DbSet<Schedule> Schedule { get; set; }
    public DbSet<Account> Account { get; set; }
    public DbSet<Driving_School> Driving_School { get; set; }
    public DbSet<City> City { get; set; }
    public DbSet<Admin> Admin { get; set; }
    public DbSet<Attachment> Attachment { get; set; }

    protected override void OnModelCreating(ModelBuilder
modelBuilder) {
    // Фрагмент
```

Листинг Б.7 – StudentController.cs

```
[ApiController]
[Route("/students")]
public class StudentsController : ControllerBase {
    private readonly IStudentService _studentService;

    public StudentsController(IStudentService studentService) {
        _studentService = studentService;
    }

    [HttpGet]
    public async Task<IActionResult> GetAllStudents() {
        try { return Ok(await
        _studentService.GetAllStudentsAsync()); }
        catch (Exception ex) { return StatusCode(500, new {
        Message = "Произошла ошибка на сервере", Details = ex.Message });
        }}
    // Фрагмент кода
```

Листинг Б.8 – StudentService.cs

```

public class StudentService : IStudentService
{
    private readonly IStudentRepository _studentRepository;

    public StudentService(IStudentRepository studentRepository)
    {
        _studentRepository = studentRepository;
    }

    // получение данных всех студентов
    public async Task<IEnumerable<Student>> GetAllStudentsAsync()
    {
        return await _studentRepository.GetAllStudentsAsync();
    }

    // получить данные студента по id
    public async Task<Student> GetStudentByIdAsync(int id)
    {
        return await _studentRepository.GetStudentByIdAsync(id);
    }

    // получение отфильтрованных студентов с пагинацией
    public async Task<(IEnumerable<Student> Data, int TotalCount,
int TotalPages)> GetFilteredStudentsAsync(StudentFilterDto filter)
    {
        var (data, totalCount) = await
        _studentRepository.GetFilteredStudentsAsync(filter);
        var totalPages = (int)Math.Ceiling((double)totalCount /
filter.PageSize);
        return (data, totalCount, totalPages);
    }
}
// Фрагмент

```

Листинг Б.9 – StudentRepository.cs

```

public class StudentRepository : IStudentRepository {
    private readonly ApplicationDbContext _context;

    public StudentRepository(ApplicationDbContext context) {
        _context = context; }

    // получение данных всех студентов
    public async Task<IEnumerable<Student>> GetAllStudentsAsync()
    { return await _context.Student.ToListAsync(); }

    // получение данных студента по id
    public async Task<Student> GetStudentByIdAsync(int id) {
return await _context.Student.FindAsync(id); }

    // получение отфильтрованных студентов

```

```

        public async Task<IEnumerable<Student> Data, int TotalCount>
        GetFilteredStudentsAsync(StudentFilterDto filter) {
            var query = _context.Student.AsQueryable();

            // Фильтрация по фамилии студента
            if (!string.IsNullOrEmpty(filter.Surname)) { query =
            query.Where(s =>
            s.Surname.ToLower().Contains(filter.Surname.ToLower())); }

            // Фильтрация по имени студента
            if (!string.IsNullOrEmpty(filter.Name)) { query =
            query.Where(s =>
            s.Name.ToLower().Contains(filter.Name.ToLower())); }

            // Фильтрация по отчеству студента
            if (!string.IsNullOrEmpty(filter.Patronymic)) { query =
            query.Where(s =>
            s.Patronymic.ToLower().Contains(filter.Patronymic.ToLower())); }

            // Фильтрация по номеру телефона студента
            if (!string.IsNullOrEmpty(filter.PhoneNumber)) { query =
            query.Where(s => s.PhoneNumber.Contains(filter.PhoneNumber)); }

            // Фильтрация по эл. почте студента
            if (!string.IsNullOrEmpty(filter.Email)) { query =
            query.Where(s => s.Email.Contains(filter.Email)); }

            // Фильтрация по автошколе
            if (filter.DrivingSchool_ID.HasValue) { query =
            query.Where(s => s.DrivingSchool_ID == filter.DrivingSchool_ID); }

            // Фильтрация по городу
            if (filter.City_ID.HasValue) {
                query = query.Where(s => (!filter.City_ID.HasValue ||
            _context.Driving_School
                                .Where(ds => ds.Id ==
            s.DrivingSchool_ID)
                                .Any(ds => ds.City_ID ==
            filter.City_ID.Value)));
            }
        }
    }
    // Фрагмент

```

Листинг Б.10 – Auth.cs

```

namespace Driving_School_API.Models.Car;

public class Car {
    public int Id { get; set; }
    public string Brand { get; set; }
    public string Model { get; set; }
    public string Color { get; set; }
    public string Car_Number { get; set; }
    public int? Attachment_ID { get; set; }
}

```

```

using System.ComponentModel.DataAnnotations;
namespace Driving_School_API.Models;

public class Account {
    public int Id { get; set; }
    public int User_Type { get; set; }
    public string Password { get; set; }
    public string Login { get; set; }
    public int? Student_ID { get; set; }
    public int? Instructor_ID { get; set; }
    public int? Admin_ID { get; set; }
}

public class LoginDto {
    [Required(ErrorMessage = "Логин не указан")]
    public string Login { get; set; }
    [Required(ErrorMessage = "Пароль не указан")]
    public string Password { get; set; }
}

public class RegisterInstructorDto {
    public string Surname { get; set; }
    public string Name { get; set; }
    public string? Patronymic { get; set; }
    public string PhoneNumber { get; set; }
    public string Email { get; set; }
    public int DrivingSchool_ID { get; set; }
    public string CarBrand { get; set; }
    public string CarModel { get; set; }
    public string CarColor { get; set; }
    public string CarNumber { get; set; }
    public string Password { get; set; }
}
// Фрагмент кода

```

Листинг Б.11 – AuthController.cs

```

[Route("")]
[ApiController]
public class AuthController : ControllerBase {
    private readonly IAuthService _authService;
    public AuthController(IAuthService authService) { _authService
= authService; }

    [HttpGet("/accounts")]
    public async Task<IActionResult> GetAllAccounts([FromQuery]
int page = 1, [FromQuery] int pageSize = 10) {
        try {
            var accounts = await
_authService.GetAllAccountsAsync();
            var totalCount = accounts.Count();
            var paginatedAccounts = accounts
                .Skip((page - 1) * pageSize)
                .Take(pageSize)

```

```

        .ToList();

        return Ok(new {
            TotalCount = totalCount,
            Page = page,
            PageSize = pageSize,
            TotalPages = (int)Math.Ceiling((double)totalCount
/ pageSize),
            Data = paginatedAccounts
        });
    }
    catch (Exception ex) { return StatusCode(500, new {
Message = "Произошла ошибка на сервере", Details = ex.Message });
}
// Фрагмент

```

Листинг Б.12 – AuthService.cs

```

public class AuthService : IAuthService {
    private readonly IAuthRepository _accountRepository;
    private readonly IInstructorRepository _instructorRepository;
    private readonly IStudentRepository _studentRepository;
    private readonly ICarRepository _carRepository;
    private readonly IAdminRepository _adminRepository;

    public AuthService(IAuthRepository accountRepository,
IInstructorRepository instructorRepository,
        IStudentRepository studentRepository,
ICarRepository carRepository, IAdminRepository adminRepository) {
        _accountRepository = accountRepository;
        _instructorRepository = instructorRepository;
        _studentRepository = studentRepository;
        _carRepository = carRepository;
        _adminRepository = adminRepository;
    }

    // получение данных всех аккаунтов
    public async Task<IEnumerable<Account>> GetAllAccountsAsync()
{ return await _accountRepository.GetAllAccountsAsync(); }

    public async Task<(bool, string, Account)> LoginAsync(string
login, string password) {
        var account = await
_accountRepository.GetAccountByLoginAsync(login, password);
        if (account == null || account.Password != password) {
return (false, "Неверный логин или пароль!", null); }
        return (true, "Успешная авторизация", account);
    }
}
// Фрагмент кода

```

Листинг Б.13 – AuthRepository.cs

```

public class AuthRepository : IAuthRepository {

```

```

        private readonly ApplicationDbContext _context;
        public AuthRepository(ApplicationDbContext context) { _context
= context; }

        // получение данных всех аккаунтов
        public async Task<IEnumerable<Account>> GetAllAccountsAsync()
{ return await _context.Account.ToListAsync(); }

        // получение данных аккаунта по id
        public async Task<Account> GetAccountByIdAsync(int id) {
return await _context.Account.FindAsync(id); }

        // получение отфильтрованных аккаунтов
        public async Task<(IEnumerable<Account> Data, int TotalCount)>
GetFilteredAccountsAsync(AccountFilterDto filter) {
            var query = _context.Account
                .AsQueryable();

            // Фильтрация по логину
            if (!string.IsNullOrEmpty(filter.Login)) { query =
query.Where(a =>
a.Login.ToLower().Contains(filter.Login.ToLower())); }

            // Фильтрация по типу пользователя
            if (filter.UserType.HasValue) { query = query.Where(a =>
a.User_Type == filter.UserType.Value); }

            // Фильтрация по ФИО
            if (!string.IsNullOrEmpty(filter.FullName)) {
                var fullNameLower = filter.FullName.ToLower();
                query = query.Where(a =>
                    (a.Student_ID.HasValue && _context.Student
                        .Where(s => s.Id == a.Student_ID)
                        .Any(s => (s.Surname + " " + s.Name + " " +
s.Patronymic).ToLower().Contains(fullNameLower))) ||
                    (a.Instructor_ID.HasValue && _context.Instructor
                        .Where(i => i.Id == a.Instructor_ID)
                        .Any(i => (i.Surname + " " + i.Name + " " +
i.Patronymic).ToLower().Contains(fullNameLower)))
                );
            }
        }
    }
// Фрагмент кода

```

Приложение В
(обязательное)
Листинг клиентской части системы

Листинг В.1 – Auth.js

```

let isRedirecting = false;

document.addEventListener('DOMContentLoaded', async () => {
    if (isRedirecting) { return; }

    const currentPath = window.location.pathname;
    console.log('auth.js: Текущая страница:', currentPath);

    if (currentPath.endsWith('auth.html')) {
        const loginBtn = document.getElementById('login-btn');
        const loginInput = document.getElementById('login-input');
        const passwordInput = document.getElementById('password-
input');
        const errorMessage = document.getElementById('error-
message');

        // Проверяем, авторизован ли пользователь
        const user = JSON.parse(localStorage.getItem('user'));
        if (user && isAuthenticated()) {
            isRedirecting = true;
            redirectByRole(user);
            return;
        }

        // Обработка авторизации
        loginBtn.onclick = async () => {
            const login = loginInput.value.trim();
            const password = passwordInput.value.trim();
            if (!login || !password) {
                errorMessage.textContent = 'Пожалуйста, заполните
все поля';

                errorMessage.style.display = 'block';
                return;
            }
            try {
                const response = await
fetch('https://localhost:7174/login', {
                    method: 'POST',
                    headers: {
                        'Accept': '*/*',
                        'Content-Type': 'application/json'
                    },
                    body: JSON.stringify({
                        login: login,
                        password: password
                    })
                })
            } catch {
                // Фрагмент кода
            }
        }
    }
}

```

Листинг В.2 – Roles.js

```

const ROLES = { STUDENT: 2, INSTRUCTOR: 1, ADMIN: 3 };

```

```

// Получение роли пользователя из localStorage
function getUserRole() {
    const user = JSON.parse(localStorage.getItem('user'));
    return user ? user.user_Type : null;
}

// Проверка, аутентифицирован ли пользователь
function isAuthenticated() { return getUserRole() !== null; }

// Ограничение доступа к странице или функции на основе роли
function restrictAccess(roleRequired, redirectUrl = '/main.html')
{
    const userRole = getUserRole();
    const isRoleAllowed = roleRequired === null ?
isAuthenticated() :
    Array.isArray(roleRequired) ?
roleRequired.includes(userRole) : userRole === roleRequired;
    if (!isAuthenticated() || !isRoleAllowed) { return false; }
    return true;
}

// Инициализация интерфейса на основе роли пользователя
function initializeRoleBasedUI() {
    const userRole = getUserRole();
    if (!userRole) { return; }

    const navLinks = document.querySelectorAll('nav ul li a');
    const buttons = document.querySelectorAll('nav ul li button');

    // Разрешения для ролей
    const rolePermissions = {
        [ROLES.STUDENT]: ['/schedule.html', '/instructors.html',
'/student-schedules.html', '/profile.html'],
        [ROLES.INSTRUCTOR]: ['/schedule.html', '/students.html',
'/instructor-schedules.html', '/driving-schools.html',
'/profile.html'],
        [ROLES.ADMIN]: ['/students.html', '/instructors.html',
'/cars.html', '/driving-schools.html', '/accounts.html',
'/profile.html']
    };
}
// Фрагмент кода

```

Листинг В.3 – Schedule.html

```

<!DOCTYPE html>
<html lang="ru">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-
scale=1.0">
    <title>ИС автошколы</title>
    <link rel="stylesheet" href="/css/styles.css">

```

```

    <link rel="stylesheet" href="/css/schedule.css">
</head>
<body>
    <header>
        <h2>Информационная система управления учебным процессом
автошколы</h2>
        <nav id="nav-menu">
            <ul>
                <li><a href="/schedule.html">Расписание</a></li>
                <li><a href="/students.html">Студенты</a></li>
                <li><a
href="/instructors.html">Инструкторы</a></li>
                <li><a href="/driving-
schools.html">Автошколы</a></li>
                <li><a href="/accounts.html">Аккаунты</a></li>
                <li><a href="/student-schedules.html">Мои
записи</a></li>
                <li><a href="/instructor-schedules.html">Мои
записи</a></li>
                <li><a href="/profile.html">Профиль</a></li>
            </ul>
        </nav>
    </header>
    <main>
        <section id="schedule">
            <div class="section-header">
                <h2>Расписание</h2>
                <button id="add-schedule">Добавить</button>
            </div>
            <div id="schedule-table"></div>
        </section>
    </main>
// Фрагмент кода

```

Листинг В.4 – Schedule.js

```

let isFetchingSchedule = false;
let displayedDate = new Date();
const currentDate = new Date(); // Текущая дата
const currentYear = currentDate.getFullYear();
const currentMonth = currentDate.getMonth();
const allowedMonths = [
    { year: currentYear, month: currentMonth },
    { year: currentMonth >= 10 ? currentYear + 1 : currentYear,
month: (currentMonth + 1) % 12 },
    { year: currentMonth >= 9 ? currentYear + 1 : currentYear,
month: (currentMonth + 2) % 12 }
];

document.addEventListener('DOMContentLoaded', async () => {
    console.log('schedules.js: Событие DOMContentLoaded
сработало');
    if (!isAuthenticated()) {

```

```

        console.log('schedules.js: Пользователь не авторизован,
перенаправление на /auth.html');
        window.location.href = '/auth.html';
        return;
    }
    if (!restrictAccess(null)) {
        console.log('schedules.js: Доступ ограничен');
        return;
    }

    const userRole = getUserRole();
    console.log('schedules.js: Роль пользователя:', userRole);
    if (userRole !== ROLES.INSTRUCTOR) {
        const addScheduleBtn = document.getElementById('add-
schedule');
        const addScheduleModal = document.getElementById('add-
schedule-modal');
        if (addScheduleBtn) addScheduleBtn.style.display = 'none';
        if (addScheduleModal) addScheduleModal.style.display =
'none';
    }

    if (!isFetchingSchedule) {
        isFetchingSchedule = true;
        await fetchSchedule();
        setupAddScheduleModal();
    }
});

async function fetchSchedule() {
    const scheduleTable = document.getElementById('schedule-
table');
    if (!scheduleTable) {
        console.error('schedules.js: Элемент с id "schedule-table"
не найден');
        return;
    }
    scheduleTable.innerHTML = '';

    const headerContainer = document.createElement('div');
    headerContainer.classList.add('calendar-header');

    const prevButton = document.createElement('button');
    prevButton.textContent = '◀';
    prevButton.classList.add('nav-button');
    headerContainer.appendChild(prevButton);
}
// Фрагмент кода

```

Приложение Г
(обязательное)
Листинг модульных тестов

Листинг Г.1 – ScheduleControllerTests.cs

```

namespace DrivingSchoolApi.Tests.Controller{
    public class ScheduleControllerTests{
        private readonly Mock<IScheduleService> _mockService;
        private readonly SchedulesController _controller;
        private readonly ITestOutputHelper _output;

        private readonly Dictionary<int, Schedule> _fakeDb =
new();
        private int ID = 1;

        public ScheduleControllerTests(ITestOutputHelper output){
            _mockService = new Mock<IScheduleService>();
            _controller = new
SchedulesController(_mockService.Object);
            _output = output;
            SetupMocks();
        }

        private void SetupMocks(){
            _mockService.Setup(s =>
s.GetScheduleByIdAsync(It.IsAny<int>()))
.ReturnsAsync((int id) =>
{
    _fakeDb.TryGetValue(id, out var schedule);
    return schedule;
});

            _mockService.Setup(s =>
s.AddScheduleAsync(It.IsAny<Schedule>()))
.Callback<Schedule>(s =>
{
    s.Id = ID++;
    s.CreatedAt = DateTime.Now;
    _fakeDb[s.Id] = s;
})
.Returns(Task.CompletedTask);

            _mockService.Setup(s =>
s.UpdateScheduleAsync(It.IsAny<Schedule>()))
.Callback<Schedule>(s =>
{
    _fakeDb[s.Id] = s;
})
.Returns(Task.CompletedTask);

            _mockService.Setup(s =>
s.DeleteScheduleAsync(It.IsAny<int>()))
.Callback<int>(id =>
{
    _fakeDb.Remove(id);
})

```

```

        .Returns(Task.CompletedTask);
    }

    private ScheduleDto CreateScheduleDto() {
        return new ScheduleDto
        {
            Date =
DateOnly.FromDateTime(DateTime.Today.AddDays(1)),
            SlotTime = new TimeOnly(9, 0),
            IsAvailable = true,
            Instructor_ID = 1,
            Student_ID = null
        };
    }

    private Schedule AddFakeSchedule(int? studentId, bool
isAvailable) {
        var schedule = new Schedule
        {
            Id = ID,
            Date =
DateOnly.FromDateTime(DateTime.Today.AddDays(3)),
            SlotTime = new TimeOnly(11, 0),
            Instructor_ID = 1,
            Student_ID = studentId,
            IsAvailable = isAvailable,
            CreatedAt = DateTime.Now
        };
        _fakeDb[ID] = schedule;
        return schedule;
    }

    [Fact]
    public async Task CreateSchedule(){
        var dto = CreateScheduleDto();
        var result = await _controller.AddSchedule(dto);

        result.Should().BeOfType<CreatedAtActionResult>();
        var created = (result as CreatedAtActionResult)!.Value
as Schedule;
        created.Should().NotNull();

        _fakeDb.Should().ContainKey(created!.Id);
        _fakeDb[created.Id].Instructor_ID.Should().Be(1);

        _output.WriteLine($"Занятие создано\nID занятия:
{created.Id},\nID инструктора: {created.Instructor_ID}");
    }
}
// Фрагмент кода

```

Приложение Д
(рекомендуемое)
Атрибуты сущностей базы данных

Таблица Д.1 – Атрибуты отношения «Instructor»

Идентификатор столбца	Описание	Тип данных	Размерность	Область допустимых значений	Возможность значения Null	Роль	Пример
ID	Уникальный идентификатор инструктора	integer	10	от 1 до 2 147 483 647	нет	PK	1
Surname	Фамилия инструктора	varchar	50	а-Я	нет		Иванов
Name	Имя инструктора	varchar	50	а-Я	нет		Петр
Patronymic	Отчество инструктора	varchar	50	а-Я	да		Викторович
PhoneNumber	Телефон инструктора	varchar	12	Строка, формат: +7XXXXXXXXXX	да		+79998887766
Email	Email инструктора	varchar	100	Строка, email-адрес	да		pochta@mail.ru
Rating	Рейтинг инструктора	float	10	от 0.0 до 5.0	да		4.7
DrivingSchool_ID	Идентификатор связанной автошколы	integer	10	от 1 до 2 147 483 647	нет	FK	2
Car_ID	Идентификатор связанного автомобиля	integer	10	от 1 до 2 147 483 647	да	FK	5
Attachment_ID	Идентификатор связанного вложения	integer	10	от 1 до 2 147 483 647	да	FK	12

Таблица Д.2 – Атрибуты отношения «Student»

Идентификатор столбца	Описание	Тип данных	Размерность	Область допустимых значений	Возможность значения Null	Роль	Пример
ID	Уникальный идентификатор студента	integer	10	от 1 до 2 147 483 647	нет	PK	1
Surname	Фамилия студента	varchar	50	а-Я	нет		Петров
Name	Имя студента	varchar	50	а-Я	нет		Иван
Patronymic	Отчество студента	varchar	50	а-Я	да		Сергеевич
Birthdate	Дата рождения	date		01.01.1900 – 01.01.2100	нет		2003-05-01

Продолжение таблицы Д.2

Идентификатор столбца	Описание	Тип данных	Размерность	Область допустимых значений	Возможность значения Null	Роль	Пример
PhoneNumber	Телефон студента	varchar	12	Строка, формат: +7XXXXXXXXXX	да		+79997776655
Email	Email студента	varchar	100	Email-адрес	да		pochta@mail.ru
Rating	Рейтинг студента	float	10	от 0.0 до 5.0	да		4.9
DrivingSchool_ID	Идентификатор связанной автошколы	integer	10	от 1 до 2 147 483 647	нет	FK	2
Attachment_ID	Идентификатор связанного вложения	integer	10	от 1 до 2 147 483 647	да	FK	10

Таблица Д.3 – Атрибуты отношения «Admin»

Идентификатор столбца	Содержательное описание	Тип данных	Размерность	Область допустимых значений	Null	Роль	Пример
ID	Уникальный идентификатор администратора	integer	10	от 1 до 2 147 483 647	нет	PK	1
Surname	Фамилия администратора	varchar	50	а-Я	нет		Смирнов
Name	Имя администратора	varchar	50	а-Я	нет		Алексей
Patronymic	Отчество администратора	varchar	50	а-Я	да		Олегович
PhoneNumber	Телефон администратора	varchar	12	Строка, формат: +7XXXXXXXXXX	да		+79995554433
Email	Email администратора	varchar	100	Email-адрес	да		pochta@mail.ru
DrivingSchool_ID	Идентификатор связанной автошколы	integer	10	от 1 до 2 147 483 647	нет	FK	3
Attachment_ID	Идентификатор связанного вложения	integer	10	от 1 до 2 147 483 647	да	FK	7

Таблица Д.4 – Атрибуты отношения «Car»

Идентификатор столбца	Содержательное описание	Тип данных	Размерность	Область допустимых значений	Null	Роль	Пример
ID	Уникальный идентификатор авто	integer	10	от 1 до 2 147 483 647	нет	PK	1
Brand	Марка автомобиля	varchar	50	а-Я, а-Z	нет		Hyundai
Model	Модель автомобиля	varchar	50	а-Я, а-Z	нет		Solaris
Color	Цвет автомобиля	varchar	50	а-Я, а-Z	нет		Белый
Car_Number	Госномер автомобиля	varchar	9	Формат: A000AA000	нет		A123BC77
Attachment_ID	Идентификатор вложения	integer	10	от 1 до 2 147 483 647	да	FK	4

Таблица Д.5 – Атрибуты отношения «Attachment»

Идентификатор столбца	Содержательное описание	Тип данных	Размерность	Область допустимых значений	Null	Роль	Пример
ID	Уникальный идентификатор	integer	10	от 1 до 2 147 483 647	нет	PK	1
Name	Ссылка на файл	varchar	max		нет		/Uploads/img.jpg

Таблица Д.6 – Атрибуты отношения «Account»

Идентификатор столбца	Содержательное описание	Тип данных	Размерность	Область допустимых значений	Null	Роль	Пример
ID	Уникальный идентификатор	integer	10	от 1 до 2 147 483 647	нет	PK	1
User_Type	Тип пользователя	integer	10	1 – студент, 2 – инструктор, 3 – админ	нет		1
Login	Логин	varchar	50		нет		petrov_i
Password	Пароль	varchar	50		нет		password123
Student_ID	Связанный студент	integer	10	от 1 до 2 147 483 647	да	FK	3
Instructor_ID	Связанный инструктор	integer	10	от 1 до 2 147 483 647	да	FK	5
Admin_ID	Связанный администратор	integer	10	от 1 до 2 147 483 647	да	FK	2

Таблица Д.7 – Атрибуты отношения «Schedule»

Идентификатор столбца	Содержательное описание	Тип данных	Размерность	Область допустимых значений	Null	Роль	Пример
ID	Уникальный идентификатор	integer	10	от 1 до 2 147 483 647	нет	PK	1
Date	Дата занятия	date		01.01.2020 – 01.01.2100	нет		2025-05-21
SlotTime	Время начала	time	7	00:00 – 23:59	нет		14:00
IsAvailable	Доступность слота	bit		0 – нет, 1 – да	нет		1
Instructor_ID	Идентификатор инструктора	integer	10	от 1 до 2 147 483 647	нет	FK	2
Student_ID	Идентификатор студента	integer	10	от 1 до 2 147 483 647	да	FK	5
CreatedAt	Дата создания записи	date		01.01.2020 – 01.01.2100	нет		2025-05-01

Таблица Д.8 – Атрибуты отношения «Review»

Идентификатор столбца	Содержательное описание	Тип данных	Размерность	Область допустимых значений	Null	Роль	Пример
ID	Уникальный идентификатор	integer	10	от 1 до 2 147 483 647	нет	PK	1
Type_Id	Тип отзыва	integer	10	0 – текст, 1 – рейтинг	нет		1
Mark	Оценка	integer	10	от 1 до 5	да		5
Text	Содержание отзыва	varchar	max	а-Я, а-Z, 0-9	да		Отличный инструктор!
Student_ID	Идентификатор студента	integer	10	от 1 до 2 147 483 647	нет	FK	2
Instructor_ID	Идентификатор инструктора	integer	10	от 1 до 2 147 483 647	нет	FK	4
CreatedAt	Дата публикации	date		01.01.2020 – 01.01.2100	нет		2025-05-01

Таблица Д.9 – Атрибуты отношения «Driving_School»

Идентификатор столбца	Содержательное описание	Тип данных	Размерность	Область допустимых значений	Null	Роль	Пример
ID	Уникальный идентификатор	integer	10	от 1 до 2 147 483 647	нет	РК	1
Address	Адрес автошколы	varchar	200		нет		г. Москва, ул. Ленина, 5
PhoneNumber	Телефон автошколы	varchar	12	Строка, формат: +7XXXXXXXXXX	нет		+74991234567
Email	Электронная почта автошколы	varchar	100	Строка, формат: email-адрес	нет		Автошкола №1
City_ID	Идентификатор города	integer	10	от 1 до 2 147 483 647	нет	FK	3

Таблица Д.10 – Атрибуты отношения «City»

Идентификатор столбца	Содержательное описание	Тип данных	Размерность	Область допустимых значений	Null	Роль	Пример
ID	Уникальный идентификатор	integer	10	от 1 до 2 147 483 647	нет	РК	1
Name	Название города	varchar	100	а-Я	нет		Москва

Приложение Е
(рекомендуемое)
Графическая часть

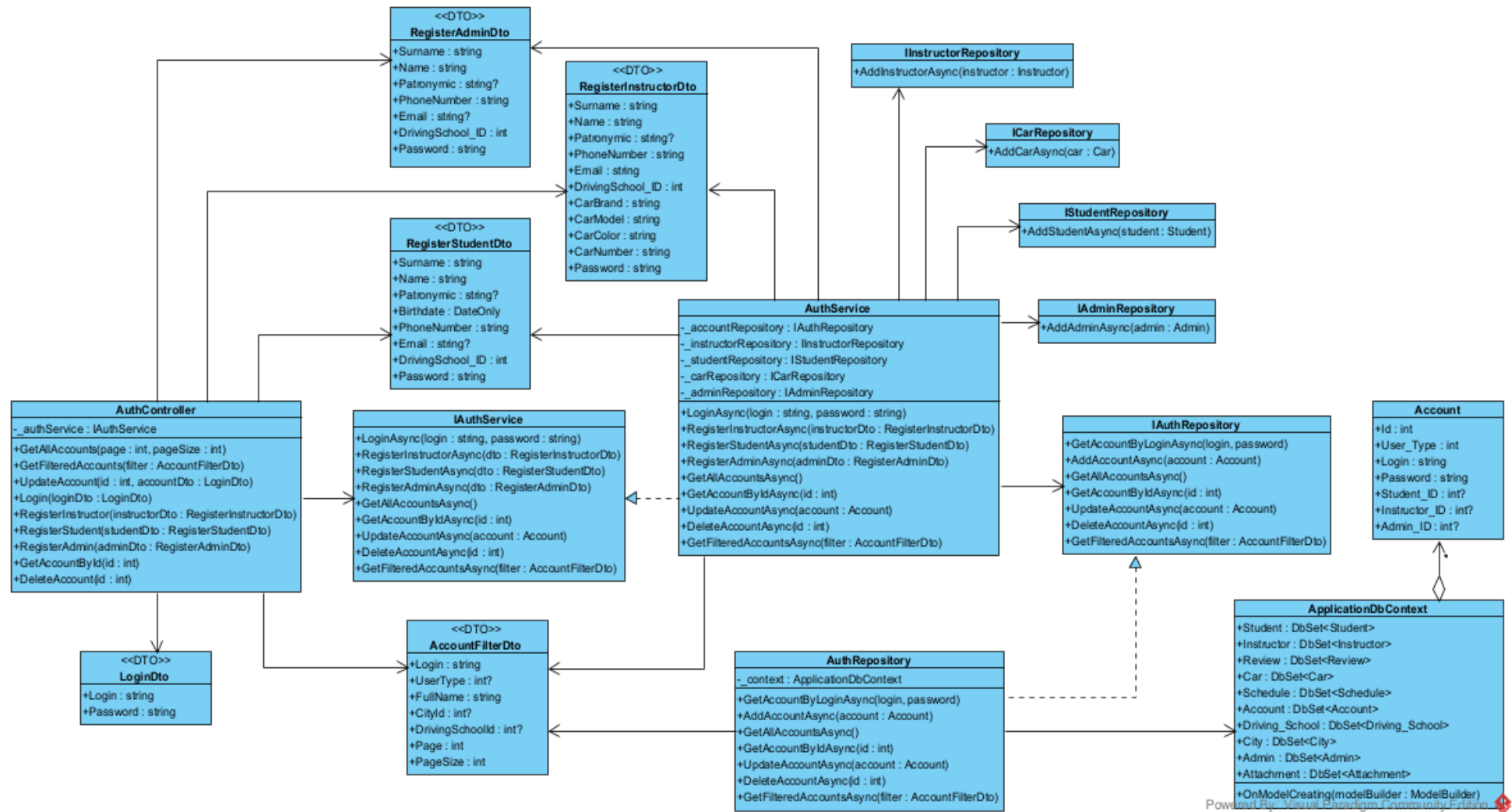


Рисунок Е.1 – Диаграмма классов модуля «Аутентификация»

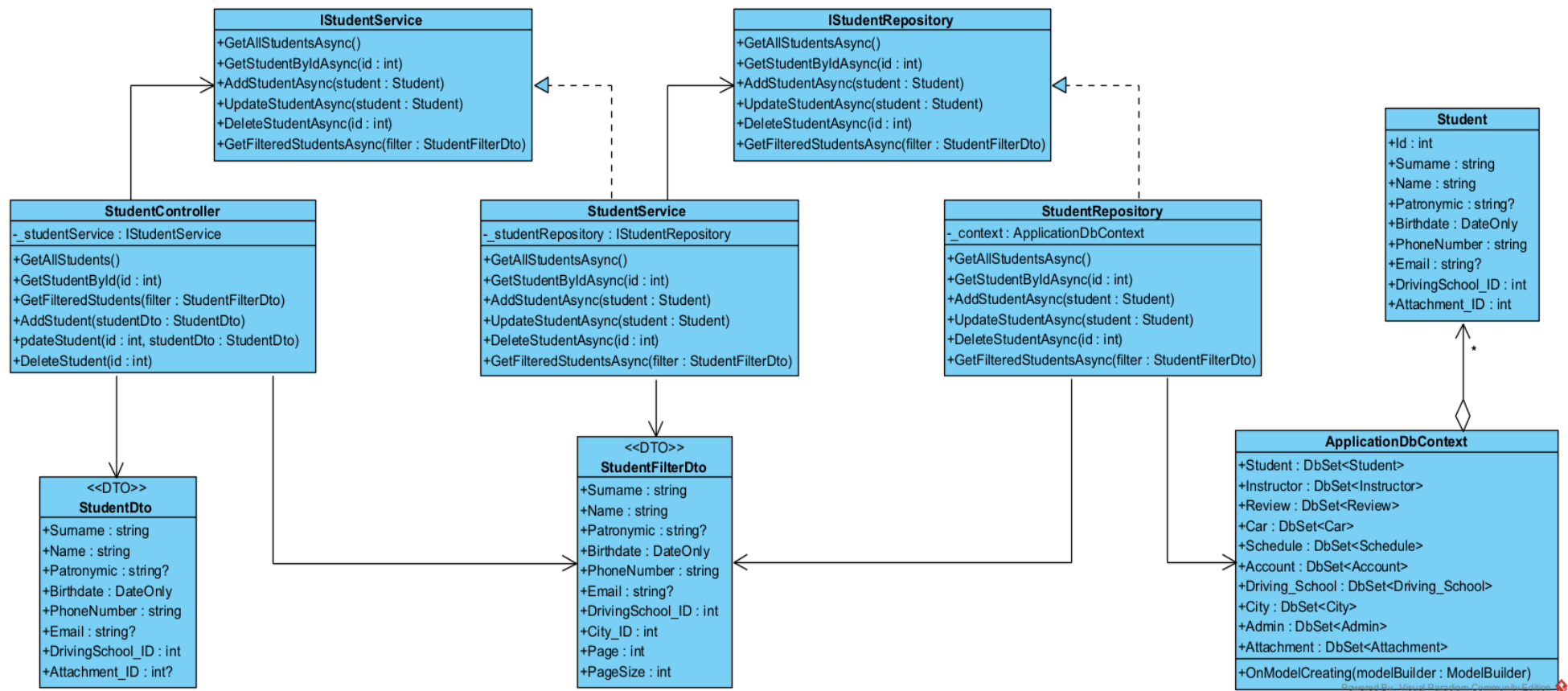


Рисунок Е.2 – Диаграмма классов модуля «Студенты»

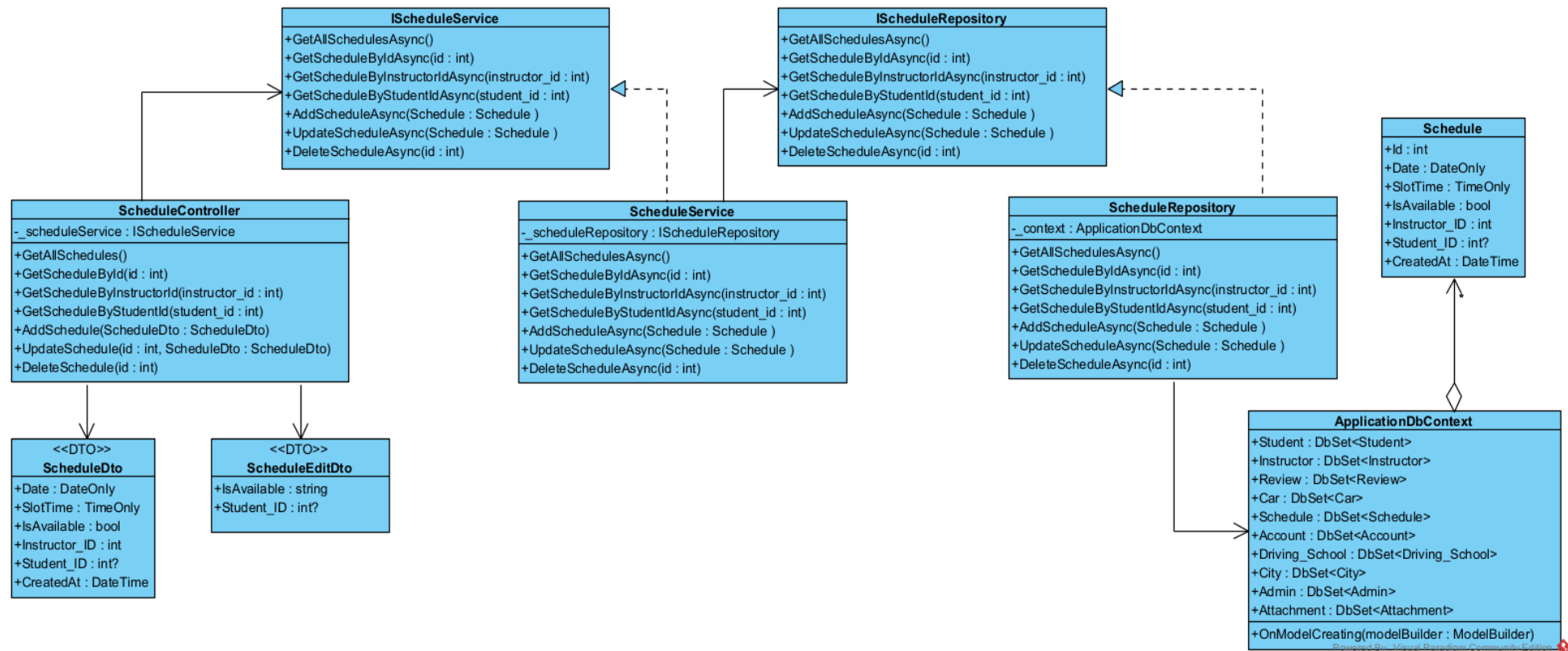


Рисунок Е.3 – Диаграмма классов модуля «Расписание»

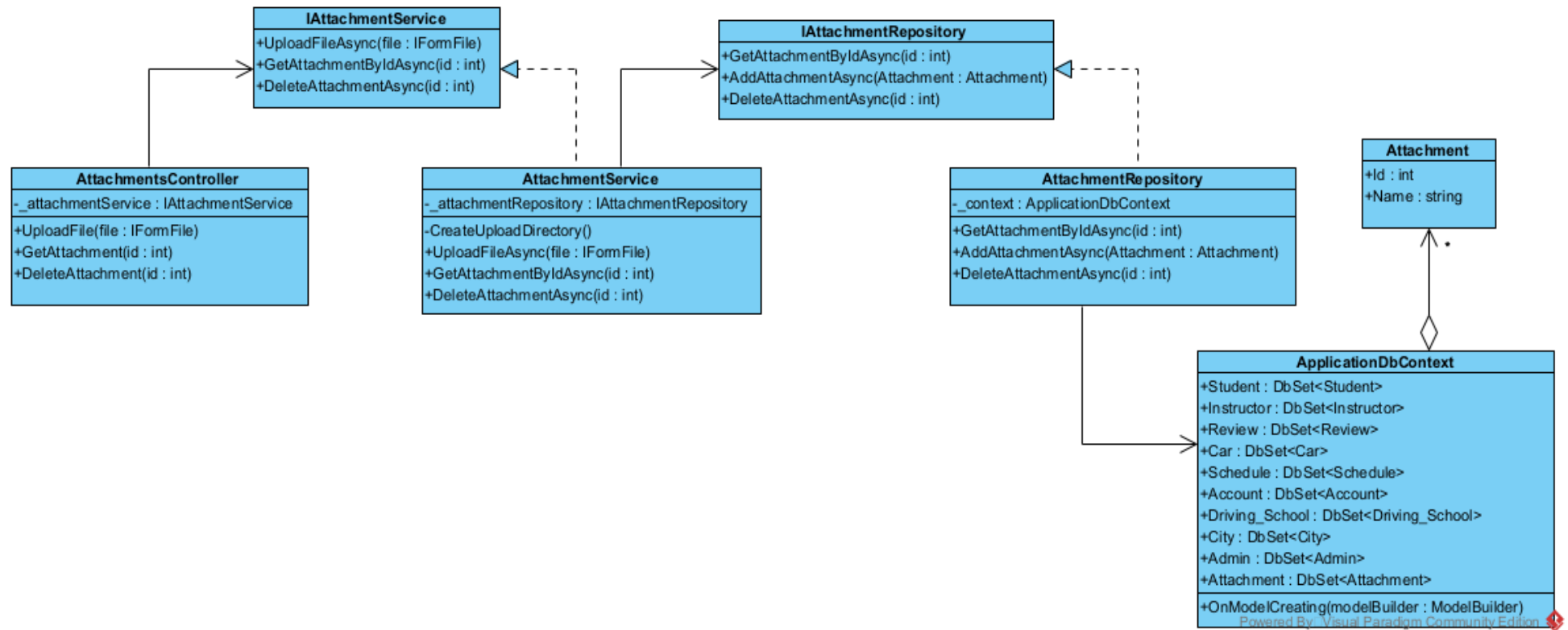


Рисунок Е.4 – Диаграмма классов модуля «Вложения»

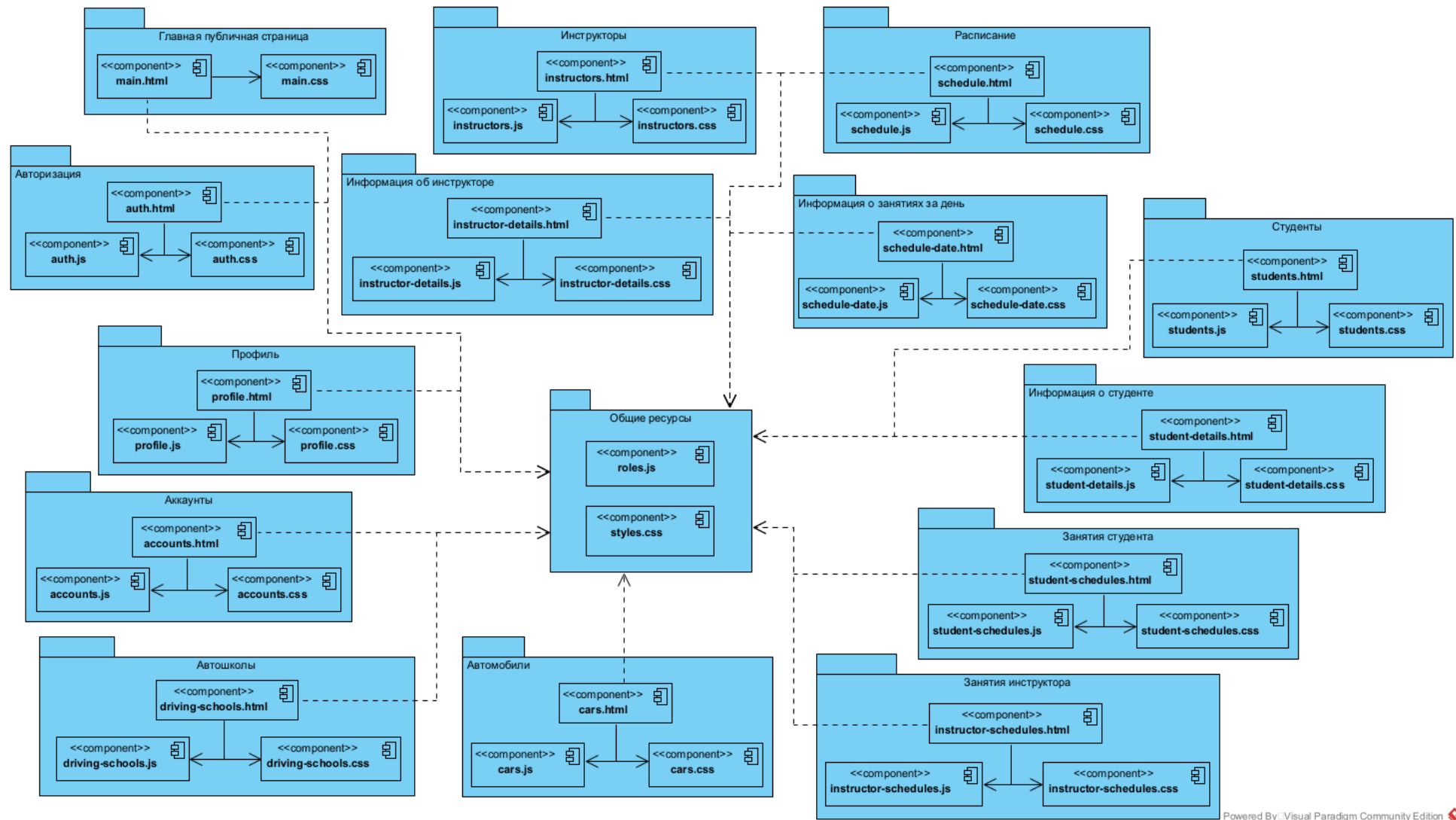


Рисунок Е.5 – Диаграмма компонентов системы

Приложение Ж
(рекомендуемое)
Результаты функционального тестирования

Расписание							Добавить
Июнь 2025							
Пн	Вт	Ср	Чт	Пт	Сб	Вс	
						1	
2	3	4	5	6	7	8	
9	10 Свободно: 1	11 Свободно: 2	12	13 Свободно: 1	14	15 Свободно: 1	
16 Свободно: 1	17	18	19 Свободно: 1	20	21	22	
23 Свободно: 1	24 Свободно: 1	25 Свободно: 1	26	27	28	29	
30							

Рисунок Ж.1 – Расписание занятий

Создание записи

Дата:

28.06.2025

Время:

11:30

Создать запись

Отмена

Рисунок Ж.2 – Ввод даты и времени занятия

Июнь 2025							
Пн	Вт	Ср	Чт	Пт	Сб	Вс	
						1	
2	3	4	5	6	7	8	
9	10 Свободно: 1	11 Свободно: 2	12	13 Свободно: 1	14	15 Свободно: 1	
16 Свободно: 1	17	18	19 Свободно: 1	20	21	22	
23 Свободно: 1	24 Свободно: 1	25 Свободно: 1	26	27	28 Свободно: 1	29	
30							

Рисунок Ж.3 – Расписание с добавленным занятием

Записи на 28 июня 2025**11:30***Свободно***Инструктор:** Карпов Игорь
Анатольевич**Номер телефона:**
89012345678**Автомобиль:** Volkswagen Polo,
Черный**Рег. номер:** H131KA790[Записаться](#)

Рисунок Ж.4 – Свободное занятие

Мои записи**28 июня 2025****Время: 11:30***Занято***Инструктор:** Карпов Игорь
Анатольевич**Телефон:** 89012345678**Автомобиль:** Volkswagen Polo,
Черный**Рег. номер:** H131KA790[Отменить](#)

Рисунок Ж.5 – Забронированное занятие

Июнь 2025						
Пн	Вт	Ср	Чт	Пт	Сб	Вс
						1
2	3	4	5	6	7	8
9	10 Свободно: 1	11 Свободно: 2	12	13 Свободно: 1	14	15 Свободно: 1
16 Свободно: 1	17	18	19 Свободно: 1	20	21	22
23 Свободно: 1	24 Свободно: 1	25 Свободно: 2	26	27	28	29
30						

Рисунок Ж.6 – Забронированное занятие в расписании

Подтверждение отмены

Вы уверены, что хотите отменить эту запись?

Внимание: эта операция необратима.

Подтвердить Отмена

Рисунок Ж.7 – Окно подтверждения отмены записи на занятие

Расписание						
Июнь 2025						
Пн	Вт	Ср	Чт	Пт	Сб	Вс
						1
2	3	4	5	6	7	8
9	10 Свободно: 1	11 Свободно: 2	12	13 Свободно: 1	14	15 Свободно: 1
16 Свободно: 1	17	18	19 Свободно: 1	20	21	22
23 Свободно: 1	24 Свободно: 1	25 Свободно: 2	26	27	28 Свободно: 1	29
30						

Рисунок Ж.8 – Расписание после отмены записи на занятие

Мои записи

10 июня 2025

Время: 11:00

Свободно

Отменить

Рисунок Ж.9 – Карточка с занятием для инструктора

Расписание

Добавить

Июнь 2025

Пн	Вт	Ср	Чт	Пт	Сб	Вс
						1
2	3	4	5	6	7	8
9	10	11 Свободно: 2	12	13 Свободно: 1	14	15 Свободно: 1
16 Свободно: 1	17	18	19 Свободно: 1	20	21	22
23 Свободно: 1	24 Свободно: 1	25 Свободно: 2	26	27	28 Свободно: 1	29
30						

Рисунок Ж.10 – Расписание после отмены занятия

Профиль пользователя

Фамилия: Никифоров

Имя: Игорь

Отчество: Иванович

Телефон: 89002223344

Email: ivanov@mail.com

Логин: 88005553535

Пароль:

Дата рождения: 2005-02-12

Автошкола: г. Пенза, ул. Пушкина, 44

Рейтинг: 4.5

Редактировать профиль

Выйти

Рисунок Ж.11 – Профиль пользователя

Профиль пользователя



Фамилия: Никифоров
Имя: Игорь
Отчество: Иванович
Телефон: 89002223344
Email: ivanov@mail.com
Логин: 88005553535
Пароль: Введите новый пароль
Дата рождения: 2005-02-12
Автошкола: г. Пенза, ул. Пушкина, 44
Рейтинг: 4.5

Сохранить

Отмена

Выйти

Рисунок Ж.12 – Профиль пользователя в режиме редактирования

Профиль пользователя



Фамилия: Лобанов
Имя: Семен
Отчество: Иванович
Телефон: 89001112233
Email: lobanov@mail.com
Логин: 88005553535
Пароль: Введите новый пароль
Дата рождения: 2005-04-13
Автошкола: г. Пенза, ул. Пушкина, 44
Рейтинг: 4.5

Сохранить

Отмена

Выйти

Рисунок Ж.13 – Измененные данные профиля в режиме редактирования

Профиль пользователя



Фамилия: Лобанов
Имя: Семен
Отчество: Иванович
Телефон: 89001112233
Email: lobanov@mail.com
Логин: 88005553535
Пароль:
Дата рождения: 2005-04-13
Автошкола: г. Пенза, ул. Пушкина, 44
Рейтинг: 4.5

Рисунок Ж.14 – Измененные данные профиля

Профиль пользователя



Фамилия:
Имя:
Отчество:
Телефон:
Email:
Логин:
Пароль:
Дата рождения:
Автошкола: г. Пенза, ул. Пушкина, 44
Рейтинг: 4.5

Рисунок Ж.15 – Измененные данные профиля в режиме редактирования

Профиль пользователя



Фамилия: Лобанов
Имя: Семен
Отчество: Иванович
Телефон: 89001112233
Email: lobanov@mail.com
Логин: 88005553535
Пароль:
Дата рождения: 2005-04-13
Автошкола: г. Пенза, ул. Пушкина, 44
Рейтинг: 4.5

Рисунок Ж.16 – Профиль без изменений (данные не изменены)

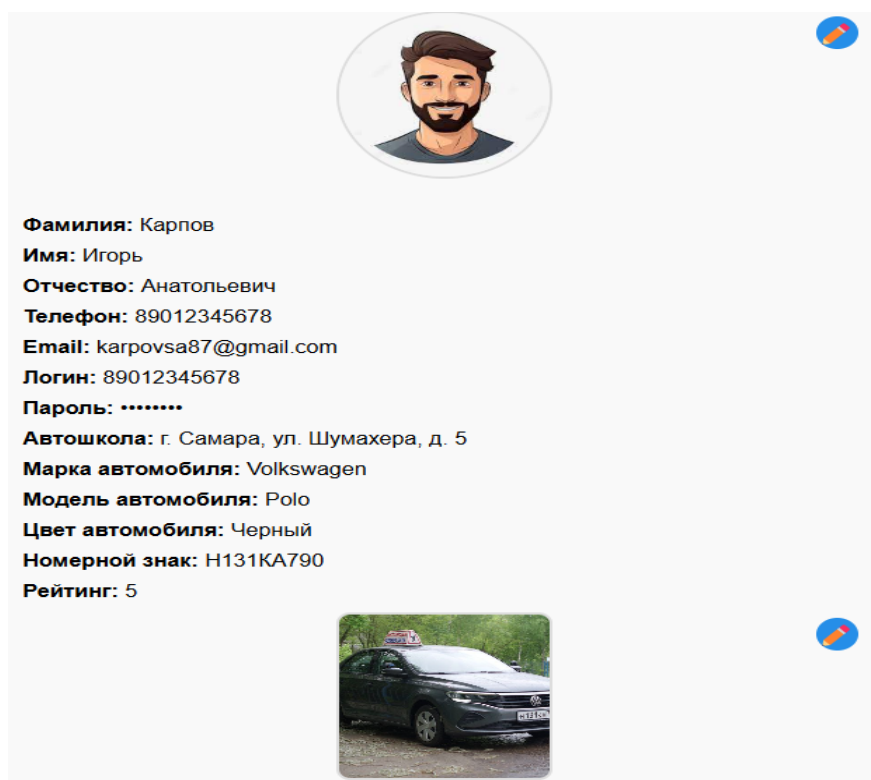


Рисунок Ж.17 – Профиль инструктора до изменений

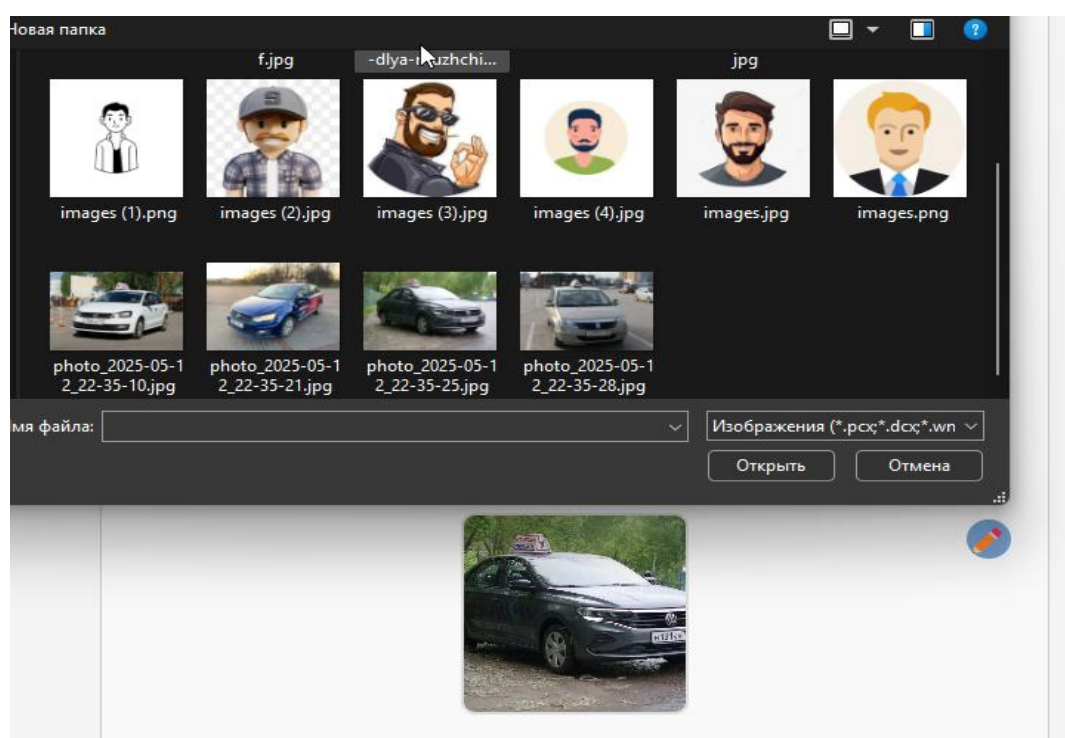
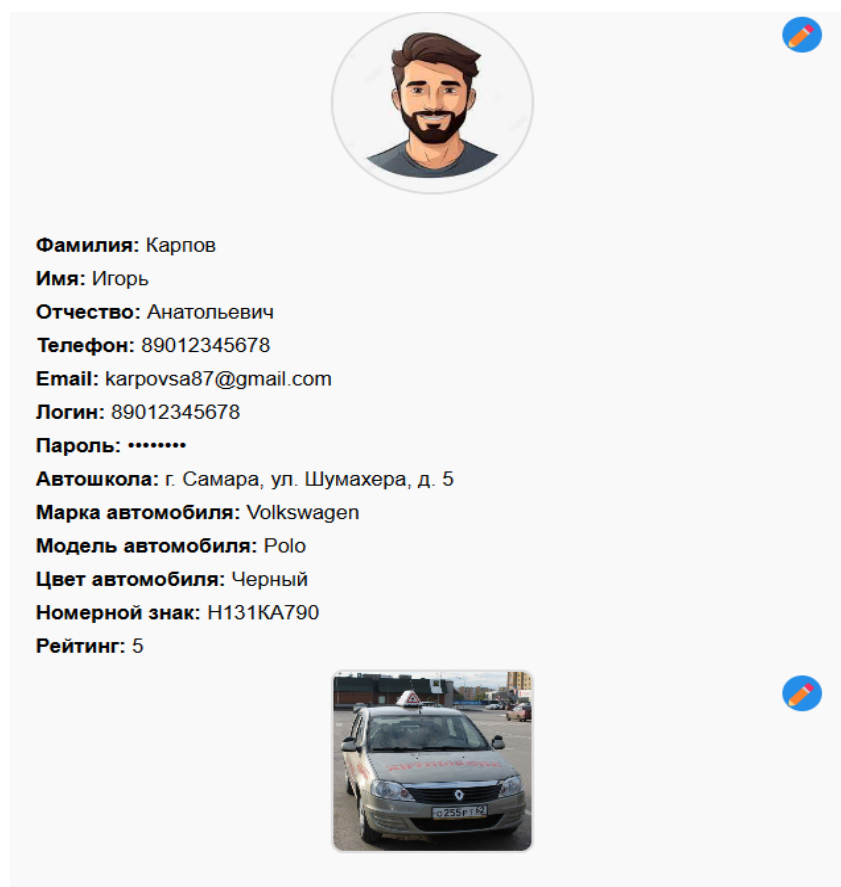


Рисунок Ж.18 – Выбор файла на устройстве



Фамилия: Карпов
Имя: Игорь
Отчество: Анатольевич
Телефон: 89012345678
Email: karpovsa87@gmail.com
Логин: 89012345678
Пароль:
Автошкола: г. Самара, ул. Шумахера, д. 5
Марка автомобиля: Volkswagen
Модель автомобиля: Polo
Цвет автомобиля: Черный
Номерной знак: H131KA790
Рейтинг: 5



Рисунок Ж.19 – Обновленное фото автомобиля инструктора

Список аккаунтов Добавить

Логин (номер телефона)	Пароль	Тип пользователя	ФИО	Город	Автошкола	Действия
Аккаунты не найдены.						

Записей на странице: 10 Назад Страница 1 из 1 Вперед

Фильтры

Логин (номер телефона):

Тип пользователя:

ФИО:

Город:

Автошкола:

Применить Сбросить фильтры

Рисунок Ж.20 – Список аккаунтов

Добавление аккаунта

Тип пользователя: *

Фамилия: *

Имя: *

Отчество (если есть):

Номер телефона (логин): *

Пароль: *

День рождения: *

Email:

Город: *

Автошкола: *

Рисунок Ж.21 – Модальное окно добавления аккаунта

Список аккаунтов							Добавить
Фильтры	Логин (номер телефона)	Пароль	Тип пользователя	ФИО	Город	Автошкола	Действия
Логин (номер телефона): <input type="text" value="Введите логин..."/> Тип пользователя: <input type="text" value="Любой"/> ФИО: <input type="text" value="царев"/> Город: <input type="text" value="Любой"/>	88005554545	<input type="password" value="....."/>	Студент	Царев Виктор Семенович	Саратов	ул. Советская, д.6	<input type="button" value="Редактировать"/> <input type="button" value="Удалить"/>
Записей на странице: <input type="text" value="10"/>							<input type="button" value="Назад"/> Страница 1 из 1 <input type="button" value="Вперед"/>

Рисунок Ж.22 – Список аккаунтов с добавленным аккаунтом

Редактирование аккаунта

Логин (номер телефона):

Пароль:

Рисунок Ж.23 – Модальное окно редактирования аккаунта

Редактирование аккаунта

Логин (номер телефона):

Пароль:

Рисунок Ж.24 – Измененные данные в модальном окне редактирования

Логин (номер телефона)	Пароль	Тип пользователя	ФИО	Город
89354678923	ПАРОЛЬ	Студент	Царев Виктор Семенович	Саратов

Рисунок Ж.25 – Список аккаунтов с измененным аккаунтом

Логин (номер телефона)	Пароль	Тип пользователя	ФИО	Город	Автошкола	Действия
79056783421	ПАРОЛЬ	Студент	Филимонов Петр Кириллович	Нижний Ломов	ул. Курчанова, 73	 
79085327856	ПАРОЛЬ	Студент	Самсонов Игорь Кириллович	Нижний Ломов	ул. Курчанова, 73	 
89003458967	ПАРОЛЬ	Инструктор	Ивакуров Артем Григорьевич	Нижний Ломов	ул. Иванова, 52	 
83546789245	ПАРОЛЬ	Инструктор	Перивалов Тимофей Семенович	Нижний Ломов	ул. Курчанова, 73	 
89067864512	ПАРОЛЬ	Инструктор	Криваев Антон Федорович	Нижний Ломов	ул. Иванова, 52	 

Записей на странице: 10 Страница 1 из 1

Рисунок Ж.26 – Список аккаунтов

Подтверждение удаления

Вы уверены, что хотите удалить аккаунт из системы?

Внимание: при удалении аккаунта вся информация о владельце учетной записи будет удалена.

Рисунок Ж.27 – Окно подтверждения удаления

Логин (номер телефона)	Пароль	Тип пользователя	ФИО	Город	Автошкола	Действия
79056783421	пароль	Студент	Филимонов Петр Кириллович	Нижний Ломов	ул. Курчанова, 73	 
79085327856	пароль	Студент	Самсонов Игорь Кириллович	Нижний Ломов	ул. Курчанова, 73	 
83546789245	пароль	Инструктор	Перивалов Тимофей Семенович	Нижний Ломов	ул. Курчанова, 73	 
89067864512	пароль	Инструктор	Криваев Антон Федорович	Нижний Ломов	ул. Иванова, 52	 

Записей на странице: 

[Назад](#) [Страница 1 из 1](#) [Вперед](#)

Рисунок Ж.28 – Список аккаунтов после удаления аккаунта