

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РФ
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«ПЕНЗЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»
ПОЛИТЕХНИЧЕСКИЙ ИНСТИТУТ

<i>Факультет</i>	<i>Кафедра</i>
вычислительной техники	МОиПЭВМ
Направление подготовки	09.03.02 «Информационные системы и технологии»
Направленность	Информационные системы и технологии

БАКАЛАВРСКАЯ РАБОТА
на тему
Разработка и внедрение информационной
системы детейлинг-центра

Студент	_____	Краснопеев Даниил Сергеевич
	(подпись, дата)	(ФИО полностью)
Руководитель	_____	Кольчугина Е.А.
	(подпись, дата)	(фамилия, инициалы)
Нормоконтролёр	_____	Такташкин Д.В.
	(подпись, дата)	(фамилия, инициалы)

Работа допущена к защите (протокол заседания кафедры от _____ № _____)

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, 2026

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
«26» декабря 2025 г

**ЗАДАНИЕ
НА ВЫПУСКНУЮ КВАЛИФИКАЦИОННУЮ РАБОТУ
БАКАЛАВРА**

1. Студент Краснопеев Даниил Сергеевич гр. 22ВИ1 факультета ВТ

2. Тема работы Разработка и внедрение информационной системы детейлинг-центра

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

3. Руководитель работы д.т.н., доцент кафедры «МОиПЭВМ» Е. А. Кольчугина

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

1) Разработать и внедрить информационную систему детейлинг-центра.

2) Среда выполнения системы: ОС Windows.

Языковые средства разработки: C#, UML.

Инструментальные средства разработки: Microsoft Visual Studio, Microsoft SQL Server Management Studio, ProjectLibre, draw.io.

Технологии разработки: Windows Forms, ADO.NET, ООП.

3) Основные функции системы:

а) авторизация пользователей в системе;

б) создание и обработка клиентских заявок;

в) ведение данных о клиентах и автомобилях;

г) формирование и ведение заказ-нарядов;

д) учет выполняемых услуг и используемых материалов;

е) контроль статусов выполнения работ;

ж) управление складскими остатками материалов;

з) управление данными сотрудников;

и) формирование отчетов за выбранный период.

4. Перечень подлежащих разработке вопросов

1) *Анализ предметной области и постановка задачи.*

2) *Проектирование информационной системы.*

3) *Реализация информационной системы.*

4) *Тестирование программного обеспечения.*

5) *Оформление пояснительной записки.*

5. Календарный график выполнения работы

№ п/п	Наименование этапов работы	Объем работы	Срок выполнения	Подпись руководителя
1	Анализ предметной области и требований к системе	10%	07.04.2026 – 13.04.2026	
2	Проектирование программных средств	20%	14.04.2026 – 25.04.2026	
3	Реализация программных средств	30%	27.04.2026 – 11.05.2026	
4	Тестирование программных средств	20%	13.05.2026 – 22.05.2026	
5	Оформление пояснительной записки	20%	22.05.2026 – 02.06.2026	

Дата выдачи задания «26» декабря 2025г.

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 114 листов, 38 рисунков, 17 таблиц, 20 использованных источников, 3 приложения.

АВТОДЕТЕЙЛИНГ, ИНФОРМАЦИОННАЯ СИСТЕМА, КЛИЕНТСКАЯ ЗАЯВКА, ЗАКАЗ-НАРЯД, МАТЕРИАЛЫ, СОТРУДНИКИ, ОТЧЕТНОСТЬ, C#, WINDOWS FORMS, MICROSOFT SQL SERVER, UML

Объектом разработки является информационная система детейлинг-центра.

Цель работы – разработка и внедрение информационной системы, позволяющей автоматизировать учет клиентских заявок, автомобилей, заказ-нарядов, выполняемых работ, используемых материалов, сотрудников и отчетности в детейлинг-центре.

Разработка информационной системы осуществлялась на языке программирования C# с использованием технологии Windows Forms. Для реализации программного средства использовалась среда разработки Microsoft Visual Studio. В качестве системы управления базами данных использовалась Microsoft SQL Server.

Информационная система предназначена для работы на устройствах под управлением операционной системы Windows. Система обеспечивает авторизацию пользователей, разграничение доступа в зависимости от роли, создание и обработку клиентских заявок, ведение заказ-нарядов, учет материалов, управление сотрудниками и формирование отчетов за выбранный период.

					ПГУ 09.03.02 – 08ВР221.12 ПЗ						
Изм.	Лист	№ докум.	Подп.	Дата	Разработка и внедрений информационной система детейлинг-центра. Пояснительная записка			Лит.		Лист	Листов
Разраб.		Краснопеев Д. С.								4	114
Пров.		Кольчугина Е. А.									
Реценз.											
Н. контр.		Такташкин Д.В.									
Утверд.											
					Группа 22ВИ1						

Содержание

Введение.....	7
1 Анализ предметной области и требований к системе	9
1.1 Исследование предметной области	9
1.2 Анализ информационных систем аналогичного назначения	13
1.3 Анализ функциональных требований	21
1.4 Анализ нефункциональных требований	27
1.5 Планирование разработки и оценка бюджета	29
2 Проектирование программных средств	36
2.1 Архитектура приложения	36
2.2 Проектирование структур данных.....	40
2.2.1 Концептуальная модель базы данных	40
2.2.2 Логическая модель базы данных	44
2.3 Проектирование алгоритмов	46
2.4 Проектирование пользовательского интерфейса.....	54
3 Реализация программных средств	60
3.1 Реализация базы данных.....	60
3.2 Реализация модуля авторизации.....	69
3.3 Реализация главного окна приложения	74
3.4 Реализация работы с клиентскими заявками.....	78
3.5 Реализация работы с задачами.....	81
3.6 Реализация складского учета	85
3.7 Реализация управления сотрудниками	87
3.8 Реализация формирования отчетности	88
4 Тестирование системы.....	92
4.1 Методика ручного тестирования	92
4.2 Функциональное тестирование.....	93
4.3 Тестирование обработки ошибок	95
4.4 Метрики качества кода	96

Заключение	98
Список использованных источников	100
Приложение А (справочное) Глоссарий	103
Приложение Б (обязательное) Исходный код приложения	105
Приложение В (рекомендуемое) Исходный код модульного тестирования	110

Введение

В современных условиях информационные технологии являются важным инструментом повышения эффективности организаций сферы услуг. Автоматизация позволяет сократить время обработки заявок, снизить количество ошибок при учете, повысить качество обслуживания клиентов и обеспечить удобное хранение информации.

Сфера детейлинга развивается, поскольку владельцы автомобилей все чаще обращаются за услугами профессионального ухода за кузовом, салоном и отдельными элементами транспортного средства. В работе детейлинг-центра необходимо учитывать заявки клиентов, автомобили, выполняемые работы, используемые материалы, сотрудников, статусы заказов и результаты оказанных услуг.

При ручном учете или использовании разрозненных таблиц возрастает риск потери данных, дублирования записей, ошибок при расчете стоимости и сложностей при формировании отчетности. Поэтому актуальность разработки информационной системы обусловлена необходимостью автоматизации основных бизнес-процессов детейлинг-центра.

Объектом исследования является процесс организации и учета работ в детейлинг-центре. Предметом исследования является информационная система, предназначенная для автоматизации учета заявок, работ, сотрудников, автомобилей, материалов и отчетности.

Целью выпускной квалификационной работы является разработка и внедрение информационной системы для автоматизации деятельности детейлинг-центра. Для достижения поставленной цели необходимо решить следующие задачи:

- проанализировать предметную область и выявить основные проблемы учета работ в детейлинг-центре;
- рассмотреть существующие информационные системы аналогичного назначения и определить их преимущества и недостатки;

- сформулировать функциональные и нефункциональные требования к разрабатываемой системе;
- определить технические средства реализации информационной системы;
- спроектировать структуру базы данных и основные программные компоненты;
- реализовать приложение с учетом поставленных требований;
- провести проверку работоспособности основных функций системы.

Разрабатываемая информационная система должна быть ориентирована на использование сотрудниками детейлинг-центра и обеспечивать удобную работу с клиентскими заявками, заказами, материалами, сотрудниками и отчетами. Предполагается, что приложение сможет использоваться в локальной сети организации и не будет требовать постоянного подключения к интернету для выполнения основных функций.

В рамках выпускной квалификационной работы выполняется анализ предметной области, постановка задачи на разработку, формирование требований к системе, проектирование базы данных и программных средств, реализация информационной системы детейлинг-центра, а также проверка работоспособности основных функций разработанного приложения.

1 Анализ предметной области и требований к системе

1.1 Исследование предметной области

Детейлинг представляет собой направление деятельности, связанное с профессиональным уходом, восстановлением внешнего вида и защитой автомобиля. В рамках данной работы детейлинг рассматривается не только как мойка, полировка и химчистка, но и как комплекс услуг по устранению внешних дефектов транспортного средства. К таким дефектам могут относиться мелкие вмятины, следы коррозии, царапины, потертости, повреждения лакокрасочного покрытия, загрязнения салона и другие недостатки, влияющие на внешний вид и потребительские свойства автомобиля.

В Российской Федерации деятельность, связанная с обслуживанием автомобилей, относится к сфере технического обслуживания и ремонта автотранспортных средств. В соответствии с ОКВЭД 2 код 45.20 включает техническое обслуживание и ремонт автотранспортных средств, а код 45.20.3 включает мойку автотранспортных средств, полирование и предоставление аналогичных услуг [1]. Это позволяет рассматривать детейлинг как одно из направлений деятельности в сфере обслуживания автомобилей, включающее как уходные, так и восстановительные работы, направленные на улучшение состояния автомобиля.

В отличие от обычной автомобильной мойки, детейлинг предполагает более детальный и индивидуальный подход к каждому заказу. Перед выполнением работ специалист оценивает состояние автомобиля, выявляет имеющиеся дефекты, определяет перечень необходимых услуг, подбирает материалы и согласовывает порядок выполнения работ. В зависимости от состояния автомобиля заказ может включать химчистку салона, полировку кузова, восстановление лакокрасочного покрытия, локальную обработку следов коррозии, устранение мелких повреждений, удаление загрязнений и нанесение защитных составов.

К основным направлениям деятельности детейлинг-центра можно отнести:

- прием и оформление заявок клиентов;
- регистрацию данных о клиенте и автомобиле;
- предварительную оценку состояния транспортного средства;
- определение перечня необходимых работ;
- назначение ответственного сотрудника или исполнителя;
- учет используемых материалов и расходных средств;
- контроль статуса выполнения заказа;
- фиксацию завершения работ;
- расчет итоговой стоимости;
- формирование отчетности за выбранный период.

Работа детейлинг-центра связана с большим количеством взаимосвязанных данных. При оформлении заявки необходимо учитывать сведения о клиенте, автомобиле, характере повреждений или загрязнений, требуемых работах, используемых материалах, сотрудниках и текущем статусе заказа. Например, одна заявка может включать только химчистку салона, другая – восстановление лакокрасочного покрытия и устранение мелких дефектов кузова, третья – комплексную обработку автомобиля с применением нескольких видов материалов. Поэтому учет в такой организации должен обеспечивать связь между заявкой, автомобилем, работами, исполнителями и расходными материалами.

Особое значение имеет контроль сроков выполнения работ. Деятельность детейлинг-центра относится к оказанию услуг, поэтому организация должна соблюдать согласованные с клиентом сроки выполнения заказа. Закон РФ «О защите прав потребителей» устанавливает, что исполнитель обязан выполнить работу или оказать услугу в срок, предусмотренный правилами выполнения соответствующих работ либо договором [2]. Следовательно, для детейлинг-центра важно фиксировать дату создания заявки, этап выполнения, дату завершения и текущий статус заказа.

Кроме того, в процессе работы детейлинг-центр обрабатывает персональные данные клиентов и сотрудников. К таким данным могут относиться фамилия, имя, отчество, номер телефона, сведения о заказе, а также иная информация, позволяющая прямо или косвенно определить физическое лицо. Федеральный закон «О персональных данных» определяет персональные данные как любую информацию, относящуюся к прямо или косвенно определенному или определяемому физическому лицу [3]. Поэтому при проектировании информационной системы необходимо предусмотреть хранение данных в контролируемой среде и ограничение доступа к информации в зависимости от роли пользователя.

На практике небольшие детейлинг-центры часто используют простые способы учета: бумажные журналы, электронные таблицы, заметки, мессенджеры или универсальные CRM-системы. Такой подход может быть удобен на начальном этапе работы организации, однако при увеличении количества заказов возникают проблемы. Бумажный учет затрудняет быстрый поиск информации и не позволяет оперативно сформировать отчет. Электронные таблицы не обеспечивают полноценного разграничения доступа, контроля статусов и связности данных. Мессенджеры удобны для общения, но не подходят для системного хранения информации о клиентах, автомобилях, материалах и выполненных работах.

Универсальные CRM-системы позволяют автоматизировать часть процессов, однако они не всегда подходят для небольших детейлинг-центров. Такие решения могут быть избыточными по функциональности, требовать настройки под конкретные бизнес-процессы, работать по подписочной модели и зависеть от постоянного подключения к интернету. Кроме того, при использовании облачных сервисов данные клиентов и сотрудников могут храниться на стороне внешнего поставщика, что не всегда удобно для организаций, стремящихся самостоятельно контролировать хранение и обработку информации.

Для детейлинг-центра важны следующие информационные процессы:

- авторизация сотрудников в системе;
- ведение данных о клиентах;
- ведение данных об автомобилях;
- создание клиентских заявок;
- обработка заявок и изменение их статусов;
- формирование задач или заказов на выполнение работ;
- назначение сотрудников на выполнение работ;
- учет материалов и расходных средств;
- фиксация факта завершения работ;
- формирование отчетов за выбранный период;
- анализ выполненных заказов и работы сотрудников.

Автоматизация перечисленных процессов позволяет снизить объем ручной работы, уменьшить вероятность ошибок, ускорить обработку заявок и повысить качество обслуживания клиентов. При наличии единой информационной системы сотрудники получают возможность быстро находить нужную информацию, отслеживать состояние заказа, контролировать наличие материалов и формировать отчеты без ручного сбора данных из разных источников.

Разрабатываемая информационная система должна быть ориентирована на использование внутри детейлинг-центра. Основной задачей системы является автоматизация учета заявок, автомобилей, сотрудников, работ, материалов и отчетности. При этом целесообразно предусмотреть возможность работы приложения в локальной инфраструктуре организации. Такой подход позволит использовать систему без обязательного постоянного подключения к интернету, например в рамках локальной сети, а также не передавать данные клиентов и сотрудников во внешние облачные сервисы.

Предметная область детейлинга характеризуется большим количеством взаимосвязанных процессов, связанных с приемом заявок, учетом автомобилей, восстановительными работами, использованием материалов, контролем

сотрудников и формированием отчетности. Ручной учет или применение разрозненных инструментов не обеспечивает достаточного уровня надежности, удобства и контроля. Поэтому разработка информационной системы для детейлинг-центра является актуальной задачей, направленной на повышение эффективности работы организации, улучшение качества обслуживания клиентов и обеспечение более удобного контроля выполняемых работ.

1.2 Анализ информационных систем аналогичного назначения

Для обоснования необходимости разработки информационной системы детейлинг-центра был проведен анализ программных решений, предназначенных для автоматизации деятельности автосервисов, детейлинг-центров, автомоек и организаций, оказывающих услуги по обслуживанию автомобилей. При рассмотрении аналогов учитывались следующие критерии: наличие учета клиентов и автомобилей, возможность оформления заявок или заказ-нарядов, учет работ и материалов, контроль статусов выполнения, формирование отчетности, стоимость использования, зависимость от интернет-соединения и возможность адаптации под процессы небольшой организации.

Одним из распространенных решений является CRM-система LiveSklad, предназначенная для автоматизации автосервиса и СТО. Система позволяет вести учет клиентов, заказов, склада, финансов и сотрудников в одном сервисе. На сайте разработчика указано, что LiveSkлад поддерживает оформление заказов, назначение мастеров, контроль статусов, учет использованных запчастей, работу с документами, аналитику и отчеты [4]. Также сервис предоставляет возможность хранения истории заказов, оплат и звонков клиентов [5]. Пользовательский интерфейс системы реализован в виде веб-приложения и предоставляет доступ к основным функциям через единое рабочее пространство. На главном экране отображается информация о клиентах, заказах, статусах выполнения работ и других объектах учета. Интерфейс CRM-системы LiveSkлад приведён на рисунке 1.

Новый заказ

ТИП ЗАКАЗА

Автосервис

КЛИЕНТ

Имя * Антонов Павел Геннадьевич +

Телефон +7 (910) 398-57-28 +

Email antonoffpg@mail.ru +

Telegram Добавить

ХАРАКТЕРИСТИКИ АВТО

Марка Toyota +

Модель Camri

Цвет белый

Год выпуска 2018

Госномер A123BC 77

Пробег (км) 126 540

Тип двигателя Бензин, 2,5л

Добавить фото

ОБРАЩЕНИЕ КЛИЕНТА

Тип обращения Диагностика

Жалоба клиента Машина не набирает обороты

Комплектация Ключи ✗ Сервисная книжка ✗

Состояние Легкие царапины на заднем бампере

Документы на авто ✗

Крайний срок 25.02.2026 12:00 ✗

Срочно: Да

Создать Закрыть

Печать

Рисунок 1 – Интерфейс CRM-системы LiveSklad

Преимуществом LiveSklad является широкий набор функций, близкий к задачам автосервиса: учет заявок, контроль статусов, складской учет, работа с клиентской базой и отчетность. Однако для небольшого детейлинг-центра такой сервис может быть избыточным, поскольку ориентирован не только на детейлинг, но и на более широкий круг задач автосервиса. Кроме того, LiveSklad является облачным решением, а значит, для полноценной работы требуется доступ к интернету. Использование облачной платформы также предполагает хранение данных в инфраструктуре поставщика услуги. Несмотря на то что разработчик заявляет о соответствии требованиям законодательства РФ и размещении данных на серверах в России [4], для некоторых организаций более

предпочтительным может быть локальное хранение данных внутри предприятия.

Другим аналогом является Clean-Control – CRM-система для автомоек, детейлинг-центров, шиномонтажа и СТО. Разработчик позиционирует систему как программу управления и контроля для предприятий автообслуживания. В функциональность входят учет клиентов, финансовый контроль, управление скидками и бонусами, расчет заработной платы сотрудников, аналитика, SMS-информирование, разграничение прав доступа и интеграция с кассовым оборудованием [6]. Также на сайте указана стоимость использования – 60 рублей в день или 1800 рублей в месяц [6]. Интерфейс CRM-системы Clean-Control приведён на рисунке 2.

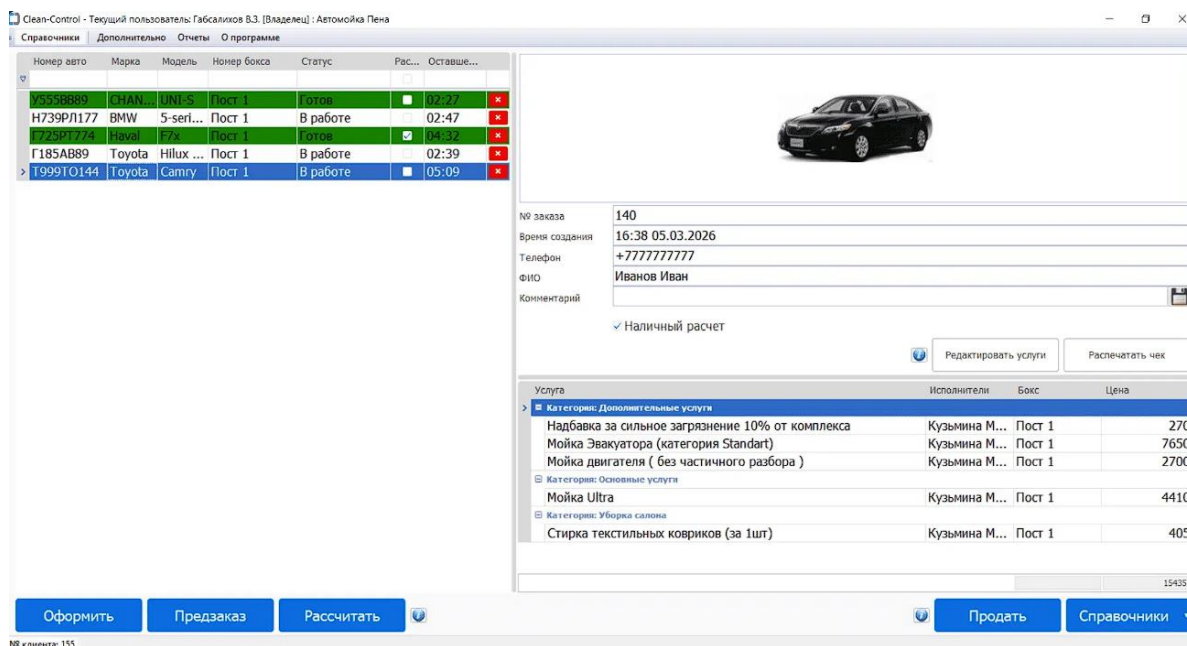


Рисунок 2 – Интерфейс CRM-системы Clean-Control

Clean-Control хорошо подходит для предприятий, работающих с большим количеством клиентских посещений, например автомоек и детейлинг-центров. Система имеет готовые инструменты контроля бизнеса, отчетности и расчета заработной платы. При этом решение является коммерческим и работает по подписочной модели. Для небольшого детейлинг-центра ежемесячная оплата может быть нежелательной, особенно если требуется автоматизировать только основные процессы: заявки, автомобили, работы, материалы, сотрудников и

отчеты. Кроме того, часть возможностей, например SMS-информирование, Telegram-боты и интеграции, может быть избыточной на начальном этапе работы организации.

Следующим рассмотренным решением является YCLIENTS – сервис онлайн-записи и автоматизации бизнеса. Для сферы автоуслуг YCLIENTS предлагает инструменты онлайн-записи, электронный журнал, учет клиентов, финансовую аналитику, складской учет, автоматические уведомления и работу с расписанием специалистов [7]. Разработчик указывает, что сервис позволяет клиентам записываться круглосуточно, вести расписание сотрудников в едином календаре и отслеживать доходы и расходы за выбранный период [7]. Интерфейс сервиса YCLIENTS приведён на рисунке 3.

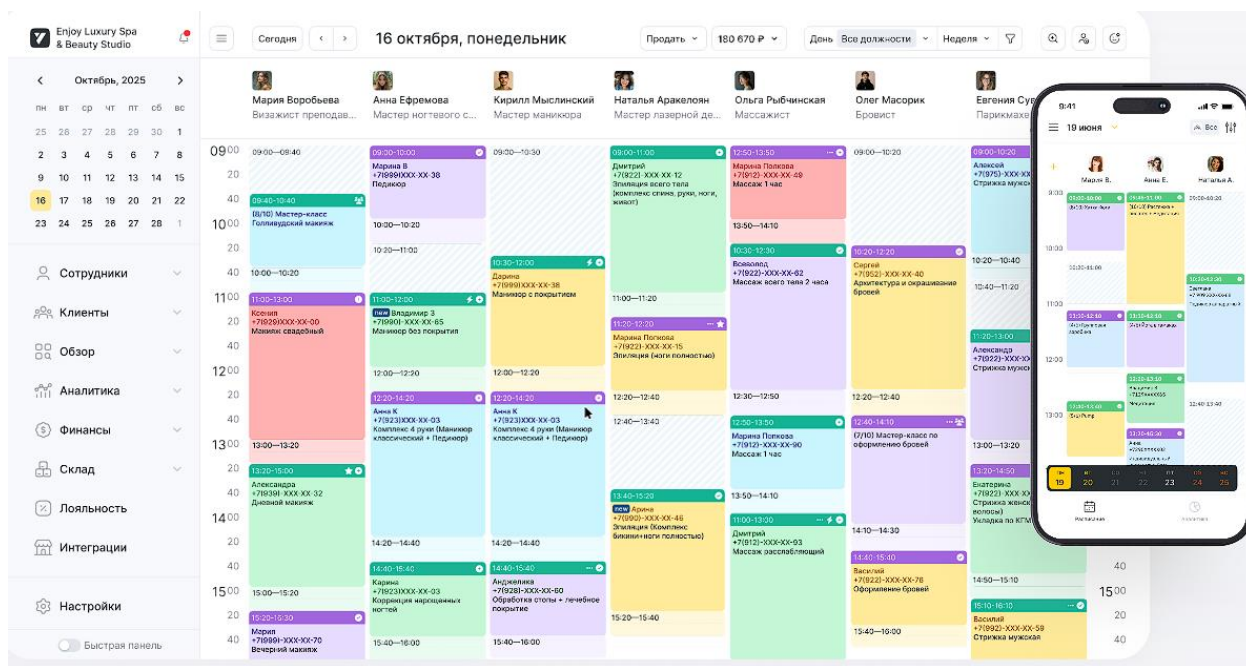


Рисунок 3 – Интерфейс сервиса YCLIENTS для автоуслуг

К преимуществам YCLIENTS можно отнести развитые инструменты онлайн-записи, клиентской базы, уведомлений и аналитики. Сервис удобен для организаций, которым важно привлекать клиентов через интернет и управлять записями через облачную платформу. Однако YCLIENTS является универсальной CRM-системой для разных сфер бизнеса, поэтому его функциональность не полностью ориентирована на учет восстановительных

работ автомобиля, материалов, статусов заказов и внутреннюю работу детейлинг-центра. Кроме того, использование сервиса связано с облачной моделью и зависимостью от интернет-соединения. Для организации, которой требуется локальная система без обязательной передачи данных во внешние сервисы, такой вариант может быть менее предпочтительным.

В качестве крупного отраслевого решения можно рассмотреть 1С:Альфа-Авто. Данная система предназначена для автоматизации автосервиса, учета запчастей, клиентов, автомобилей, калькуляции ремонта, записи на ремонт, машинозаявдов и заказ-нарядов [8]. Также решения на платформе 1С могут включать финансовый блок, складской учет, обмен данными и другие механизмы, характерные для комплексной автоматизации предприятия [8]. Интерфейс системы 1С:Альфа-Авто приведён на рисунке 4.

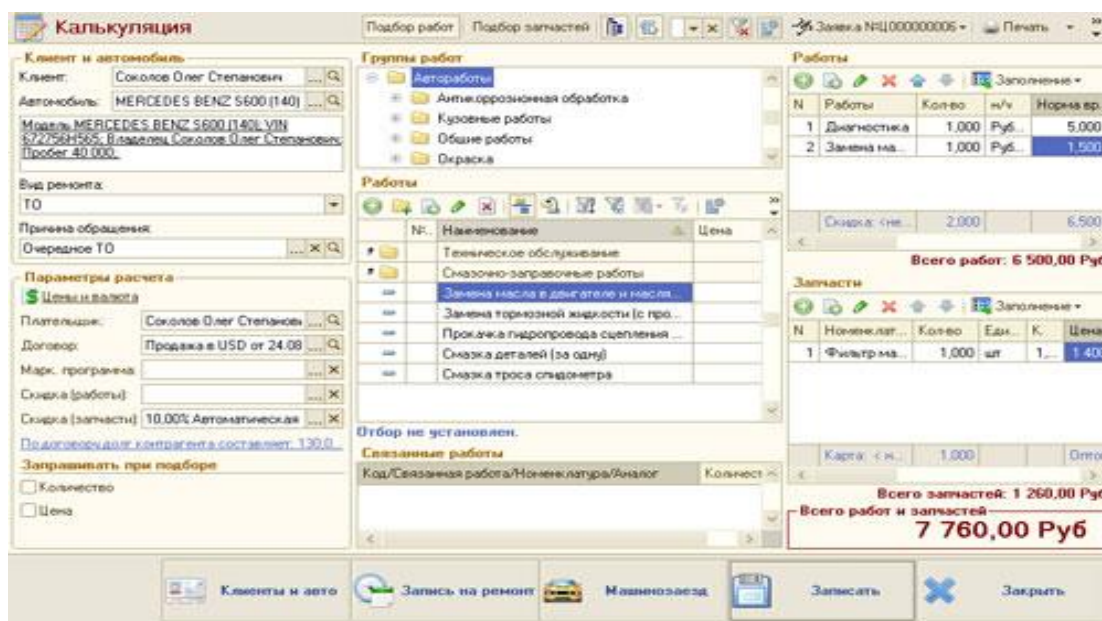


Рисунок 4 – Интерфейс системы 1С:Альфа-Авто

Преимуществом 1С:Альфа-Авто является функциональная полнота и возможность применения в организациях с большим объемом данных. Система подходит для автосервисов, автосалонов и предприятий, где требуется комплексный учет запчастей, заказ-нарядов, финансов и складских операций. Однако для небольшого детейлинг-центра такая система может оказаться слишком сложной и затратной во внедрении. Кроме того, решения на базе 1С

обычно требуют настройки, сопровождения и обучения пользователей. Для автоматизации ограниченного набора процессов детейлинга – учета заявок, работ, материалов, сотрудников и отчетности – применение крупной комплексной системы может быть экономически нецелесообразным.

Также был рассмотрен сервис CRM4Auto, представляющий собой облачную CRM для автосервисов. Система включает заказ-наряды, клиентскую базу, склад, финансы и отчеты. На сайте разработчика указано, что решение предназначено для управления автосервисом в облаке и поддерживает полный цикл работы от приемки автомобиля до финансовых отчетов [9]. Интерфейс облачной CRM-системы CRM4Auto приведён на рисунке 5.

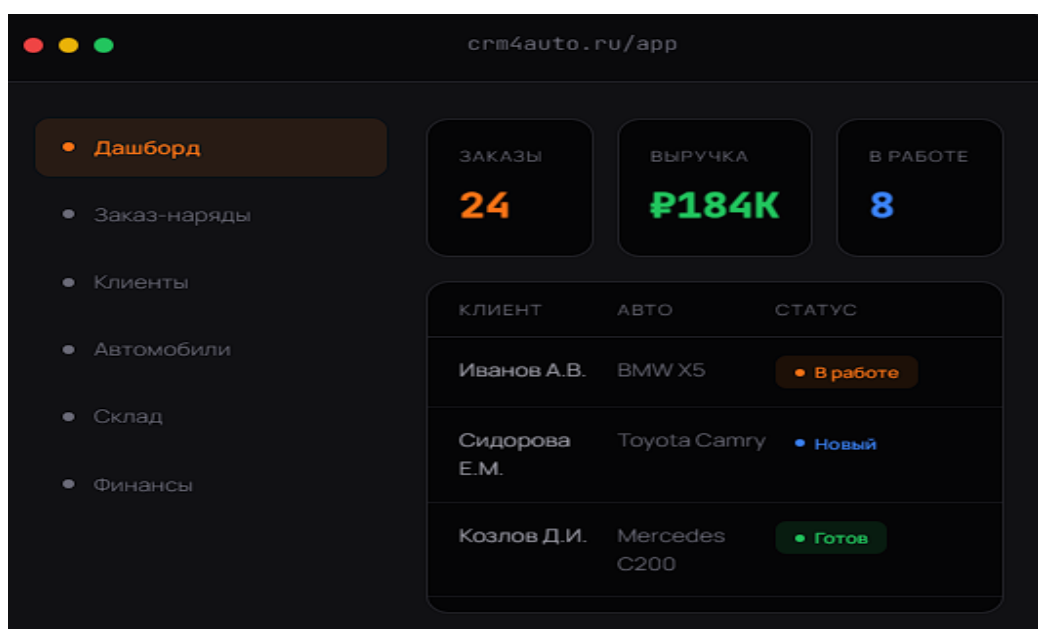


Рисунок 5 – Интерфейс облачной CRM-системы CRM4Auto

CRM4Auto имеет набор функций, близкий к задачам автосервиса: создание заказ-нарядов, отслеживание статусов, назначение механиков, ведение клиентской базы, привязка автомобилей и напоминания о техническом обслуживании [9]. Однако система также ориентирована преимущественно на автосервис, а не на специфику детейлинга и косметического восстановления автомобилей. Кроме того, разработчик прямо позиционирует сервис как облачную CRM, что означает зависимость от доступа к интернету и использование внешней инфраструктуры для работы системы.

Таблица 1 – Сравнительная таблица аналогов разрабатываемой системы

Критерий	LiveSklad	Clean-Control	YCLIENTS	1С:Альфа-Авто	CRM4Auto	Разрабатываемая система
Учет клиентов	+	+	+	+	+	+
Учет автомобилей клиентов	+	±	±	+	+	+
Создание заявок / заказ-нарядов	+	+	±	+	+	+
Учет выполняемых работ	+	+	±	+	+	+
Учет материалов и расходных средств	+	±	±	+	+	+
Контроль статусов выполнения работ	+	+	±	+	+	+
Формирование отчетности	+	+	+	+	+	+
Управление сотрудниками	+	+	+	+	+	+
Ролевая модель доступа	+	+	±	+	+	+
Возможность локальной работы без постоянного интернета	–	–	–	±	–	+
Отсутствие обязательной облачной инфраструктуры	–	–	–	±	–	+
Адаптация под процессы небольшого дотейлинг-центра	±	+	±	±	±	+
Отсутствие подписочной модели	–	–	–	±	–	+
Низкая избыточность функционала	±	+	±	–	±	+

Примечание – знак «+» означает наличие возможности, знак «–» означает отсутствие возможности в требуемом виде, знак «±» означает частичную

реализацию или зависимость от тарифа, настройки, конфигурации либо варианта внедрения.

Анализ показал, что рассмотренные программные решения позволяют автоматизировать отдельные или комплексные процессы организаций, связанных с автообслуживанием. LiveSklad, 1С:Альфа-Авто и CRM4Auto обладают широким набором функций для автосервиса, однако могут быть избыточными для небольшого детейлинг-центра. Clean-Control ближе к сфере автомоек и детейлинг-центров, но также работает как коммерческий сервис и предполагает использование подписочной модели. YCLIENTS удобен для онлайн-записи и клиентского учета, однако не ориентирован непосредственно на учет восстановительных работ, материалов и заказ-нарядов детейлинг-центра.

Разрабатываемая информационная система должна объединить необходимые функции для учета клиентов, автомобилей, заявок, заказ-нарядов, материалов, сотрудников и отчетности. При этом ее преимуществом является ориентация на локальное использование внутри организации без обязательного постоянного подключения к интернету и без передачи данных во внешнюю облачную инфраструктуру. Такой подход позволяет снизить зависимость от сторонних сервисов и адаптировать систему под конкретные процессы детейлинг-центра.

Проведенный анализ показал, что на рынке существуют различные программные решения для автоматизации автообслуживания. Они позволяют вести клиентскую базу, оформлять заказы, контролировать склад, учитывать финансы и формировать отчеты. Однако большинство рассмотренных систем являются либо универсальными CRM-сервисами, либо крупными комплексными решениями для автосервиса. Для небольшого детейлинг-центра такие системы могут иметь ряд ограничений: высокая стоимость владения, избыточный функционал, необходимость настройки, зависимость от интернет-соединения и хранение данных во внешней инфраструктуре поставщика.

Разрабатываемая информационная система должна быть ориентирована именно на процессы детейлинг-центра, включая учет клиентов, автомобилей,

заявок, восстановительных работ, материалов, сотрудников, статусов выполнения и отчетности. В отличие от облачных сервисов, предполагается возможность работы приложения в локальной инфраструктуре организации, в том числе в локальной сети без обязательного постоянного подключения к интернету. Такой подход позволяет снизить зависимость от внешних сервисов, упростить работу сотрудников и обеспечить хранение данных внутри организации.

1.3 Анализ функциональных требований

В процессе анализа предметной области были выделены функциональные требования к информационной системе детейлинг-центра. Система должна обеспечивать учет клиентских заявок, автомобилей, заказ-нарядов, услуг, материалов, сотрудников, статусов выполнения работ и отчетности.

В системе предусмотрены следующие категории пользователей:

- администратор;
- мастер;
- менеджер/приёмщик;
- склад.

Каждая категория пользователей имеет собственный набор доступных функций. Разграничение прав доступа необходимо для ограничения доступа к служебной информации, снижения вероятности ошибок и упрощения работы пользователя с системой.

К основным функциональным требованиям относятся:

- авторизация пользователя по логину и паролю;
- проверка активности учетной записи;
- разграничение доступа по ролям;
- создание и обработка клиентских заявок;
- ведение данных о клиентах и автомобилях;
- прикрепление фотографий автомобиля при создании заявки;
- перевод заявки в работу, ожидание или отклонение;

- автоматическое создание осмотра, сметы и заказ-наряда при принятии заявки в работу;
- просмотр и редактирование заказ-нарядов;
- добавление услуг и материалов в заказ-наряд;
- пересчет итоговой стоимости;
- фиксация осмотра и плановой даты завершения работ;
- прикрепление фотографий результата;
- завершение заказ-наряда;
- ведение складских остатков материалов;
- фиксация поступления и списания материалов;
- управление данными сотрудников и их ролями;
- формирование отчетов и просмотр аналитики за выбранный период.

В соответствии с указанными требованиями была построена диаграмма вариантов использования UML, отражающая взаимодействие пользователей с системой [10]. Актерами системы являются администратор, мастер, менеджер/приёмщик и сотрудник склада.

Администратор имеет доступ к основным разделам системы, включая заявки, задачи, склад, отчеты, аналитику и управление сотрудниками. Мастер работает с заказ-нарядами, добавляет услуги и материалы, фиксирует результат выполнения работ и завершает задачи. Менеджер/приёмщик оформляет клиентские заявки, вводит данные клиента и автомобиля, прикрепляет фотографии и изменяет статус заявки. Пользователь с ролью «Склад» отвечает за учет материалов, контроль остатков, поступления и списания.

В ходе анализа функциональных требований разрабатываемой системы были выделены ключевые сценарии взаимодействия пользователей с системой. Для каждой роли определен перечень доступных действий, соответствующий обязанностям пользователя в детейлинг-центре. Полученные требования являются основой для дальнейшего проектирования структуры данных, пользовательского интерфейса и программной логики информационной системы.

Диаграмма вариантов использования представлена на рисунке 6.

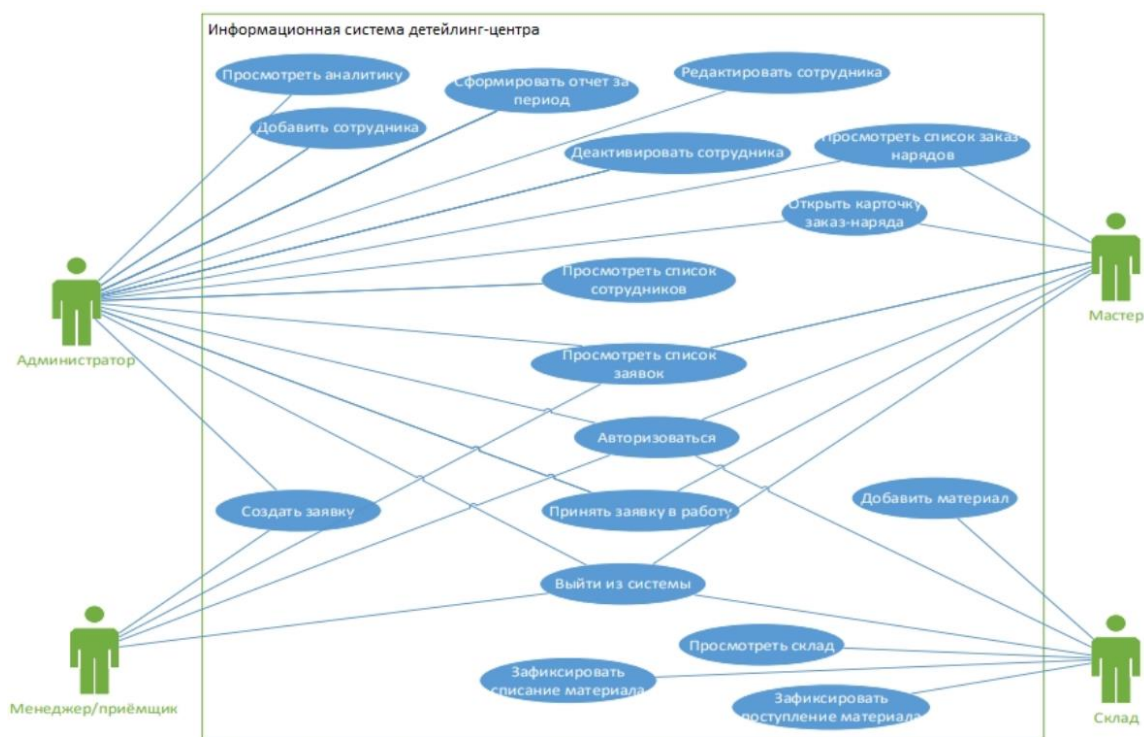


Рисунок 6 – Диаграмма вариантов использования

Спецификация варианта использования «Авторизоваться» приведена в таблице 2.

Таблица 2 – Спецификация варианта использования «Авторизоваться»

Поле	Описание
Прецедент	Авторизоваться
ID	1
Краткое описание	Пользователь выполняет вход в информационную систему детейлинг-центра
Главные актеры	Пользователь системы: администратор, мастер, менеджер/приёмщик, склад
Второстепенные актеры	Нет
Предусловие	Пользователь зарегистрирован в системе, его учетная запись активна
Основной поток	<ol style="list-style-type: none"> 1. Пользователь открывает окно авторизации. 2. Пользователь вводит логин и пароль. 3. Пользователь нажимает кнопку входа. 4. Система проверяет наличие пользователя в базе данных. 5. Система проверяет активность учетной записи. 6. Система определяет роль пользователя. 7. Система открывает главное окно приложения с учетом прав доступа пользователя.
Альтернативные потоки	<ol style="list-style-type: none"> 1. Если логин или пароль не введены, система выводит сообщение о необходимости заполнить поля. 2. Если логин или пароль указаны неверно, система выводит сообщение об ошибке авторизации. 3. Если учетная запись деактивирована, система запрещает вход и выводит соответствующее сообщение.

Продолжение таблицы 2

Поле	Описание
Постусловия	Пользователь вошел в систему и получил доступ к функциям, соответствующим его роли

Спецификация варианта использования «Создать клиентскую заявку» приведена в таблице 3.

Таблица 3 – Спецификация варианта использования «Создать клиентскую заявку»

Поле	Описание
Прецедент	Создать клиентскую заявку
ID	2
Краткое описание	Менеджер/приёмщик оформляет новую заявку клиента на выполнение работ по детейлингу
Главные актеры	Менеджер/приёмщик, администратор
Второстепенные актеры	Клиент
Предусловие	Пользователь авторизован в системе и имеет доступ к разделу заявок
Основной поток	<ol style="list-style-type: none"> 1. Пользователь открывает раздел заявок. 2. Пользователь нажимает кнопку создания новой заявки. 3. Система открывает форму создания заявки. 4. Пользователь выбирает существующего клиента или вводит данные нового клиента. 5. Пользователь выбирает существующий автомобиль клиента или вводит данные нового автомобиля. 6. Пользователь указывает описание проблемы или перечень требуемых работ. 7. Пользователь при необходимости прикрепляет фотографии автомобиля. 8. Пользователь сохраняет заявку. 9. Система сохраняет данные клиента, автомобиля и заявки в базе данных. 10. Система присваивает заявке начальный статус.
Альтернативные потоки	<ol style="list-style-type: none"> 1. Если обязательные поля не заполнены, система выводит сообщение об ошибке. 2. Если данные введены в некорректном формате, система запрещает сохранение заявки. 3. Если пользователь отменяет операцию, система закрывает форму без сохранения данных.
Постусловия	Новая клиентская заявка создана и отображается в списке заявок

Спецификация варианта использования «Принять заявку в работу» приведена в таблице 4.

Таблица 4 – Спецификация варианта использования «Принять заявку в работу»

Поле	Описание
Прецедент	Принять заявку в работу
ID	3
Краткое описание	Пользователь переводит клиентскую заявку в работу, после чего система создает связанные записи для дальнейшего выполнения заказа
Главные актеры	Менеджер/приёмщик, администратор

Продолжение таблицы 4

Поле	Описание
Второстепенные актеры	Мастер
Предусловие	Пользователь авторизован в системе, заявка существует и выбрана в списке заявок
Основной поток	<ol style="list-style-type: none"> 1. Пользователь открывает раздел заявок. 2. Пользователь выбирает нужную заявку. 3. Пользователь нажимает кнопку принятия заявки в работу. 4. Система изменяет статус заявки на «В работе». 5. Система создает осмотр автомобиля. 6. Система создает смету. 7. Система создает заказ-наряд, связанный с заявкой. 8. Система отображает сообщение об успешном выполнении операции. 9. Созданный заказ-наряд становится доступен в разделе задач.
Альтернативные потоки	<ol style="list-style-type: none"> 1. Если заявка не выбрана, система выводит сообщение о необходимости выбрать запись. 2. Если по заявке уже создан заказ-наряд, система не создает его повторно. 3. Если при сохранении данных возникает ошибка, система выводит сообщение об ошибке и отменяет операцию.
Постусловия	Заявка переведена в работу, создан заказ-наряд для выполнения работ

Спецификация варианта использования «Открыть карточку заказ-наряда» приведена в таблице 5.

Таблица 5 – Спецификация варианта использования «Открыть карточку задачи»

Поле	Описание
Прецедент	Открыть карточку задачи
ID	4
Краткое описание	Пользователь просматривает подробную информацию по выбранному заказ-наряду
Главные актеры	Администратор, мастер
Второстепенные актеры	Нет
Предусловие	Пользователь авторизован в системе, заказ-наряд существует и выбран в списке задач
Основной поток	<ol style="list-style-type: none"> 1. Пользователь открывает раздел задач. 2. Пользователь выбирает заказ-наряд в таблице. 3. Пользователь нажимает кнопку просмотра подробностей. 4. Система открывает карточку заказ-наряда. 5. Система отображает номер заказ-наряда, даты, исполнителя, клиента, автомобиль, описание заявки, статус и итоговую сумму. 6. Система отображает список услуг и материалов, связанных с заказ-нарядом. 7. Пользователь просматривает необходимые данные.
Альтернативные потоки	<ol style="list-style-type: none"> 1. Если заказ-наряд не выбран, система выводит сообщение о необходимости выбрать запись. 2. Если данные заказ-наряда не найдены, система выводит сообщение об ошибке.
Постусловия	Пользователь получил подробную информацию о заказ-наряде

Спецификация варианта использования «Добавить услугу в заказ-наряд» приведена в таблице 6.

Таблица 6 – Спецификация варианта использования «Добавить услугу в заказ-наряд»

Поле	Описание
Прецедент	Добавить услугу в заказ-наряд
ID	5
Краткое описание	Пользователь добавляет услугу, которая должна быть выполнена в рамках выбранного заказ-наряда
Главные актеры	Мастер, администратор
Второстепенные актеры	Нет
Предусловие	Пользователь авторизован, открыт существующий заказ-наряд, заказ-наряд не завершен
Основной поток	<ol style="list-style-type: none"> 1. Пользователь открывает карточку заказ-наряда. 2. Пользователь выбирает действие добавления услуги. 3. Система открывает список доступных услуг. 4. Пользователь выбирает услугу и указывает количество при необходимости. 5. Пользователь подтверждает добавление. 6. Система добавляет услугу в заказ-наряд. 7. Система пересчитывает итоговую стоимость заказ-наряда. 8. Система обновляет список позиций в карточке заказ-наряда.
Альтернативные потоки	<ol style="list-style-type: none"> 1. Если услуга не выбрана, система выводит сообщение об ошибке. 2. Если заказ-наряд завершен, система запрещает изменение его состава. 3. Если при добавлении услуги возникает ошибка, система выводит сообщение об ошибке.
Постусловия	Услуга добавлена в заказ-наряд, итоговая сумма обновлена

Спецификация варианта использования «Добавить материал в заказ-наряд» приведена в таблице 7.

Таблица 7 – Спецификация варианта использования «Добавить материал в заказ-наряд»

Поле	Описание
Прецедент	Добавить материал в заказ-наряд
ID	6
Краткое описание	Пользователь добавляет материал или расходное средство, используемое при выполнении работ
Главные актеры	Мастер, администратор
Второстепенные актеры	Склад
Предусловие	Пользователь авторизован, открыт существующий заказ-наряд, в системе есть доступные материалы

Продолжение таблицы 7

Поле	Описание
Основной поток	<ol style="list-style-type: none"> 1. Пользователь открывает карточку заказ-наряда. 2. Пользователь выбирает действие добавления материала. 3. Система открывает список материалов. 4. Пользователь выбирает материал и указывает количество. 5. Система проверяет наличие достаточного количества материала на складе. 6. Пользователь подтверждает добавление. 7. Система добавляет материал в заказ-наряд. 8. Система списывает указанное количество материала со склада. 9. Система пересчитывает итоговую стоимость заказ-наряда.
Альтернативные потоки	<ol style="list-style-type: none"> 1. Если материал не выбран, система выводит сообщение об ошибке. 2. Если указанное количество превышает остаток на складе, система запрещает добавление материала. 3. Если заказ-наряд завершен, система запрещает изменение состава материалов. 4. Если пользователь отменяет операцию, материал не добавляется.
Постусловия	Материал добавлен в заказ-наряд, складской остаток и итоговая сумма обновлены

В ходе анализа функциональных требований разрабатываемой системы были выделены ключевые сценарии взаимодействия пользователей с системой. Для каждой роли определен перечень доступных функций, соответствующий обязанностям пользователя в детейлинг-центре. Для наиболее важных сценариев были приведены спецификации, которые отображают действия пользователя и системы. Полученные функциональные требования являются основой для дальнейшего проектирования структуры базы данных, пользовательского интерфейса и программной логики информационной системы.

1.4 Анализ нефункциональных требований

Нефункциональные требования определяют качественные характеристики работы информационной системы: надежность, безопасность, производительность, удобство эксплуатации и возможность дальнейшего развития программного средства. Для информационной системы детейлинг-центра были выделены требования к производительности, безопасности, надежности, удобству использования, локальности работы, целостности данных, модульности, сопровождаемости и техническим средствам реализации.

Система должна обеспечивать быструю реакцию на действия пользователя при авторизации, загрузке списков заявок, заказ-нарядов, материалов, сотрудников и формировании отчетов. Работа с табличными данными должна выполняться без существенных задержек даже при увеличении количества клиентов, автомобилей и заказов.

Безопасность системы обеспечивается авторизацией пользователей по логину и паролю, проверкой активности учетной записи и разграничением доступа по ролям. Так как система обрабатывает персональные данные клиентов и сотрудников, доступ к сведениям должен предоставляться только авторизованным пользователям в соответствии с их должностными обязанностями [3].

Надежность системы предусматривает корректную обработку пользовательских ошибок. При создании заявки, добавлении клиента, автомобиля, материала или формировании отчета приложение должно проверять заполнение обязательных полей и выводить понятные сообщения об ошибках. Операции, связанные с изменением нескольких записей базы данных, должны выполняться согласованно, чтобы исключить частичное сохранение данных.

Интерфейс приложения должен быть понятным для сотрудников детейлинг-центра и не требовать длительного обучения. Основные разделы системы должны иметь логичную структуру: заявки, задачи, склад, анализ, отчеты и сотрудники. Для удобства работы сведения отображаются в табличном виде, а названия кнопок и форм соответствуют предметной области.

Система должна предусматривать возможность эксплуатации внутри организации без обязательного постоянного подключения к интернету. Размещение базы данных в локальной инфраструктуре позволяет сотрудникам работать с единой базой данных и не передавать сведения о клиентах, сотрудниках и заказах во внешние облачные сервисы.

Целостность данных обеспечивается связями между объектами предметной области. Клиентская заявка должна быть связана с клиентом и автомобилем, заказ-наряд с заявкой, исполнителем, услугами, материалами и

статусом выполнения. Использование единой базы данных снижает риск дублирования сведений и ошибок при расчете стоимости или формировании отчетов.

Модульность и сопровождаемость системы достигаются за счет разделения приложения на отдельные функциональные части: авторизацию, главное окно, работу с заявками, задачами, складом, сотрудниками и отчетностью. Такой подход позволяет дорабатывать отдельные модули без полной переработки приложения.

Информационная система реализуется как настольное приложение для Windows на языке C# с использованием технологии Windows Forms [12, 13]. Для хранения данных применяется Microsoft SQL Server, управление базой данных выполняется через SQL Server Management Studio [15, 17]. В качестве среды разработки используется Microsoft Visual Studio [18]. Минимальный состав технических средств включает персональный компьютер или ноутбук сотрудника, установленное приложение, базу данных Microsoft SQL Server и локальную сеть при необходимости совместной работы нескольких рабочих мест.

Сформулированные нефункциональные требования позволяют создать надежное, удобное и расширяемое приложение, пригодное для эксплуатации в условиях детейлинг-центра.

1.5 Планирование разработки и оценка бюджета

Планирование разработки и оценка бюджета являются важными этапами при создании информационной системы детейлинг-центра. На данном этапе необходимо определить последовательность выполнения работ, продолжительность каждого этапа, состав задействованных ресурсов и ориентировочную стоимость разработки. Такое планирование позволяет заранее оценить объем работ, распределить задачи между исполнителями и обеспечить выполнение проекта в установленные сроки.

Разработка информационной системы детейлинг-центра была разделена на четыре основных этапа:

- 1) анализ предметной области и постановка задачи;
- 2) проектирование информационной системы;
- 3) реализация информационной системы;
- 4) контроль качества информационной системы.

Первый этап – анализ предметной области и постановка задачи – включает изучение специфики работы детейлинг-центра, анализ существующих информационных систем аналогичного назначения, выявление проблем ручного учета и определение требований к разрабатываемой системе. На данном этапе формируются функциональные и нефункциональные требования, определяется состав пользователей системы, а также основные процессы, которые должны быть автоматизированы.

В рамках первого этапа выполняются следующие задачи:

- исследование предметной области;
- анализ информационных систем аналогичного назначения;
- анализ функциональных требований;
- анализ нефункциональных требований;
- определение технических средств реализации требований.

Второй этап – проектирование информационной системы – направлен на разработку структуры будущего программного средства. На данном этапе определяется архитектура приложения, проектируется структура базы данных, описываются основные сущности предметной области и связи между ними. Также выполняется проектирование пользовательского интерфейса и логики работы приложения.

В рамках второго этапа выполняются следующие задачи:

- определение архитектуры приложения;
- проектирование структуры базы данных;
- проектирование программных средств;
- проектирование пользовательского интерфейса.

Третий этап – реализация информационной системы – включает непосредственную разработку программного продукта. На данном этапе создается база данных, реализуются формы приложения, программная логика обработки заявок, заказ-нарядов, материалов, сотрудников и отчетов. Также выполняется подключение приложения к базе данных и реализация основных пользовательских сценариев.

В рамках третьего этапа выполняются следующие задачи:

- реализация базы данных;
- реализация формы авторизации;
- реализация главной формы приложения;
- реализация формы создания клиентской заявки;
- реализация формы карточки заказ-наряда;
- реализация складского учета материалов;
- реализация формы управления сотрудниками;
- реализация формирования отчетов;
- настройка взаимодействия приложения с базой данных.

Четвертый этап – контроль качества информационной системы – включает проверку работоспособности приложения и выявление возможных ошибок. На данном этапе выполняется тестирование основных функций системы: авторизации, создания заявок, изменения статусов, работы с заказ-нарядами, добавления услуг и материалов, складского учета, управления сотрудниками и формирования отчетов. По результатам тестирования выполняется исправление выявленных ошибок и проверка корректности работы системы.

В рамках четвертого этапа выполняются следующие задачи:

- выбор подхода к тестированию;
- тестирование авторизации пользователей;
- тестирование работы с клиентскими заявками;
- тестирование работы с заказ-нарядами;
- тестирование складского учета;
- тестирование управления сотрудниками;

- тестирование формирования отчетов;
- исправление выявленных ошибок.

На рисунке 7 приведено описание этапов разработки системы.

	Ⓐ	Название	Продолжи...	Начало	Окончание	Предшествующие
1		Начало проекта	0 дней	11.05.2026, 08:00	11.05.2026, 08:00	
2		Анализ предметной обл	5 дней	11.05.2026, 08:00	15.05.2026, 17:00	
3		Исследование предметно	1 день	11.05.2026, 08:00	11.05.2026, 17:00	1
4		Анализ информационных	1 день	12.05.2026, 08:00	12.05.2026, 17:00	3
5		Анализ функциональных	1 день	13.05.2026, 08:00	13.05.2026, 17:00	4
6		Анализ нефункциональн	1 день	14.05.2026, 08:00	14.05.2026, 17:00	5
7		Определение технически	1 день	15.05.2026, 08:00	15.05.2026, 17:00	6
8		Проектирование инфор	6 дней	18.05.2026, 08:00	25.05.2026, 17:00	
9		Определение архитекту	1 день	18.05.2026, 08:00	18.05.2026, 17:00	7
10		Проектирование структу	2 дней	19.05.2026, 08:00	20.05.2026, 17:00	9
11		Проектирование програм	2 дней	21.05.2026, 08:00	22.05.2026, 17:00	10
12		Проектирование пользо	1 день	25.05.2026, 08:00	25.05.2026, 17:00	11
13		Реализация информаци	16 дней	26.05.2026, 08:00	16.06.2026, 17:00	
14		Реализация базы данных	2 дней	26.05.2026, 08:00	27.05.2026, 17:00	12
15		Реализация формы автор	1 день	28.05.2026, 08:00	28.05.2026, 17:00	14
16		Реализация главной форм	2 дней	29.05.2026, 08:00	01.06.2026, 17:00	15
17		Реализация формы созда	2 дней	02.06.2026, 08:00	03.06.2026, 17:00	16
18		Реализация карточки зак	3 дней	04.06.2026, 08:00	08.06.2026, 17:00	17
19		Реализация складского у	2 дней	09.06.2026, 08:00	10.06.2026, 17:00	18
20		Реализация управления с	2 дней	11.06.2026, 08:00	12.06.2026, 17:00	19
21		Реализация отчетности	2 дней	15.06.2026, 08:00	16.06.2026, 17:00	20
22		Контроль качества инф	6 дней	17.06.2026, 08:00	24.06.2026, 17:00	
23		Выбор подхода к тестир	1 день	17.06.2026, 08:00	17.06.2026, 17:00	21
24		Тестирование авторизац	1 день	18.06.2026, 08:00	18.06.2026, 17:00	23
25		Тестирование работы с з	1 день	19.06.2026, 08:00	19.06.2026, 17:00	24
26		Тестирование работы с з	1 день	22.06.2026, 08:00	22.06.2026, 17:00	25
27		Тестирование складского	1 день	23.06.2026, 08:00	23.06.2026, 17:00	26
28		Тестирование отчетности	1 день	24.06.2026, 08:00	24.06.2026, 17:00	27
29		Конец проекта	0 дней	24.06.2026, 17:00	24.06.2026, 17:00	28

Рисунок 7 – Описание этапов разработки информационной системы детейлинг-центра

Для наглядного представления последовательности выполнения работ и длительности каждого этапа используется диаграмма Ганта. Она позволяет отразить временные интервалы выполнения задач, определить зависимость между этапами и оценить общую продолжительность разработки.

Диаграмма Ганта с этапами разработки информационной системы представлена на рисунке 8.

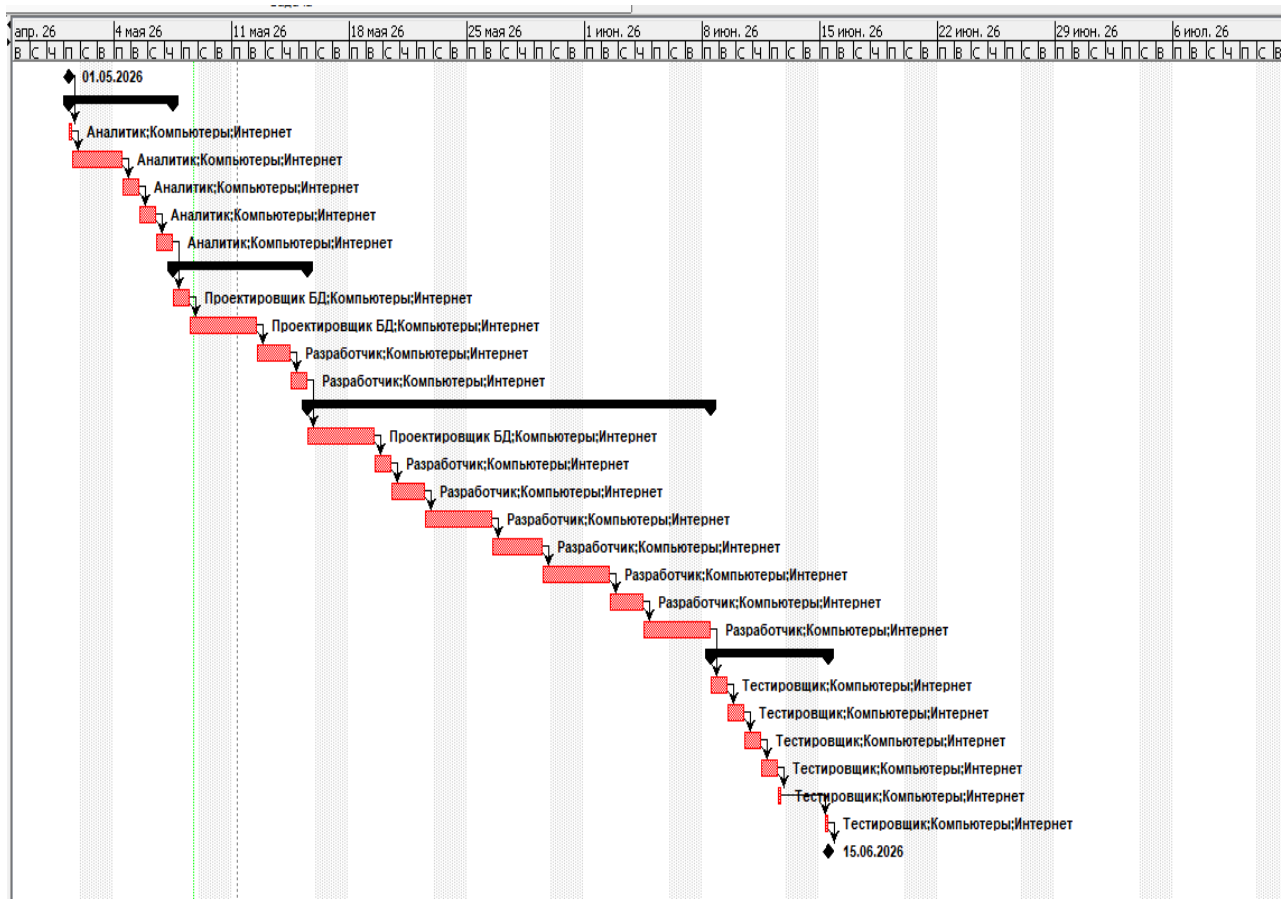


Рисунок 8 – Диаграмма Ганта с этапами разработки информационной системы

Работы над проектом были запланированы в рамках периода прохождения преддипломной практики. Общая продолжительность разработки определяется с учетом времени, необходимого для анализа предметной области, проектирования базы данных и интерфейса, реализации основных функций приложения, тестирования и подготовки отчетной документации. Каждый этап состоит из нескольких задач, выполнение которых необходимо для перехода к следующему этапу.

Для расчета бюджета проекта были определены основные ресурсы, задействованные при разработке информационной системы. К ним относятся аналитик, проектировщик базы данных, разработчик, тестировщик, а также технические ресурсы, необходимые для выполнения работ. Стоимость рабочего времени ресурсов задается в соответствии с их ролью в проекте. На рисунке 9 приведена стоимость рабочего времени задействованных ресурсов.

		Название	Стандартная ставка
1		Аналитик	0/час
2		Проектировщик БД	0/час
3		Разработчик	0/час
4		Тестировщик	0/час
5		Компьютеры	0
6		Интернет	0

Рисунок 9 – Стоимость ресурсов проекта

При расчете бюджета учитывается продолжительность выполнения задач и стоимость рабочего времени каждого ресурса. Основные затраты связаны с анализом предметной области, проектированием структуры базы данных, реализацией приложения и тестированием. Также учитываются трудозатраты на подготовку документации и оформление проектных материалов. Общие затраты проекта с учетом продолжительности задач представлены на рисунке 10.

Название	Продолжи...	Начало	Окончание	Стоимость
Начало проекта	0 дней	01.05.2026, 08:00	01.05.2026, 08:00	0,00 Р
☒ Анализ предметной обл	4,5 дней	01.05.2026, 08:00	07.05.2026, 13:00	18450,00 Р
Исследование предметно	0,5 дней	01.05.2026, 08:00	01.05.2026, 13:00	2250,00 Р
Анализ информационных	1 день	01.05.2026, 13:00	04.05.2026, 13:00	4050,00 Р
Анализ функциональных	1 день	04.05.2026, 13:00	05.05.2026, 13:00	4050,00 Р
Анализ нефункциональн	1 день	05.05.2026, 13:00	06.05.2026, 13:00	4050,00 Р
Определение технически	1 день	06.05.2026, 13:00	07.05.2026, 13:00	4050,00 Р
☒ Проектирование инфор	6 дней	07.05.2026, 13:00	15.05.2026, 13:00	24700,00 Р
Определение архитектур	1 день	07.05.2026, 13:00	08.05.2026, 13:00	3600,00 Р
Проектирование структу	2 дней	08.05.2026, 13:00	12.05.2026, 13:00	6800,00 Р
Проектирование програм	2 дней	12.05.2026, 13:00	14.05.2026, 13:00	9350,00 Р
Проектирование пользо	1 день	14.05.2026, 13:00	15.05.2026, 13:00	4950,00 Р
☒ Реализация информаци	16 дней	15.05.2026, 13:00	08.06.2026, 13:00	72250,00 Р
Реализация базы данных	2 дней	15.05.2026, 13:00	19.05.2026, 13:00	6800,00 Р
Реализация формы автор	1 день	19.05.2026, 13:00	20.05.2026, 13:00	4950,00 Р
Реализация главной форм	2 дней	20.05.2026, 13:00	22.05.2026, 13:00	9350,00 Р
Реализация формы созда	2 дней	22.05.2026, 13:00	26.05.2026, 13:00	9350,00 Р
Реализация карточки зак	3 дней	26.05.2026, 13:00	29.05.2026, 13:00	13750,00 Р
Реализация складского у	2 дней	29.05.2026, 13:00	02.06.2026, 13:00	9350,00 Р
Реализация управления с	2 дней	02.06.2026, 13:00	04.06.2026, 13:00	9350,00 Р
Реализация отчетности	2 дней	04.06.2026, 13:00	08.06.2026, 13:00	9350,00 Р
☒ Контроль качества инф	5 дней	08.06.2026, 13:00	15.06.2026, 13:00	20700,00 Р
Выбор подхода к тестир	1 день	08.06.2026, 13:00	09.06.2026, 13:00	4050,00 Р
Тестирование авторизац	1 день	09.06.2026, 13:00	10.06.2026, 13:00	4050,00 Р
Тестирование работы с з	1 день	10.06.2026, 13:00	11.06.2026, 13:00	4050,00 Р
Тестирование работы с з	1 день	11.06.2026, 13:00	12.06.2026, 13:00	4050,00 Р
Тестирование складског	0,5 дней	12.06.2026, 13:00	12.06.2026, 17:00	2250,00 Р
Тестирование отчетности	0,5 дней	15.06.2026, 08:00	15.06.2026, 13:00	2250,00 Р
Конец проекта	0 дней	15.06.2026, 13:00	15.06.2026, 13:00	0,00 Р

Рисунок 10 – Общие затраты проекта

Итоговая стоимость проекта определяется как сумма затрат по всем этапам разработки. Распределение затрат по этапам может быть представлено следующим образом:

- «Анализ предметной области и постановка задачи» – 18 450 руб.;
- «Проектирование информационной системы» – 24 700 руб.;
- «Реализация информационной системы» – 72 250 руб.;
- «Контроль качества информационной системы» – 20 700 руб.;
- итоговая стоимость проекта составляет 136 100 руб.

Проведенное планирование позволило определить основные этапы разработки информационной системы детейлинг-центра, распределить задачи по времени, оценить необходимые ресурсы и рассчитать ориентировочную стоимость проекта. Такой подход обеспечивает более организованное выполнение работ и снижает риск нарушения сроков разработки.

2 Проектирование программных средств

2.1 Архитектура приложения

Основной целью проектирования является разработка архитектуры информационной системы, предназначенной для автоматизации деятельности детейлинг-центра. Проектируемое приложение должно обеспечивать удобную работу с клиентскими заявками, автомобилями, заказ-нарядами, услугами, материалами, сотрудниками и отчетностью.

Разрабатываемая система представляет собой настольное приложение с графическим пользовательским интерфейсом. Для реализации интерфейса используется технология Windows Forms, а разработка программной логики выполняется на языке C#[12, 13]. Хранение данных осуществляется в базе данных Microsoft SQL Server. Такой вариант реализации подходит для локального использования внутри организации и позволяет работать с данными без обязательного применения внешних облачных сервисов.

Архитектура приложения строится на взаимодействии трех основных частей: пользовательского интерфейса, программной логики и базы данных. Пользователь взаимодействует с системой через экранные формы приложения. Программная логика обрабатывает действия пользователя, выполняет проверки, изменяет статусы записей, рассчитывает итоговую стоимость работ и формирует запросы к базе данных. База данных обеспечивает централизованное хранение информации, необходимой для работы детейлинг-центра.

В составе архитектуры можно выделить следующие уровни:

- слой представления;
- слой прикладной логики;
- слой доступа к данным;
- база данных.

Слой представления отвечает за взаимодействие пользователя с приложением. К данному слою относятся формы Windows Forms, через которые сотрудник выполняет основные операции. В системе предусмотрены форма

авторизации, главное окно приложения, форма создания клиентской заявки, форма выбора деталей и материалов, форма создания материала, форма карточки заказ-наряда, форма управления сотрудниками и форма формирования отчета за выбранный период.

Слой прикладной логики отвечает за выполнение основных правил работы системы. На этом уровне происходит проверка введенных данных, определение роли пользователя, ограничение доступа к отдельным разделам приложения, создание заявок, изменение их статусов, формирование заказ-нарядов, добавление услуг и материалов, расчет итоговой стоимости работ, обработка складских операций и подготовка данных для отчетов.

Слой доступа к данным обеспечивает взаимодействие приложения с Microsoft SQL Server. Он выполняет операции чтения, добавления, изменения и удаления данных. Через данный слой приложение получает сведения о пользователях, клиентах, автомобилях, заявках, заказ-нарядах, услугах, материалах, сотрудниках и отчетных данных.

База данных предназначена для хранения структурированной информации предметной области. В ней размещаются данные о пользователях системы, ролях, клиентах, автомобилях, заявках, заказ-нарядах, услугах, материалах, складских остатках, сотрудниках и результатах выполненных работ. Использование единой базы данных позволяет связать между собой основные объекты системы и обеспечить согласованность информации.

Основными компонентами информационной системы являются:

- модуль авторизации, который обеспечивает вход пользователя в систему по логину и паролю, проверку активности учетной записи и определение роли пользователя;
- модуль главного окна, который предоставляет доступ к основным разделам приложения и отображает функции в зависимости от роли авторизованного пользователя;

- модуль клиентских заявок, который используется для создания новой заявки, выбора или добавления клиента, указания автомобиля, описания проблемы и сохранения данных в базе;

- модуль заказ-нарядов, который обеспечивает просмотр задач, открытие карточки выбранного заказ-наряда, добавление услуг и материалов, изменение статуса и завершение работ;

- модуль складского учета, который предназначен для ведения материалов, фиксации поступлений и списаний, а также контроля складских остатков;

- модуль управления сотрудниками, который обеспечивает просмотр, добавление, редактирование и деактивацию учетных записей сотрудников;

- модуль отчетности, который формирует отчет по выполненным заказам за выбранный период;

- модуль взаимодействия с базой данных, который выполняет SQL-запросы и передает данные между приложением и Microsoft SQL Server.

Начальной точкой работы с системой является форма авторизации. Пользователь вводит логин и пароль, после чего приложение проверяет учетные данные в базе данных. При успешной авторизации система определяет роль сотрудника и открывает главное окно приложения. Если учетная запись неактивна или данные введены неверно, пользователю выводится сообщение об ошибке.

Главное окно приложения выполняет роль основного средства навигации. Через него пользователь получает доступ к разделам заявок, задач, склада, отчетов, анализа и сотрудников. Набор доступных разделов зависит от роли пользователя. Администратору доступны основные функции управления системой, менеджер или приемщик работает с клиентскими заявками, мастер работает с заказ-нарядами, а сотрудник склада выполняет операции с материалами.

Модуль клиентских заявок предназначен для первичной обработки обращения клиента. При создании заявки пользователь может выбрать существующего клиента или добавить нового, указать автомобиль, описать требуемые работы и сохранить заявку. После принятия заявки в работу система создает связанные данные, необходимые для дальнейшего выполнения заказа.

Модуль заказ-нарядов используется для сопровождения выполнения работ. В карточке заказ-наряда отображаются сведения о клиенте, автомобиле, описании заявки, исполнителе, статусе и итоговой стоимости. Пользователь может добавлять услуги и материалы, удалять ошибочно добавленные позиции, фиксировать результат выполнения работ и завершать заказ. При изменении состава работ или материалов итоговая стоимость пересчитывается.

Модуль складского учета предназначен для контроля материалов, используемых при выполнении работ по детейлингу. Пользователь может просматривать список материалов, добавлять новые позиции, фиксировать поступление и списание материалов. Наличие складского учета позволяет контролировать расходные средства и своевременно выявлять позиции с недостаточным остатком.

Модуль управления сотрудниками используется для ведения учетных записей работников детейлинг-центра. В данном разделе можно просматривать список сотрудников, добавлять новые записи, редактировать данные, назначать роль и деактивировать учетные записи. Это необходимо для разграничения доступа к функциям приложения.

Модуль отчетности предназначен для получения сведений о выполненных заказах за выбранный период. Пользователь указывает начальную и конечную дату, после чего система выбирает данные из базы и формирует отчет. В отчет включаются сведения о заказах, клиентах, автомобилях, количестве услуг и материалов, итоговых суммах и сроках выполнения работ.

Разрабатываемая информационная система ориентирована на эксплуатацию внутри детейлинг-центра. Приложение может быть установлено на рабочем месте сотрудника, а база данных может размещаться на том же

компьютере или на отдельном сервере в локальной сети организации. Такой вариант архитектуры позволяет нескольким сотрудникам работать с единой базой данных и сохранять контроль над информацией внутри организации.

Архитектура информационной системы детейлинг-центра представлена на рисунке 11. На схеме показано взаимодействие пользователя, интерфейса Windows Forms, прикладной логики, слоя доступа к данным, базы данных Microsoft SQL Server и файловой системы, используемой для хранения фотографий.

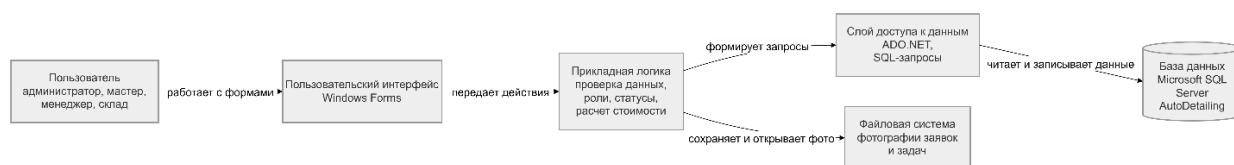


Рисунок 11 – Архитектура информационной системы детейлинг-центра

Разработанная архитектура позволяет разделить функции приложения на отдельные компоненты, упростить сопровождение программного средства и обеспечить возможность дальнейшего расширения системы. При необходимости приложение может быть дополнено печатью документов, расширенной аналитикой, историей обслуживания автомобилей и дополнительными отчетами.

2.2 Проектирование структур данных

2.2.1 Концептуальная модель базы данных

Концептуальная модель представляет собой модель, отображающую основные данные предметной области и связи между ними без учета особенностей физической реализации в конкретной системе управления базами данных.

Для разработки концептуальной модели базы данных информационной системы детейлинг-центра необходимо выделить основные сущности предметной области. Система должна обеспечивать хранение информации о клиентах, автомобилях, заявках, осмотрах, сметах, заказ-нарядах, услугах,

материалах, сотрудниках, ролях пользователей, статусах выполнения работ, фотографиях и складских операциях.

Описание сущностей предметной области представлено в таблице 12.

Таблица 12 – Сущности базы данных

Название сущности	Идентификатор сущности	Назначение сущности
Клиент	Client	Содержит сведения о клиентах детейлинг-центра
Автомобиль	Vehicle	Содержит сведения об автомобилях клиентов
Заявка	Request	Содержит сведения о клиентских заявках на выполнение работ
Фотография заявки	RequestPhoto	Содержит сведения о фотографиях автомобиля, прикрепленных к заявке
История статусов заявки	RequestStatusHistory	Содержит сведения об изменении статусов клиентской заявки
Осмотр	Inspection	Содержит сведения об осмотре автомобиля перед выполнением работ
История статусов осмотра	InspectionStatusHistory	Содержит сведения об изменении статусов осмотра
Смета	Estimate	Содержит сведения о предварительном расчете стоимости работ
Услуга в смете	EstimateServiceItem	Содержит сведения об услугах, включенных в смету
Материал в смете	EstimateMaterialItem	Содержит сведения о материалах, включенных в смету
Заказ-наряд	WorkOrder	Содержит сведения о заказе на выполнение работ по заявке
Услуга заказ-наряда	WorkOrderService	Содержит сведения об услугах, включенных в заказ-наряд
Материал заказ-наряда	WorkOrderMaterial	Содержит сведения о материалах, использованных в заказ-наряде
Фотография заказ-наряда	WorkOrderPhoto	Содержит сведения о фотографиях результата выполненных работ
История статусов заказ-наряда	WorkOrderStatusHistory	Содержит сведения об изменении статусов заказ-наряда
Материал	MaterialCatalog	Содержит сведения о материалах и складских остатках
Услуга	ServiceCatalog	Содержит сведения об услугах детейлинг-центра
Складское движение	StockMovement	Содержит сведения о поступлении и списании материалов
Пользователь	User	Содержит сведения об учетных записях сотрудников
Складское движение	StockMovement	Содержит сведения о поступлении и списании материалов
Пользователь	User	Содержит сведения об учетных записях сотрудников

Продолжение таблицы 12

Название сущности	Идентификатор сущности	Назначение сущности
Роль	Role	Содержит сведения о ролях пользователей системы
Статус	Status	Содержит справочник статусов заявок, осмотров и заказ-нарядов
Область кузова	BodyArea	Содержит справочник областей автомобиля для привязки фотографий
Временной интервал	ScheduleSlot	Содержит сведения о планируемом времени выполнения работ

Основными сущностями предметной области являются клиент, автомобиль, заявка и заказ-наряд. Клиент может иметь один или несколько автомобилей. Каждый автомобиль связан с конкретным клиентом. Клиентская заявка создается для определенного клиента и автомобиля, содержит описание проблемы или требуемых работ, дату создания, приоритет и текущий статус.

После создания заявки может выполняться осмотр автомобиля. Осмотр связан с заявкой и сотрудником, который его проводит. На основании осмотра формируется смета, включающая перечень услуг и материалов. Для хранения состава сметы используются отдельные сущности услуг и материалов сметы, что позволяет включать в один расчет несколько позиций.

Заказ-наряд отражает выполнение работ по заявке. Он связан с заявкой, сметой, ответственным сотрудником и текущим статусом. В состав заказ-наряда входят услуги и материалы, которые хранятся в отдельных сущностях. Такой подход позволяет учитывать несколько услуг и несколько материалов в рамках одного заказа, а также выполнять расчет итоговой стоимости работ.

Для контроля выполнения работ в модели предусмотрены сущности истории статусов. Они позволяют фиксировать изменение состояния заявки, осмотра и заказ-наряда, дату изменения, комментарий и пользователя, выполнившего изменение. Это необходимо для отслеживания этапов выполнения заказа и контроля действий сотрудников.

Для учета материалов используется сущность материалов, содержащая сведения о названии, единице измерения, цене, текущем остатке и минимальном остатке. Складские операции фиксируются в сущности складского движения.

Это позволяет учитывать поступление и списание материалов, используемых при выполнении работ.

Графическое представление концептуальной модели базы данных представлено на рисунке 12.

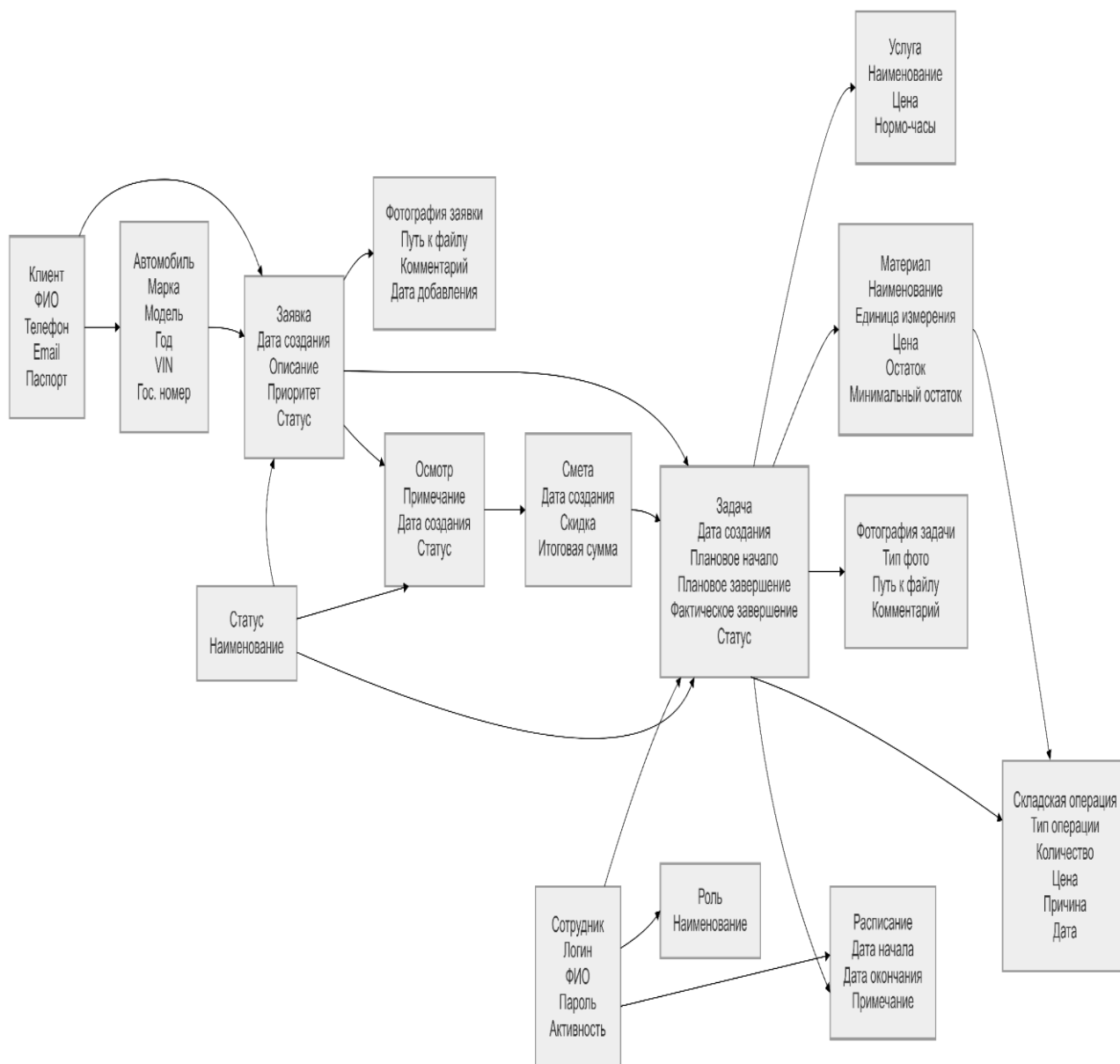


Рисунок 12 – Концептуальная модель базы данных информационной системы
детейлинг-центра

Для разграничения доступа используются сущности пользователя и роли. Пользователь связан с ролью, которая определяет набор доступных функций в системе. Такой подход позволяет ограничить доступ к отдельным разделам приложения в зависимости от должностных обязанностей сотрудника.

2.2.2 Логическая модель базы данных

Логическая модель данных представляет собой более детальную и технически проработанную версию концептуальной модели. В ней осуществляется описание таблиц, атрибутов, первичных и внешних ключей, а также взаимосвязей между таблицами. Логическая модель служит основой для последующей физической реализации базы данных в Microsoft SQL Server.

Логическая модель базы данных представлена на рисунке 13.

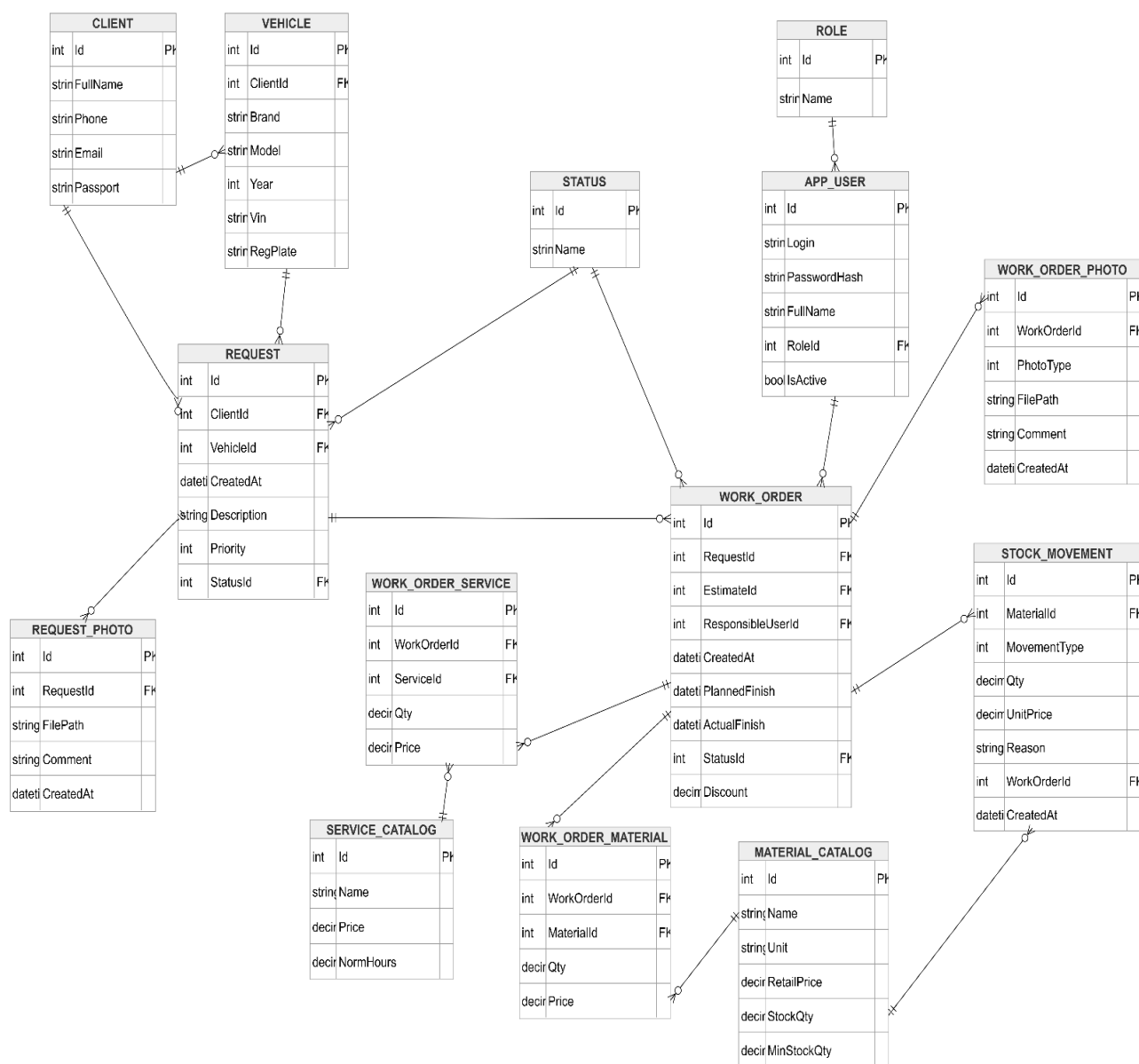


Рисунок 13 – Логическая модель базы данных информационной системы детейлинг-центра

Разработанная логическая модель отражает структуру предметной области детейлинг-центра и обеспечивает хранение данных, необходимых для работы информационной системы. В базе данных предусмотрены таблицы для учета клиентов, автомобилей, заявок, осмотров, смет, заказ-нарядов, услуг, материалов, сотрудников, ролей, статусов и складских операций.

Основной таблицей для хранения сведений о клиентах является Client. Для хранения автомобилей используется таблица Vehicle, связанная с таблицей клиентов. Клиентская заявка хранится в таблице Request и связана с клиентом, автомобилем и текущим статусом. Фотографии, прикрепленные к заявке, хранятся в таблице RequestPhoto, а изменения статусов заявки фиксируются в таблице RequestStatusHistory.

Данные осмотра автомобиля хранятся в таблице Inspection. Она связана с заявкой, пользователем, выполняющим осмотр, и текущим статусом. История изменения статусов осмотра хранится в таблице InspectionStatusHistory.

Смета хранится в таблице Estimate и связана с осмотром. Состав сметы представлен таблицами EstimateServiceItem и EstimateMaterialItem, которые связывают смету со справочниками услуг и материалов. Такой подход позволяет хранить несколько услуг и материалов в рамках одной сметы.

Заказ-наряд хранится в таблице WorkOrder. Он связан с заявкой, сметой, ответственным пользователем и текущим статусом. Состав заказ-наряда представлен таблицами WorkOrderService и WorkOrderMaterial. Фотографии результата выполненных работ хранятся в таблице WorkOrderPhoto, а история изменения статусов заказ-наряда фиксируется в таблице WorkOrderStatusHistory.

Справочник услуг хранится в таблице ServiceCatalog. Справочник материалов и складские остатки хранятся в таблице MaterialCatalog. Для фиксации поступления и списания материалов используется таблица StockMovement. Таблица ScheduleSlot предназначена для хранения планируемых временных интервалов выполнения работ.

Сведения о пользователях системы хранятся в таблице User. Таблица связана с таблицей Role, содержащей перечень ролей пользователей. Для

хранения справочника статусов используется таблица Status, а для хранения областей автомобиля, к которым могут относиться фотографии результата работ, используется таблица BodyArea.

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

2.3 Проектирование алгоритмов

На основании разработанной архитектуры были определены основные программные компоненты информационной системы детейлинга. Приложение реализуется как настольная информационная система на языке C# с использованием технологии Windows Forms. Основная логика работы системы распределена между экранными формами, каждая из которых отвечает за выполнение определенного набора функций.

Программное средство включает формы для авторизации пользователя, навигации по разделам системы, создания клиентских заявок, работы с заказ-нарядами, выбора услуг и материалов, ведения складского учета, управления сотрудниками и формирования отчетов. Взаимодействие с базой данных осуществляется через SQL-запросы к Microsoft SQL Server.

Для построения проектных диаграмм использовался редактор draw.io [19]. Диаграмма классов разрабатываемой информационной системы представлена на рисунке 14.

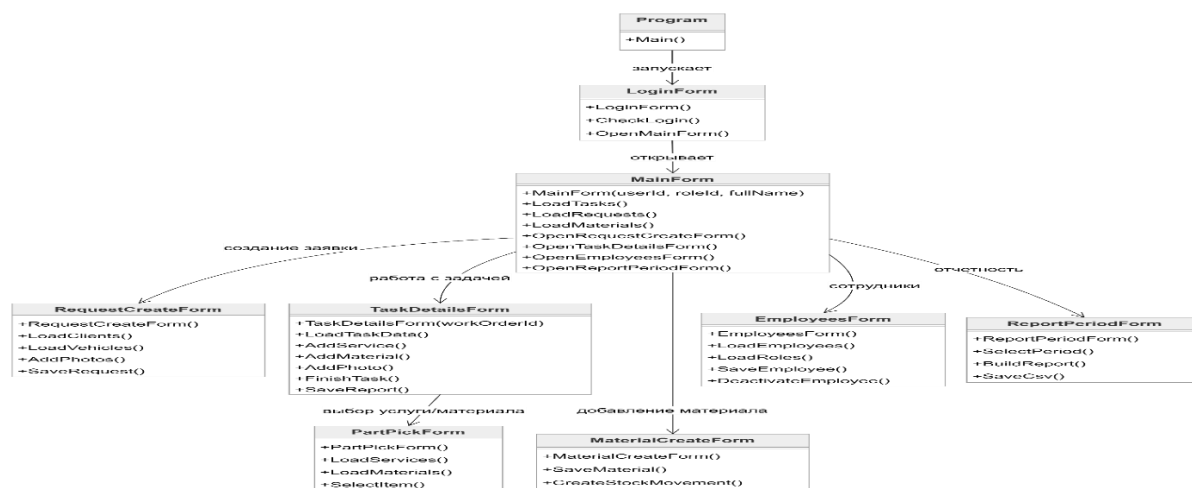


Рисунок 14 – Диаграмма классов информационной системы

Для описания порядка выполнения основных пользовательских сценариев были разработаны диаграммы алгоритмов. Они показывают последовательность действий пользователя и системы при создании клиентской заявки и выполнении задачи.

Диаграмма деятельности создания клиентской заявки представлена на рисунке 15.

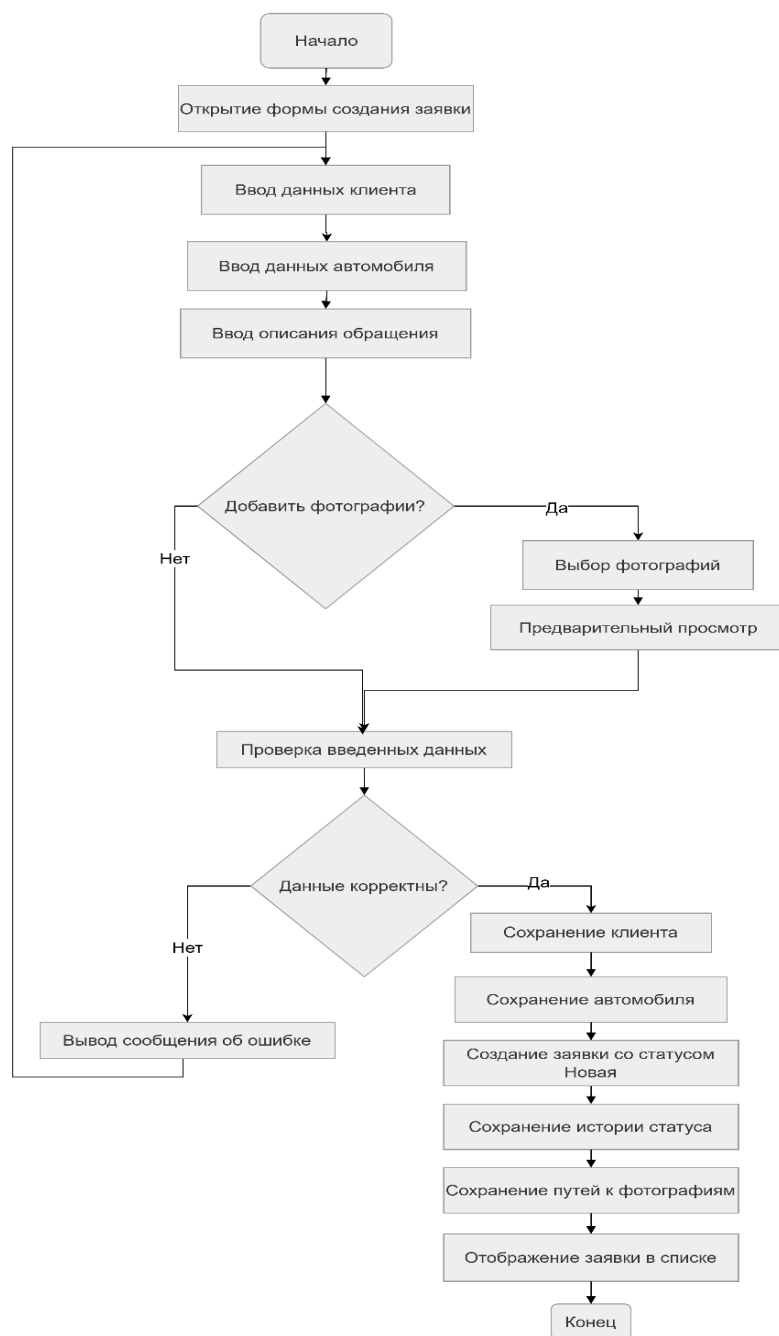


Рисунок 15 – Диаграмма деятельности создания клиентской заявки

Диаграмма деятельности выполнения задачи представлена на рисунке 16.

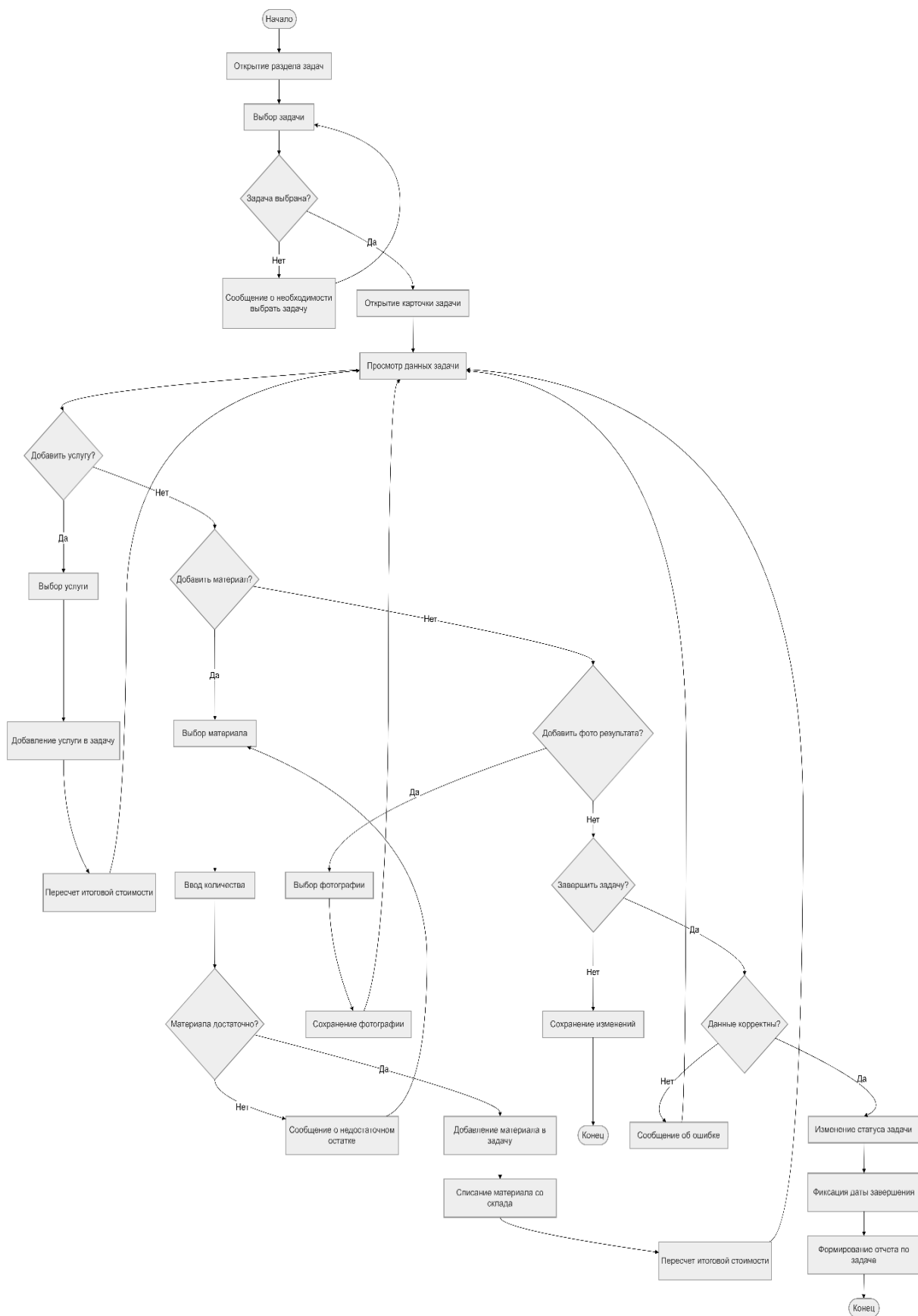


Рисунок 16 – Диаграмма деятельности выполнения задачи

Диаграмма последовательности создания клиентской заявки представлена на рисунке 17.

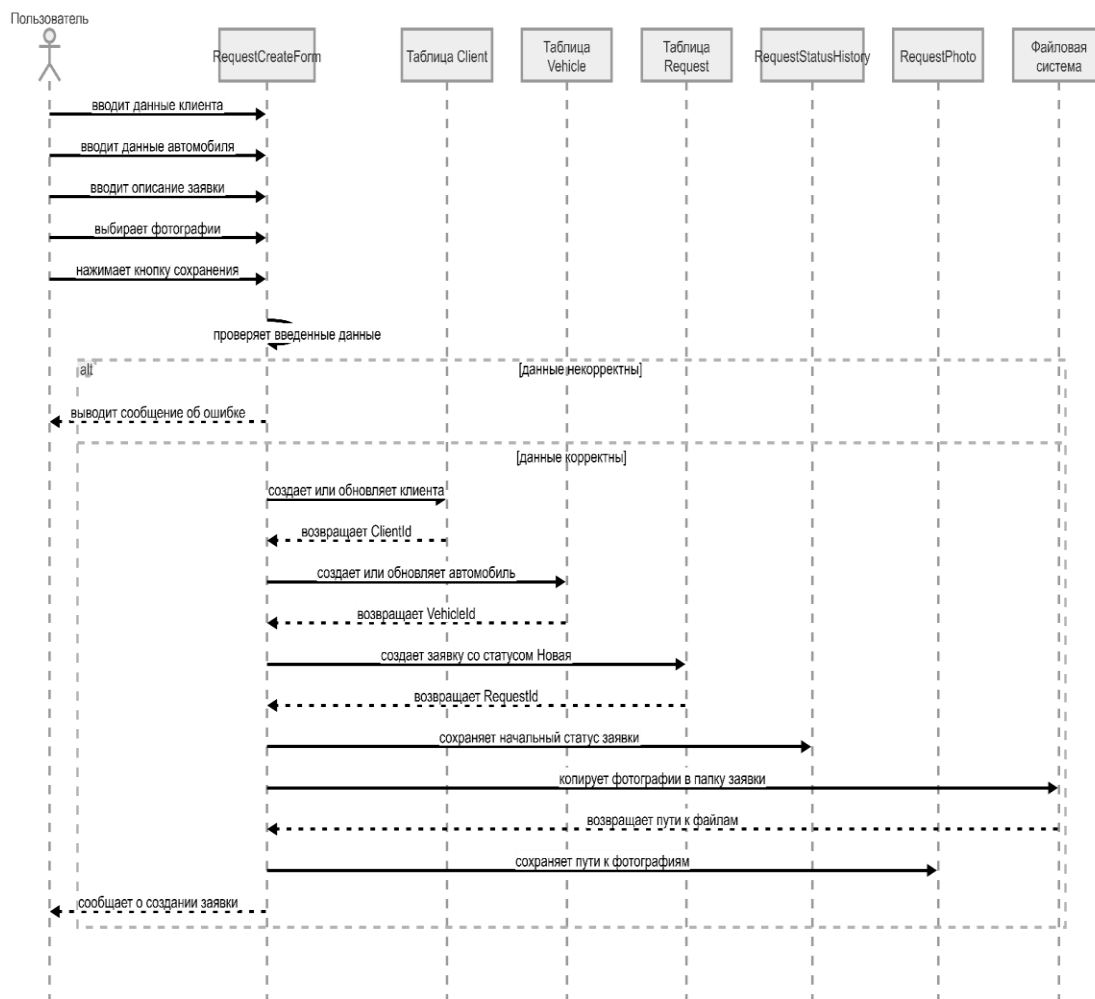


Рисунок 17 – Диаграмма последовательности создания клиентской заявки

Основные классы форм приложения представлены в таблице 13.

Таблица 13 – Основные формы приложения

Название класса	Назначение
Program	Содержит точку входа в приложение и запускает форму авторизации
LoginForm	Обеспечивает авторизацию пользователя по логину и паролю
MainForm	Представляет главное окно приложения и обеспечивает доступ к основным разделам системы
RequestCreateForm	Используется для создания клиентской заявки, выбора клиента и автомобиля
TaskDetailsForm	Отображает карточку заказ-наряда и обеспечивает работу с услугами, материалами и статусами
PartPickForm	Используется для выбора услуги или материала при добавлении позиции в заказ-наряд
MaterialCreateForm	Используется для добавления нового материала в складской каталог

Продолжение таблицы 13

Название класса	Назначение
EmployeesForm	Обеспечивает просмотр и редактирование данных сотрудников
Название класса	Назначение
ReportPeriodForm	Используется для выбора периода формирования отчета

Класс Program является начальной точкой запуска приложения. Он выполняет инициализацию Windows Forms и открывает форму авторизации. После успешного входа пользователя дальнейшая работа выполняется через главное окно приложения.

Класс LoginForm отвечает за вход пользователя в систему. На форме пользователь вводит логин и пароль. После нажатия кнопки входа приложение выполняет проверку учетных данных в базе данных. Если пользователь найден, его учетная запись активна и пароль указан верно, система определяет роль пользователя и открывает главное окно. Если данные введены неверно, пользователю выводится сообщение об ошибке.

Класс MainForm представляет основное окно информационной системы. Он отвечает за навигацию между разделами приложения и отображение данных в табличном виде. Через главное окно пользователь может перейти к заявкам, заказ-нарядам, складу, сотрудникам, аналитике и отчетам. Доступность разделов зависит от роли авторизованного пользователя.

В главной форме предусмотрена загрузка данных из базы данных и отображение их в таблицах. Для работы с разными разделами используются отдельные обработчики событий кнопок и элементов интерфейса. Например, пользователь может создать новую заявку, открыть карточку выбранного заказ-наряда, перейти к управлению складом, сформировать отчет или открыть список сотрудников.

Класс RequestCreateForm предназначен для оформления новой клиентской заявки. В данной форме пользователь может найти существующего клиента, добавить нового клиента, выбрать или указать автомобиль, заполнить описание проблемы или требуемых работ, а также сохранить заявку. После сохранения данные клиента, автомобиля и заявки заносятся в базу данных.

При создании заявки выполняется проверка обязательных полей. Если пользователь не указал необходимые сведения, приложение выводит сообщение об ошибке и не сохраняет некорректную запись. Это позволяет снизить вероятность появления неполных или ошибочных данных в базе.

Класс TaskDetailsForm используется для работы с карточкой заказ-наряда. В форме отображаются сведения о заявке, клиенте, автомобиле, ответственном сотруднике, статусе выполнения работ, услугах, материалах и итоговой стоимости. Пользователь может добавлять услуги и материалы, удалять ошибочно добавленные позиции, указывать плановую дату завершения, прикреплять фотографии результата и завершать заказ-наряд.

При добавлении услуги или материала система обновляет состав заказ-наряда и пересчитывает итоговую сумму. Для материалов дополнительно учитывается складской остаток. Если материала недостаточно, система должна запретить добавление позиции или сообщить пользователю о невозможности выполнения операции.

Класс PartPickForm предназначен для выбора позиции, добавляемой в заказ-наряд. В зависимости от режима работы форма может использоваться для выбора услуги из справочника услуг или материала из справочника материалов. Пользователь выбирает нужную позицию, указывает количество и подтверждает добавление.

Класс MaterialCreateForm используется для добавления нового материала в складской каталог. В форме указывается название материала, единица измерения, цена, текущий остаток и минимальный остаток. После сохранения материал становится доступен для использования в складском учете и при добавлении материалов в заказ-наряд.

Класс EmployeesForm отвечает за управление сотрудниками. В форме отображается список пользователей системы. Пользователь с соответствующими правами может добавлять сотрудников, изменять их данные, назначать роли и деактивировать учетные записи. Наличие отдельной формы для сотрудников позволяет централизованно управлять доступом к системе.

Класс ReportPeriodForm предназначен для выбора периода формирования отчета. Пользователь указывает начальную и конечную дату, после чего система формирует отчет по выполненным заказам за выбранный период. Данные отчета могут использоваться для анализа деятельности детейлинг-центра.

Основные функциональные модули представлены в таблице 14.

Таблица 14 – Функциональные модули приложения

Модуль	Назначение
Модуль авторизации	Проверяет учетные данные пользователя и определяет его роль
Модуль заявок	Обеспечивает создание и обработку клиентских заявок
Модуль заказ-нарядов	Обеспечивает работу с заказами, услугами, материалами и статусами
Модуль складского учета	Обеспечивает ведение материалов, поступлений и списаний
Модуль сотрудников	Обеспечивает управление учетными записями сотрудников
Модуль отчетности	Формирует отчеты за выбранный период
Модуль доступа к данным	Выполняет SQL-запросы к базе данных Microsoft SQL Server

Модуль авторизации выполняет проверку логина и пароля пользователя. После успешной проверки система получает роль пользователя и передает эти данные в главное окно. Роль используется для ограничения доступа к разделам приложения.

Модуль заявок обеспечивает оформление обращений клиентов. Он связан с таблицами клиентов, автомобилей, заявок и фотографий заявок. При создании заявки система сохраняет сведения о клиенте, автомобиле, описании проблемы и текущем статусе.

Модуль заказ-нарядов отвечает за сопровождение работ после принятия заявки. Он связан с заказ-нарядами, услугами, материалами, статусами и фотографиями результата. Этот модуль является одним из основных, так как через него выполняется учет фактически оказанных услуг и использованных материалов.

Модуль складского учета предназначен для работы с материалами. Он позволяет хранить сведения о материалах, контролировать остатки, фиксировать поступления и списания. Данные складского учета используются при добавлении материалов в заказ-наряд.

Модуль сотрудников используется для управления пользователями системы. Он обеспечивает работу с учетными записями и ролями. Данный модуль необходим для разграничения доступа между администратором, мастером, менеджером или приемщиком и сотрудником склада.

Модуль отчетности выполняет выборку данных из базы за заданный период и формирует итоговую информацию по заказам. В отчет включаются сведения о клиентах, автомобилях, выполненных работах, использованных материалах и итоговых суммах.

Модуль доступа к данным обеспечивает взаимодействие приложения с Microsoft SQL Server. Он выполняет запросы на получение, добавление, изменение и удаление данных. Через данный модуль формы приложения получают сведения из базы данных и сохраняют результаты работы пользователя.

При проектировании программных средств особое внимание уделяется обработке пользовательских действий. Большинство операций выполняется через обработчики событий элементов интерфейса. Например, нажатие кнопки создания заявки открывает форму RequestCreateForm, выбор заказ-наряда и нажатие кнопки просмотра открывает TaskDetailsForm, а выбор периода отчета открывает ReportPeriodForm.

Для снижения вероятности ошибок в приложении предусматривается проверка вводимых данных. Проверяются обязательные текстовые поля, корректность числовых значений, выбор записи в таблице, наличие прав доступа и допустимость выполнения операции с учетом текущего статуса записи.

Проектируемая структура программных средств позволяет разделить функции приложения между отдельными формами и модулями, что соответствует принципам модульной организации программного кода [11]. Это упрощает сопровождение программы, так как изменение одной части системы не требует полной переработки остальных компонентов. Например, доработка формы складского учета не затрагивает форму авторизации, а изменение логики формирования отчета не требует изменения формы создания заявки.

2.4 Проектирование пользовательского интерфейса

Пользовательский интерфейс разрабатываемой информационной системы детейлинг-центра представляет собой настольное приложение, работающее в операционной системе Windows. Интерфейс создается с использованием технологии Windows Forms, что позволяет использовать стандартные элементы управления: формы, кнопки, таблицы, текстовые поля, раскрывающиеся списки, поля выбора даты, списки изображений и области предварительного просмотра фотографий.

Целью проектирования интерфейса является создание простого, понятного и удобного средства взаимодействия сотрудников детейлинг-центра с системой. Интерфейс должен быть интуитивно понятным, чтобы пользователь без специальной подготовки мог быстро освоить основные функции приложения и выполнять повседневные операции: создавать заявки, работать с заказ-нарядами, учитывать материалы, управлять сотрудниками и формировать отчеты.

В системе предусмотрены следующие основные формы и элементы интерфейса:

- а) форма авторизации, предназначенная для входа пользователя в систему по логину и паролю;
- б) главное окно приложения, содержащее навигационное меню и основную область отображения данных;
- в) раздел заявок, предназначенный для просмотра клиентских заявок, создания новой заявки и изменения ее статуса;
- г) форма создания клиентской заявки, позволяющая указать сведения о клиенте, автомобиле, описании обращения и фотографиях автомобиля до выполнения работ;
- д) раздел задач, предназначенный для просмотра заказ-нарядов и перехода к подробной карточке выбранного заказа;

- е) карточка заказ-наряда, предназначенная для просмотра сведений о заказе, добавления услуг и материалов, установки плановой даты завершения, прикрепления фотографий и завершения работ;
- ж) форма выбора услуги или материала, используемая при добавлении позиции в заказ-наряд;
- з) раздел склада, предназначенный для просмотра материалов, добавления новых позиций, фиксации поступления и списания;
- и) форма добавления материала, позволяющая указать название материала, единицу измерения, цену, минимальный остаток и начальное количество;
- к) форма управления сотрудниками, предназначенная для добавления и редактирования учетных записей пользователей системы;
- л) раздел анализа и отчетности, предназначенный для просмотра аналитических данных и формирования отчета за выбранный период;
- м) форма выбора периода отчета, позволяющая указать начальную и конечную дату, просмотреть отчетные данные и сохранить результат.

Форма авторизации является начальной формой приложения. Пользователь вводит логин и пароль, после чего система проверяет учетные данные в базе данных. При успешной авторизации открывается главное окно приложения. Если логин или пароль указаны неверно, пользователь получает сообщение об ошибке. Наличие формы авторизации позволяет ограничить доступ к функциям системы и обеспечить работу приложения с учетом роли пользователя.

Главное окно приложения является основным элементом пользовательского интерфейса. В левой части формы размещается навигационная панель с кнопками перехода к разделам системы: заявки, задачи, склад, анализ, сотрудники и выход. Основная часть окна предназначена для отображения данных в табличном виде. В зависимости от выбранного раздела в таблице отображаются заявки, заказ-наряды, материалы или аналитическая информация.

Для работы с заявками в главном окне предусмотрены элементы управления, позволяющие создать новую заявку, перевести заявку в обработку, принять ее в работу, перевести в ожидание, отклонить или отметить получение детали. Такой набор действий соответствует основным этапам обработки клиентского обращения в детейлинг-центре.

Форма создания клиентской заявки используется для оформления нового обращения клиента. На форме предусмотрены поля для ввода ФИО клиента, паспортных данных, телефона, электронной почты, марки автомобиля, модели, года выпуска, VIN, государственного регистрационного номера и комментария. Для выбора клиента используется раскрывающийся список. Также предусмотрены кнопки перехода между автомобилями клиента, что удобно при работе с постоянными клиентами.

Отдельная часть формы создания заявки предназначена для работы с фотографиями автомобиля до выполнения работ. Пользователь может добавить фотографию, удалить ошибочно выбранное изображение и просмотреть выбранный файл в области предварительного просмотра. Это позволяет зафиксировать исходное состояние автомобиля при приеме заявки.

Карточка задачи предназначена для подробной работы с выбранным заказом. В ней отображаются основные сведения о заказ-наряде и его текущем статусе. Основная информация по услугам и материалам представлена в таблице позиций. Пользователь может добавить услугу, добавить материал, удалить выбранную строку, зафиксировать осмотр, установить плановую дату завершения, завершить работу, сформировать отчет или закрыть карточку.

В карточке заказ-наряда также предусмотрена работа с фотографиями после выполнения работ. Для этого используются список фотографий, область предварительного просмотра, кнопка добавления фотографии и кнопка удаления фотографии. Такая структура позволяет хранить не только текстовую информацию о заказе, но и визуальное подтверждение результата выполненных работ.

Диаграмма состояний интерфейса при создании клиентской заявки представлена на рисунке 18.

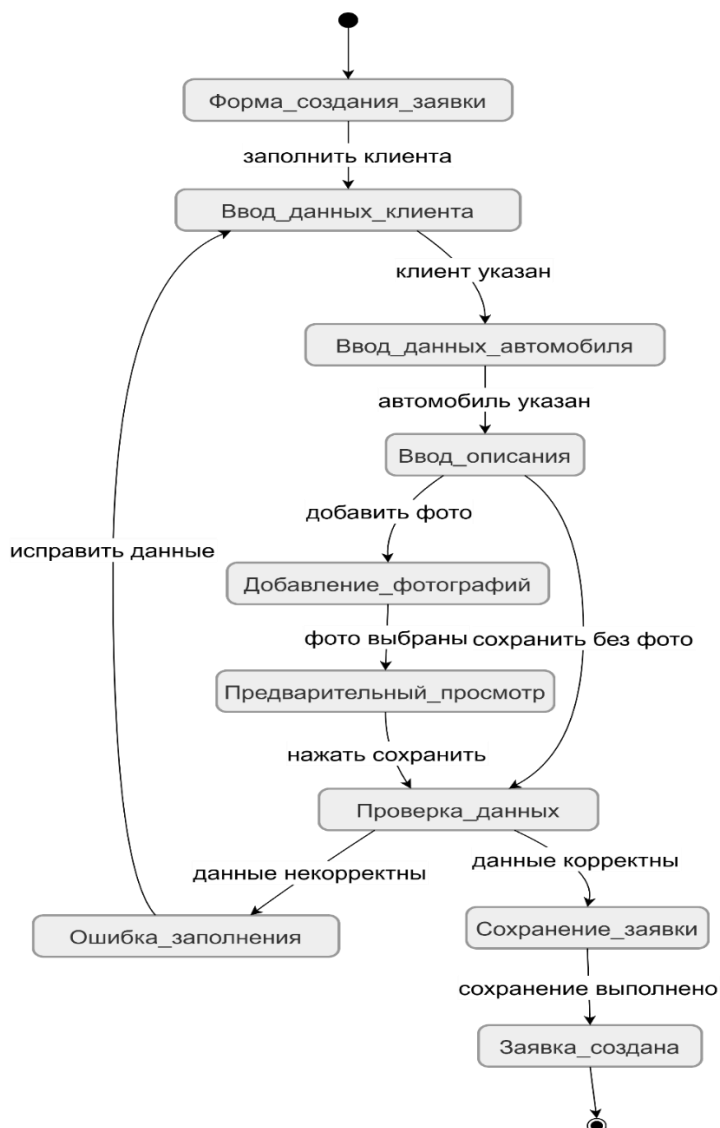


Рисунок 18 – Диаграмма состояний интерфейса при создании клиентской заявки

Форма выбора услуги или материала используется при добавлении позиции в заказ-наряд. В верхней части формы отображается таблица доступных позиций, из которой пользователь выбирает нужную услугу или материал. В нижней части формы отображается список уже запрошенных позиций. Для управления выбором предусмотрены кнопки подтверждения, отмены и удаления позиции. Это позволяет пользователю контролировать состав добавляемых работ и материалов.

Раздел склада предназначен для учета материалов, используемых в процессе работы детейлинг-центра. Через главное окно пользователь может перейти к списку материалов, добавить новый материал, зафиксировать поступление или выполнить списание со склада. Данные отображаются в табличном виде, что упрощает просмотр остатков и выбор нужной позиции.

Форма добавления материала используется для внесения новой позиции в складской каталог. На форме размещаются поля для ввода названия материала, единицы измерения, цены, минимального остатка и количества. После заполнения данных пользователь может сохранить новую запись или отменить операцию.

Форма управления сотрудниками предназначена для администрирования учетных записей пользователей. На форме предусмотрен выбор сотрудника из списка, ввод ФИО, логина, пароля и подтверждения пароля. Также пользователь может выбрать роль сотрудника и отметить учетную запись как неактивную. Для выполнения действий используются кнопки добавления, сохранения и закрытия формы.

Форма выбора периода отчета используется для формирования отчетности. Пользователь указывает начальную и конечную дату периода, после чего может отобразить отчетные данные в таблице. На форме также предусмотрено отображение итоговых значений и сохранение отчета. Это позволяет получать сведения о выполненных работах за выбранный промежуток времени.

При проектировании интерфейса учитывается разграничение доступа по ролям пользователей. Администратору доступны функции управления сотрудниками и основные разделы системы. Менеджер или приемщик работает с клиентскими заявками и заказ-нарядами. Мастер получает доступ к задачам и карточкам заказ-нарядов. Сотрудник склада работает с материалами, поступлениями и списаниями. Такое разделение снижает риск ошибочных действий и делает интерфейс более удобным для каждой категории пользователей.

Пользовательский интерфейс построен по принципу разделения функций между отдельными формами. Главное окно используется для навигации и просмотра списков, а отдельные формы применяются для выполнения конкретных операций: создания заявки, работы с заказ-нарядом, выбора позиции, добавления материала, управления сотрудниками и формирования отчета. Такой подход упрощает структуру приложения и делает работу пользователя последовательной.

Для повышения удобства работы в интерфейсе используются таблицы, позволяющие быстро просматривать списки записей и выбирать нужную строку. Кнопки действий размещаются рядом с соответствующими таблицами и разделами, что сокращает количество переходов между окнами. Подписи элементов управления соответствуют предметной области: заявка, задача, склад, сотрудник, материал, услуга, отчет, фото, плановая дата окончания.

В процессе проектирования интерфейса также учитывается необходимость проверки вводимых данных. При создании заявки проверяется заполнение обязательных сведений о клиенте и автомобиле. При добавлении материала проверяется корректность числовых значений. При работе с сотрудниками проверяются данные учетной записи. При формировании отчета проверяется выбранный период. Это позволяет уменьшить количество ошибок при работе с системой.

Разработанный пользовательский интерфейс обеспечивает доступ ко всем основным функциям информационной системы детейлинг-центра и соответствует задачам автоматизации предметной области. Структура интерфейса позволяет пользователю последовательно выполнять операции от создания клиентской заявки до завершения заказ-наряда и формирования отчетности.

3 Реализация программных средств

3.1 Реализация базы данных

Для хранения данных информационной системы детейлинг-центра была реализована база данных AutoDetailing в системе управления базами данных Microsoft SQL Server[15]. Разработка и редактирование базы данных выполнялись с использованием среды SQL Server Management Studio [17].

Развертывание информационной системы предусматривает установку клиентского приложения на рабочем месте пользователя и размещение базы данных Microsoft SQL Server на локальном компьютере или сервере локальной сети.

Диаграмма развертывания информационной системы представлена на рисунке 19.

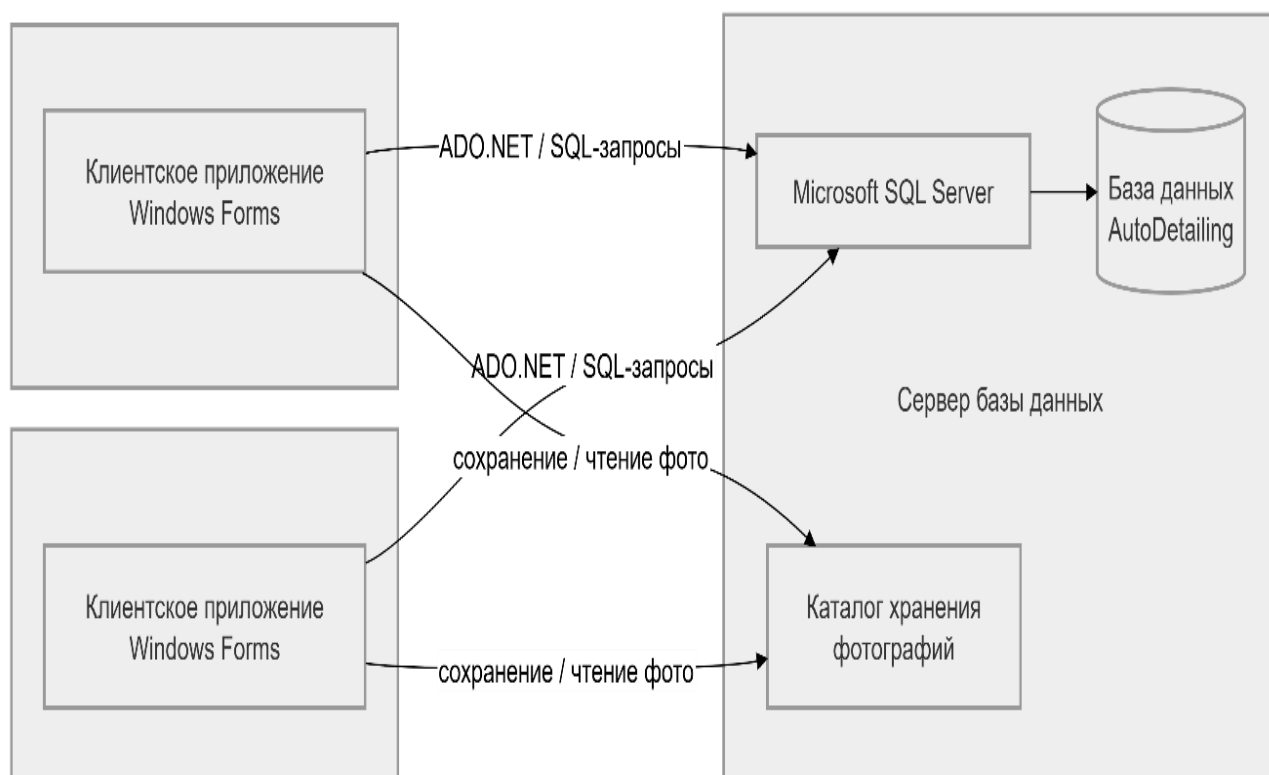


Рисунок 19 – Диаграмма развертывания информационной системы

База данных предназначена для хранения сведений о клиентах, автомобилях, заявках, фотографиях, осмотрах, сметах, заказ-нарядах, услугах,

материалах, складских операциях, сотрудниках, ролях и статусах. Структура базы данных была разработана с учетом требований предметной области и обеспечивает хранение всей информации, необходимой для работы приложения.

В базе данных были реализованы следующие основные таблицы:

- а) Client – хранение данных о клиентах;
 - б) Vehicle – хранение данных об автомобилях клиентов;
 - в) Request – хранение клиентских заявок;
 - г) RequestPhoto – хранение фотографий, прикрепленных к заявке;
 - д) RequestStatusHistory – хранение истории изменения статусов заявок;
 - е) Inspection – хранение данных осмотра автомобиля;
 - ж) InspectionStatusHistory – хранение истории изменения статусов осмотра;
 - з) Estimate – хранение данных сметы;
 - и) EstimateServiceItem – хранение услуг, включенных в смету;
 - к) EstimateMaterialItem – хранение материалов, включенных в смету;
 - л) WorkOrder – хранение заказ-нарядов;
 - м) WorkOrderService – хранение услуг, добавленных в заказ-наряд;
 - н) WorkOrderMaterial – хранение материалов, добавленных в заказ-наряд;
 - о) WorkOrderPhoto – хранение фотографий по заказ-наряду;
 - п) WorkOrderStatusHistory – хранение истории изменения статусов заказ-наряда;
 - р) MaterialCatalog – хранение справочника материалов;
 - с) ServiceCatalog – хранение справочника услуг;
 - т) StockMovement – хранение складских операций;
 - у) User – хранение учетных записей пользователей;
 - ф) Role – хранение ролей пользователей;
 - х) Status – хранение статусов;
 - ц) BodyArea – хранение областей кузова;
 - ч) ScheduleSlot – хранение временных интервалов выполнения работ.
- Список таблиц базы данных AutoDetailing представлен на рисунке 20.

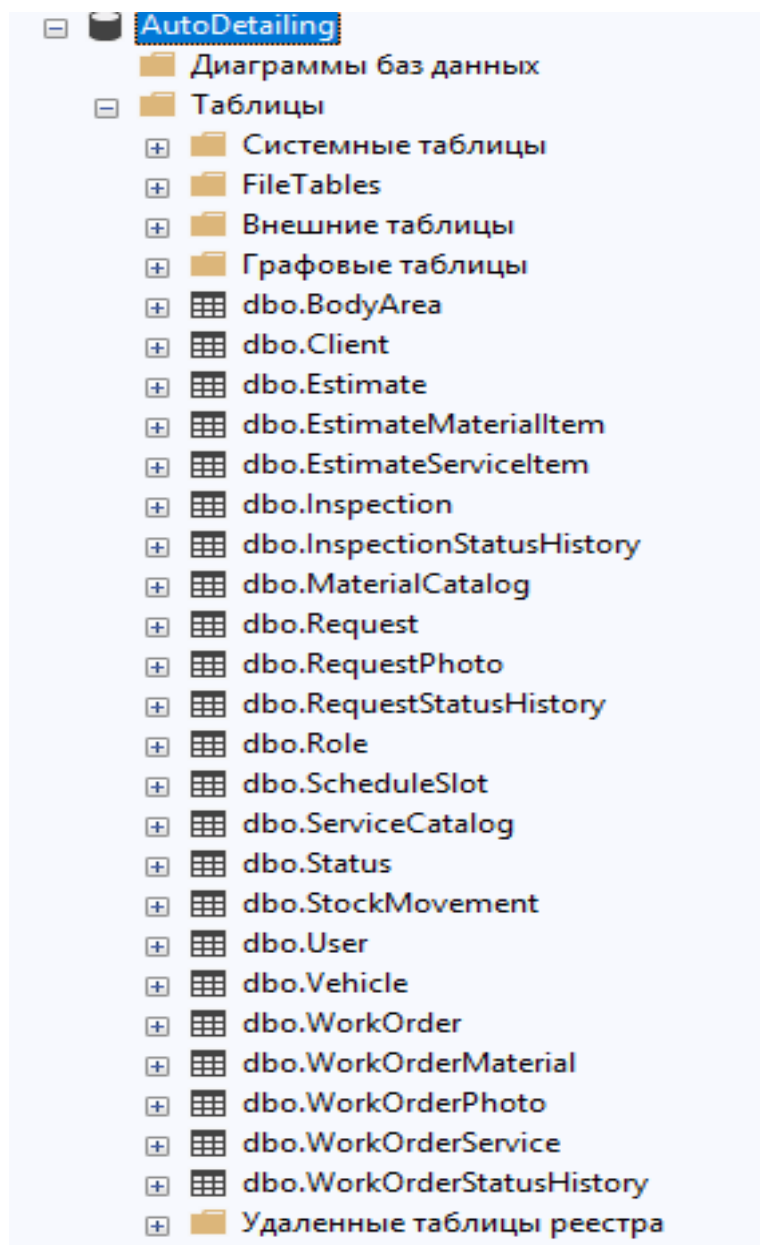


Рисунок 20 – Таблицы базы данных AutoDetailing

Создание базы данных выполняется с помощью SQL-запроса на языке Transact-SQL, представленного в листинге 1 [16].

Листинг 1 – Создание базы данных AutoDetailing

```
CREATE DATABASE AutoDetailing;
GO
USE AutoDetailing;
GO
```

Для хранения клиентов была создана таблица Client. В данной таблице хранятся ФИО клиента, телефон, электронная почта и паспортные данные. Поле Id используется в качестве первичного ключа.

Создание таблицы Client представлено в листинге 2.

Листинг 2 – Создание таблицы Client

```
CREATE TABLE dbo.Client
(
    Id INT IDENTITY(1,1) PRIMARY KEY,
    FullName NVARCHAR(200) NOT NULL,
    Phone NVARCHAR(30) NOT NULL,
    Email NVARCHAR(100) NULL,
    Passport NVARCHAR(20) NULL
);
```

Для хранения автомобилей клиентов была создана таблица Vehicle. Она связана с таблицей Client при помощи внешнего ключа ClientId. Благодаря этому одному клиенту может соответствовать несколько автомобилей.

Создание таблицы Vehicle представлено в листинге 3.

Листинг 3 – Создание таблицы Vehicle

```
CREATE TABLE dbo.Vehicle
(
    Id INT IDENTITY(1,1) PRIMARY KEY,
    ClientId INT NOT NULL
        CONSTRAINT FK_Vehicle_Client
        REFERENCES dbo.Client(Id),
    Brand NVARCHAR(60) NOT NULL,
    Model NVARCHAR(60) NOT NULL,
    [Year] INT NULL,
    Vin NVARCHAR(17) NULL,
    RegPlate NVARCHAR(20) NULL
);
```

Для хранения клиентских заявок была создана таблица Request. В ней фиксируются клиент, автомобиль, дата создания заявки, описание обращения, приоритет и текущий статус заявки. Таблица Request связана с таблицами Client, Vehicle и Status.

Создание таблицы Request представлено в листинге 4.

Листинг 4 – Создание таблицы Request

```
CREATE TABLE dbo.Request
(
    Id INT IDENTITY(1,1) PRIMARY KEY,
    ClientId INT NOT NULL
        CONSTRAINT FK_Request_Client
        REFERENCES dbo.Client(Id),
    VehicleId INT NOT NULL
```

```

        CONSTRAINT FK_Request_Vehicle
        REFERENCES dbo.Vehicle(Id),
    CreatedAt DATETIME2 NOT NULL
        CONSTRAINT DF_Request_CreatedAt DEFAULT SYSUTCDATETIME(),
    Description NVARCHAR(1000) NOT NULL,
    Priority TINYINT NOT NULL
        CONSTRAINT DF_Request_Priority DEFAULT 1,
    StatusId INT NOT NULL
        CONSTRAINT FK_Request_Status
        REFERENCES dbo.Status(Id)
);

```

Для хранения фотографий, прикрепленных к заявке, была создана таблица RequestPhoto. В базе данных сохраняется не само изображение, а путь к файлу фотографии. Такой способ позволяет уменьшить объем базы данных и хранить изображения в файловой системе приложения.

Создание таблицы RequestPhoto представлено в листинге 5.

Листинг 5 – Создание таблицы RequestPhoto

```

CREATE TABLE dbo.RequestPhoto
(
    Id INT IDENTITY(1,1) PRIMARY KEY,
    RequestId INT NOT NULL
        CONSTRAINT FK_RequestPhoto_Request
        REFERENCES dbo.Request(Id)
        ON DELETE CASCADE,
    FilePath NVARCHAR(500) NOT NULL,
    Comment NVARCHAR(200) NULL,
    CreatedAt DATETIME2 NOT NULL
        CONSTRAINT DF_RequestPhoto_CreatedAt DEFAULT
        SYSUTCDATETIME()
);

```

Для хранения заказ-нарядов была создана таблица WorkOrder. Она содержит данные о связанной заявке, смете, ответственном пользователе, датах выполнения работ, статусе и скидке. Таблица WorkOrder является одной из основных таблиц базы данных, так как именно в ней фиксируется выполнение работ по заявке клиента.

Создание таблицы WorkOrder представлено в листинге 6.

Листинг 6 – Создание таблицы WorkOrder

```

CREATE TABLE dbo.WorkOrder
(
    Id INT IDENTITY(1,1) PRIMARY KEY,
    RequestId INT NOT NULL

```

```

        CONSTRAINT FK_WorkOrder_Request
        REFERENCES dbo.Request(Id),
    EstimateId INT NULL
        CONSTRAINT FK_WorkOrder_Estimate
        REFERENCES dbo.Estimate(Id),
    ResponsibleUserId INT NULL
        CONSTRAINT FK_WorkOrder_User
        REFERENCES dbo.[User](Id),
    CreatedAt DATETIME2 NOT NULL
        CONSTRAINT DF_WorkOrder_CreatedAt DEFAULT
SYSUTCDATETIME(),
    PlannedStart DATETIME2 NULL,
    PlannedFinish DATETIME2 NULL,
    ActualFinish DATETIME2 NULL,
    StatusId INT NOT NULL
        CONSTRAINT FK_WorkOrder_Status
        REFERENCES dbo.Status(Id),
    Discount DECIMAL(12,2) NOT NULL
        CONSTRAINT DF_WorkOrder_Discount DEFAULT 0
);

```

Для хранения услуг, добавленных в заказ-наряд, была создана таблица `WorkOrderService`. Она связывает заказ-наряд со справочником услуг `ServiceCatalog`. В таблице также хранится количество услуги и цена на момент добавления в заказ-наряд.

Создание таблицы `WorkOrderService` представлено в листинге 7.

Листинг 7 – Создание таблицы `WorkOrderService`

```

CREATE TABLE dbo.WorkOrderService
(
    Id INT IDENTITY(1,1) PRIMARY KEY,
    WorkOrderId INT NOT NULL
        CONSTRAINT FK_WorkOrderService_WorkOrder
        REFERENCES dbo.WorkOrder(Id),
    ServiceId INT NOT NULL
        CONSTRAINT FK_WorkOrderService_Service
        REFERENCES dbo.ServiceCatalog(Id),
    Qty DECIMAL(12,3) NOT NULL,
    Price DECIMAL(12,2) NOT NULL
);

```

Для хранения материалов, использованных в заказ-наряде, была создана таблица `WorkOrderMaterial`. Она связывает заказ-наряд со справочником материалов `MaterialCatalog`. В таблице фиксируются материал, количество и цена.

Создание таблицы `WorkOrderMaterial` представлено в листинге 8.

Листинг 8 – Создание таблицы WorkOrderMaterial

```
CREATE TABLE dbo.WorkOrderMaterial
(
    Id INT IDENTITY(1,1) PRIMARY KEY,
    WorkOrderId INT NOT NULL
        CONSTRAINT FK_WorkOrderMaterial_WorkOrder
        REFERENCES dbo.WorkOrder(Id),
    MaterialId INT NOT NULL
        CONSTRAINT FK_WorkOrderMaterial_Material
        REFERENCES dbo.MaterialCatalog(Id),
    Qty DECIMAL(12,3) NOT NULL,
    Price DECIMAL(12,2) NOT NULL
);
```

Для учета материалов была создана таблица MaterialCatalog. В ней хранится наименование материала, единица измерения, розничная цена, текущий остаток и минимальный допустимый остаток.

Создание таблицы MaterialCatalog представлено в листинге 9.

Листинг 9 – Создание таблицы MaterialCatalog

```
CREATE TABLE dbo.MaterialCatalog
(
    Id INT IDENTITY(1,1) PRIMARY KEY,
    Name NVARCHAR(200) NOT NULL,
    Unit NVARCHAR(20) NOT NULL,
    RetailPrice DECIMAL(12,2) NOT NULL,
    StockQty DECIMAL(12,3) NOT NULL,
    MinStockQty DECIMAL(12,3) NOT NULL
);
```

Для фиксации складских операций была создана таблица StockMovement. В ней хранятся операции прихода и списания материалов. Поле MovementType определяет тип движения материала: приход или списание. При добавлении материала в заказ-наряд создается операция списания.

Создание таблицы StockMovement представлено в листинге 10.

Листинг 10 – Создание таблицы StockMovement

```
CREATE TABLE dbo.StockMovement
(
    Id INT IDENTITY(1,1) PRIMARY KEY,
    MaterialId INT NOT NULL
        CONSTRAINT FK_StockMovement_Material
        REFERENCES dbo.MaterialCatalog(Id),
    MovementType TINYINT NOT NULL,
    Qty DECIMAL(12,3) NOT NULL,
    UnitPrice DECIMAL(12,2) NOT NULL,
```

```

Reason NVARCHAR(200) NULL,
WorkOrderId INT NULL,
CreatedAt DATETIME2 NOT NULL
    CONSTRAINT DF_StockMovement_CreatedAt DEFAULT
SYSUTCDATETIME()
);

```

Для хранения учетных записей сотрудников была создана таблица User. В ней сохраняются логин, хэш пароля, ФИО пользователя, роль, признак активности учетной записи, соль пароля и количество итераций хэширования.

Создание таблицы User представлено в листинге 11.

Листинг 11 – Создание таблицы User

```

CREATE TABLE dbo.[User]
(
    Id INT IDENTITY(1,1) PRIMARY KEY,
    Login NVARCHAR(50) NOT NULL,
    PasswordHash NVARCHAR(200) NOT NULL,
    FullName NVARCHAR(200) NOT NULL,
    RoleId INT NOT NULL
        CONSTRAINT FK_User_Role
        REFERENCES dbo.Role(Id),
    IsActive BIT NOT NULL,
    PasswordSalt VARBINARY(16) NULL,
    PasswordIterations INT NOT NULL
);

```

Для хранения ролей пользователей была создана таблица Role. Роль пользователя используется в приложении для разграничения доступа к разделам системы.

Создание таблицы Role представлено в листинге 12.

Листинг 12 – Создание таблицы Role

```

CREATE TABLE dbo.Role
(
    Id INT IDENTITY(1,1) PRIMARY KEY,
    Name NVARCHAR(50) NOT NULL
);

```

Для хранения статусов была создана таблица Status. Она используется для заявок, осмотров и заказ-нарядов. Такой подход позволяет использовать единый справочник статусов во всех основных процессах системы.

Создание таблицы Status представлено в листинге 13.

Листинг 13 – Создание таблицы Status

```
CREATE TABLE dbo.Status
(
    Id INT IDENTITY(1,1) PRIMARY KEY,
    Name NVARCHAR(50) NOT NULL
);
```

После создания таблиц в справочник статусов были добавлены основные значения, используемые в системе.

Листинг 14 – Заполнение справочника статусов

```
INSERT INTO dbo.Status (Name)
VALUES
(N'Новая'),
(N'В работе'),
(N'Ожидание'),
(N'Завершена'),
(N'Отклонена');
```

Также в базу данных были добавлены роли пользователей, необходимые для разграничения доступа в приложении.

Листинг 15 – Заполнение справочника ролей

```
INSERT INTO dbo.Role (Name)
VALUES
(N'Администратор'),
(N'Мастер'),
(N'Менеджер/приёмщик'),
(N'Склад');
```

Для получения текущих складских остатков используется запрос, который рассчитывает остаток материала на основании операций прихода и списания.

Пример такого запроса представлен в листинге 16.

Листинг 16 – Получение складских остатков материалов

```
SELECT
    mc.Id AS MaterialID,
    mc.Name AS [Материал],
    mc.Unit AS [Ед. изм.],
    mc.RetailPrice AS [Цена],
    mc.MinStockQty AS [Мин. остаток],
    ISNULL(SUM(CASE
        WHEN sm.MovementType = 1 THEN sm.Qty
        ELSE -sm.Qty
    END), 0) AS [Остаток],
    CASE
        WHEN ISNULL(SUM(CASE
            WHEN sm.MovementType = 1 THEN sm.Qty
```

```

        ELSE -sm.Qty
    END), 0) < mc.MinStockQty
    THEN N'Ниже минимума'
    ELSE N'Ок'
END AS [Статус]
FROM dbo.MaterialCatalog mc
LEFT JOIN dbo.StockMovement sm ON sm.MaterialId = mc.Id
GROUP BY mc.Id, mc.Name, mc.Unit, mc.RetailPrice, mc.MinStockQty
ORDER BY mc.Name;
```

Данный запрос используется для отображения материалов в разделе складского учета. Если рассчитанный остаток материала меньше минимального значения, система показывает статус «Ниже минимума».

Связи между таблицами обеспечивают целостность данных. Например, заявка не может быть создана без клиента и автомобиля, заказ-наряд не может существовать без связанной заявки, а услуга или материал в заказ-наряде должны ссылаться на соответствующие справочники.

Реализованная база данных обеспечивает хранение информации, необходимой для работы всех основных модулей информационной системы детейлинг-центра.

3.2 Реализация модуля авторизации

Модуль авторизации реализован в форме LoginForm. Данная форма открывается первой при запуске приложения и предназначена для проверки прав пользователя перед предоставлением доступа к основным функциям информационной системы.

Форма авторизации содержит поле для ввода логина, поле для ввода пароля и кнопку входа в систему. Для поля пароля включено скрывание вводимых символов, что позволяет не отображать пароль пользователя в открытом виде.

При нажатии на кнопку «Войти» приложение получает значения из полей ввода. Логин очищается от лишних пробелов в начале и конце строки, а пароль используется без удаления пробелов, так как пробел может быть частью пароля. Если пользователь не указал логин или пароль, приложение выводит предупреждение и не выполняет обращение к базе данных.

Интерфейс формы авторизации представлен на рисунке 21.

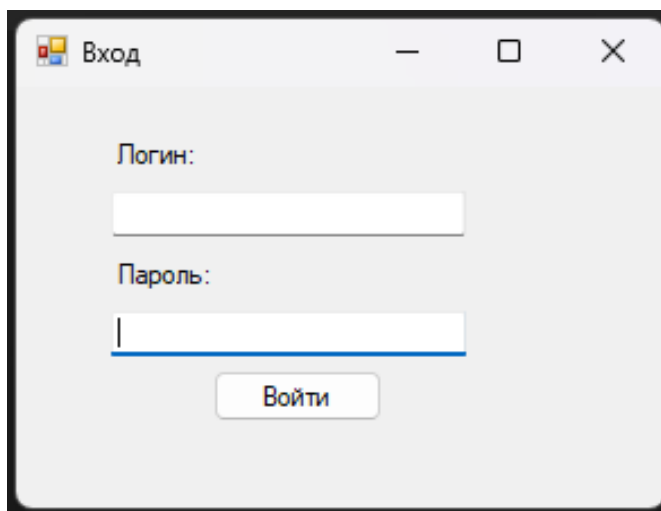


Рисунок 21 – Форма авторизации пользователя

Проверка заполнения полей авторизации представлена в листинге 17.

Листинг 17 – Проверка заполнения полей авторизации

```
string login = txtLogin.Text.Trim();
string password = txtPassword.Text;
if (string.IsNullOrEmpty(login) ||
    string.IsNullOrEmpty(password))
{
    MessageBox.Show(«Введите логин и пароль.», «Вход»,
        MessageBoxButtons.OK, MessageBoxIcon.Warning);
    return;
}
```

Для подключения к базе данных используется строка подключения к базе AutoDetailing. В модуле авторизации обращение к базе данных выполняется через класс SqlConnection, относящийся к средствам ADO.NET [14]. Для выполнения SQL-запроса используется объект SqlCommand.

Строка подключения к базе данных представлена в листинге 18.

Листинг 18 – Строка подключения к базе данных

```
private const string ConnStr =
    @«Server=localhost;Database=AutoDetailing;
    Trusted_Connection=True;»
```

После проверки заполнения полей выполняется поиск активной учетной записи пользователя в таблице User. Запрос выбирает идентификатор пользователя, ФИО, роль, хэш пароля, соль пароля и количество итераций

хэширования. Условие `IsActive = 1` позволяет разрешить вход только активным учетным записям.

SQL-запрос для поиска пользователя представлен в листинге 19.

Листинг 19 – Поиск активной учетной записи пользователя

```
const string sql = @»
SELECT TOP 1
    u.Id,
    u.FullName,
    u.RoleId,
    u.PasswordHash,
    u.PasswordSalt,
    u.PasswordIterations
FROM dbo.[User] u
WHERE u.Login = @login AND u.IsActive = 1;»
```

Для защиты от подстановки пользовательских данных в SQL-запрос используется параметр `@login`. Это позволяет передавать введенный логин как значение параметра, а не как часть текста SQL-запроса.

Выполнение запроса поиска пользователя представлено в листинге 20.

Листинг 20 – Выполнение запроса поиска пользователя

```
using (var con = new SqlConnection(ConnStr))
using (var cmd = new SqlCommand(sql, con))
{
    cmd.Parameters.AddWithValue («@login», login);
    con.Open();
    using (var reader = cmd.ExecuteReader())
    {
        if (!reader.Read())
        {
            MessageBox.Show («Неверный логин или пароль» «Вход»,
                MessageBoxButtons.OK, MessageBoxIcon.Error);
            return;
        }
        int userId = Convert.ToInt32(reader[«Id»]);
        int roleId = Convert.ToInt32(reader[«RoleId»]);
        string fio = reader[«FullName»].ToString();
    }
}
```

Если пользователь не найден, его учетная запись неактивна или введены неверные учетные данные, приложение выводит сообщение «Неверный логин или пароль». Такой подход не раскрывает информацию о существовании учетной записи в системе.

После поиска пользователя из базы данных считываются данные, необходимые для проверки пароля: хэш, соль и количество итераций. Если один из этих параметров отсутствует, авторизация не выполняется, а пользователю выводится сообщение о некорректной настройке учетной записи.

Проверка пароля выполняется методом `VerifyPasswordPBKDF2`, который повторно вычисляет хэш на основе введенного пароля и сравнивает его с сохраненным значением.

Метод проверки пароля представлен в листинге 21.

Листинг 21 – Проверка пароля пользователя

```
private static bool VerifyPasswordPBKDF2(
    string password,
    string storedHashB64,
    byte[] salt,
    int iterations)
{
    if (salt == null || salt.Length < 8) return false;
    if (iterations < 10_000) return false;
    byte[] expected = Convert.FromBase64String(storedHashB64);
    byte[] actual;
    try
    {
        using (var pbkdf2 = new Rfc2898DeriveBytes(
            password,
            salt,
            iterations,
            HashAlgorithmName.SHA256))
        {
            actual = pbkdf2.GetBytes(expected.Length);
        }
    }
    catch (MissingMethodException)
    {
        using (var pbkdf2 = new Rfc2898DeriveBytes(password, salt,
iterations))
        {
            actual = pbkdf2.GetBytes(expected.Length);
        }
    }
    return FixedTimeEquals(expected, actual);
}
```

В методе используется алгоритм PBKDF2. Для формирования хэша применяется класс `Rfc2898DeriveBytes`. В основной ветке используется алгоритм SHA256. Дополнительно предусмотрена обработка ситуации, когда

используемая версия платформы не поддерживает конструктор с указанием алгоритма хэширования. В таком случае используется запасной вариант конструктора.

Для сравнения хэшей используется отдельный метод `FixedTimeEquals`. Он выполняет сравнение массивов байтов без досрочного выхода из цикла. Такой подход снижает вероятность получения информации о совпадении части хэша по времени выполнения операции.

Метод сравнения хэшей представлен в листинге 22.

Листинг 22 – Сравнение хэшей пароля

```
private static bool FixedTimeEquals(byte[] a, byte[] b)
{
    if (a == null || b == null || a.Length != b.Length) return
false;
    int diff = 0;
    for (int i = 0; i < a.Length; i++)
        diff |= a[i] ^ b[i];
    return diff == 0;
}
```

Если пароль введен неверно, пользователю выводится сообщение об ошибке. При успешной проверке пароля создается объект главного окна `MainForm`. В главное окно передаются идентификатор пользователя, идентификатор роли и ФИО пользователя. Эти данные далее используются для настройки доступных функций системы.

После открытия главного окна форма авторизации скрывается. Это позволяет сохранить данные текущего сеанса и не завершать приложение сразу после перехода к основному окну.

В модуле авторизации также реализована обработка ошибок подключения к базе данных. Если при выполнении запроса возникает исключение, приложение выводит пользователю сообщение об ошибке подключения.

Модуль авторизации обеспечивает проверку логина и пароля пользователя, контроль активности учетной записи, безопасную проверку пароля по хэшу и передачу данных авторизованного пользователя в главное окно приложения.

3.3 Реализация главного окна приложения

Главное окно приложения реализовано в форме MainForm. Данная форма открывается после успешной авторизации пользователя и является основным средством навигации по информационной системе.

Интерфейс главного окна приложения представлен на рисунке 22.

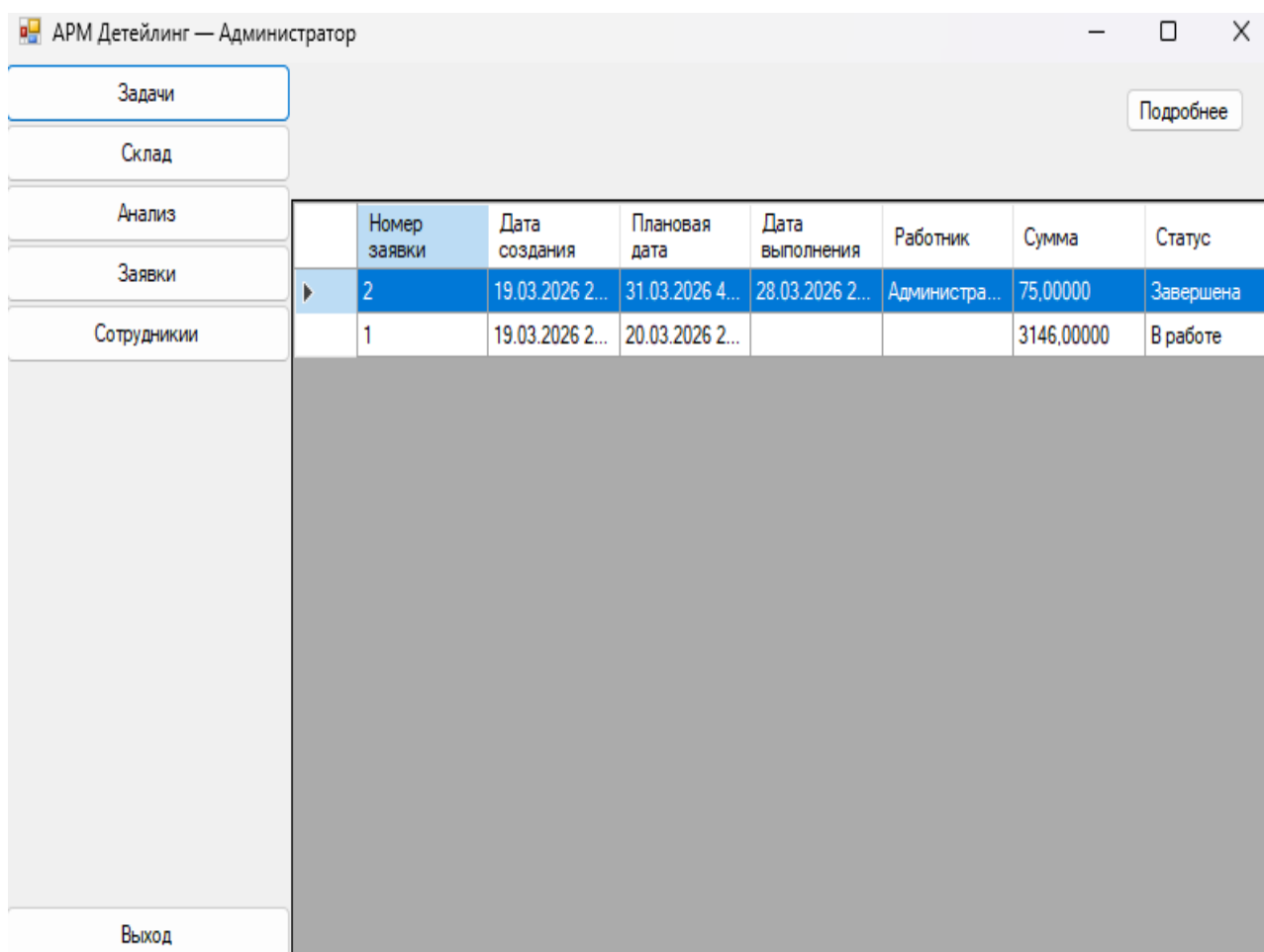


Рисунок 22 – Главное окно информационной системы

В конструктор формы MainForm передаются идентификатор пользователя, идентификатор роли и ФИО авторизованного сотрудника. Эти данные используются для отображения информации о текущем пользователе и ограничения доступа к отдельным разделам системы.

В главном окне реализованы следующие разделы:

- а) задачи;
- б) заявки;

- в) склад;
- г) анализ;
- д) сотрудники.

Раздел задач используется для просмотра заказ-нарядов. В нем отображаются данные о заказе, клиенте, автомобиле, статусе, ответственном сотруднике и плановых сроках выполнения работ. При выборе записи пользователь может открыть карточку заказ-наряда для просмотра подробной информации и внесения изменений.

Раздел заявок предназначен для просмотра клиентских заявок. Пользователь может создать новую заявку, принять заявку в работу, перевести ее в ожидание или отклонить. После принятия заявки в работу в базе данных создаются связанные записи осмотра, сметы и заказ-наряда.

Раздел склада используется для просмотра материалов и их текущих остатков. В таблице отображаются наименование материала, единица измерения, цена, минимальный остаток, текущий остаток и статус. Если остаток материала меньше минимального значения, система отображает предупреждающий статус.

Раздел анализа предназначен для просмотра сводной информации по работе детейлинг-центра. В нем отображаются данные, необходимые для оценки количества заказов, выручки и состояния выполняемых работ.

Раздел сотрудников доступен пользователю с ролью администратора. Через данный раздел выполняется переход к форме управления сотрудниками.

Доступность элементов главного окна зависит от роли пользователя. Например, раздел сотрудников отображается только администратору, а раздел склада доступен администратору и сотруднику склада.

Для отображения состава программных модулей приложения была разработана диаграмма компонентов. Она показывает взаимосвязь формы авторизации, главного окна, модулей заявок, задач, склада, сотрудников, отчетности и базы данных. Диаграмма компонентов информационной системы представлена на рисунке 23.

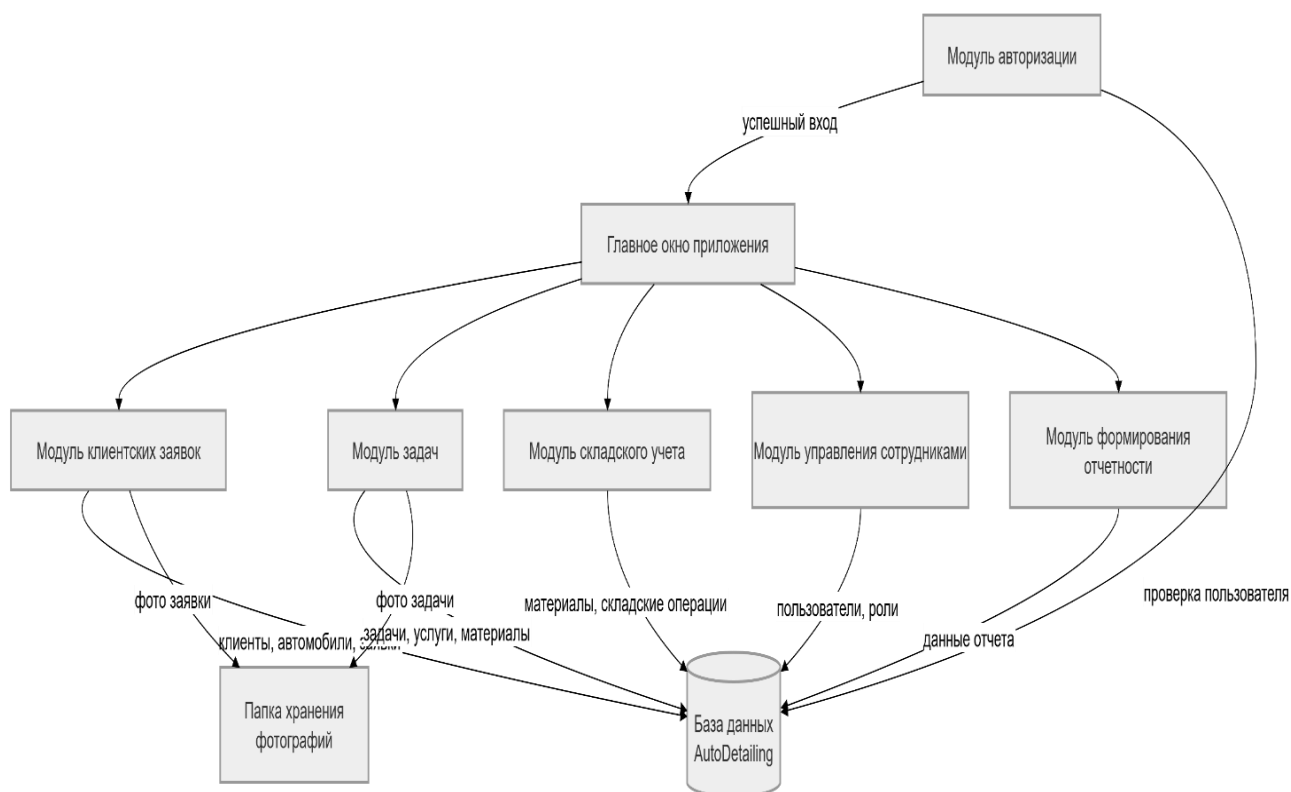


Рисунок 23 – Диаграмма компонентов информационной системы

Настройка интерфейса в зависимости от роли пользователя представлена в листинге 23.

Листинг 23 – Настройка интерфейса главного окна по роли пользователя

```

private void ApplyRoleUi()
{
    if (btnSupplies != null)
    {
        btnSupplies.Visible = (_role == 1 || _role == 4);
        btnSupplies.Text = «Склад»;
    }
    if (btnEmployees != null)
        btnEmployees.Visible = (_role == 1);
}
  
```

Для отображения данных в главном окне используется элемент DataGridView. При выборе раздела выполняется соответствующий SQL-запрос, после чего результат загружается в таблицу. Такой подход позволяет использовать одно табличное представление для разных разделов системы.

Интерфейс раздела заявок представлен на рисунке 24.

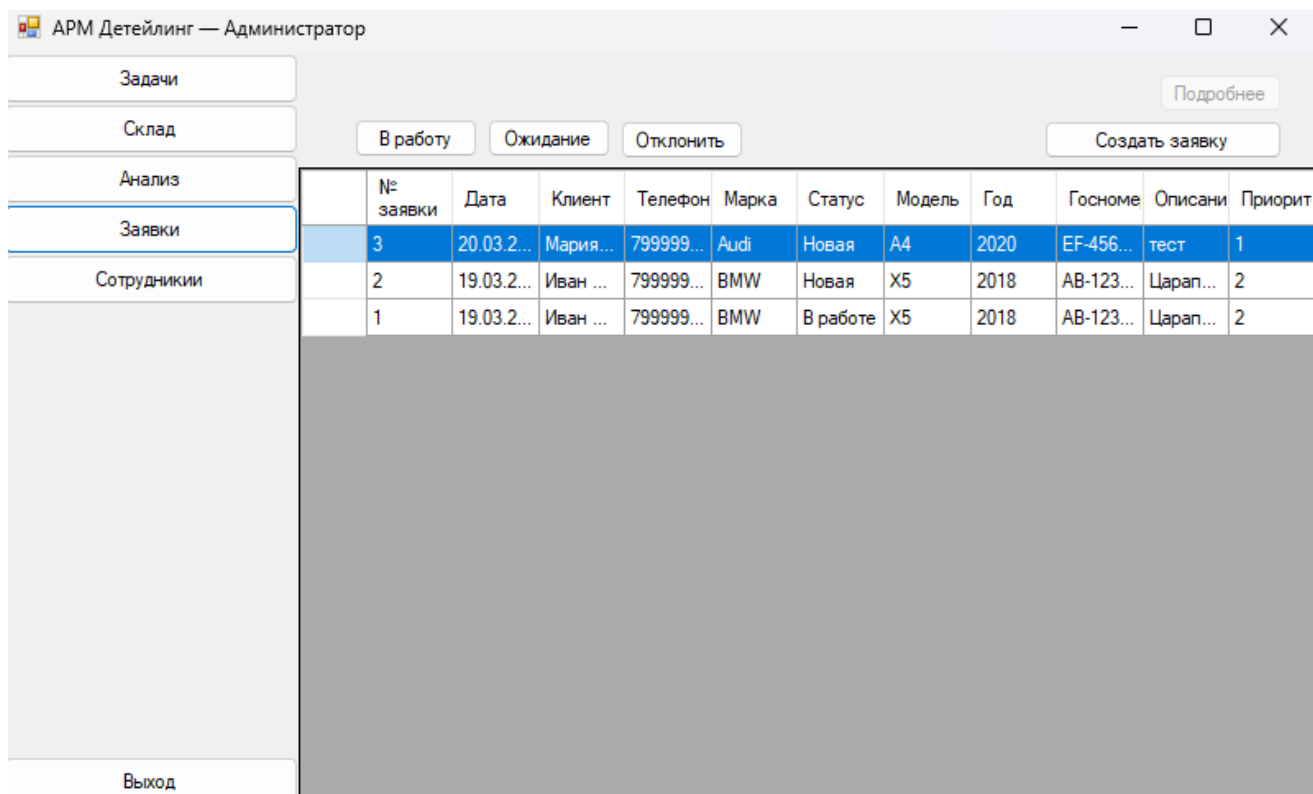


Рисунок 24 – Раздел клиентских заявок

Интерфейс раздела задач представлен на рисунке 25.

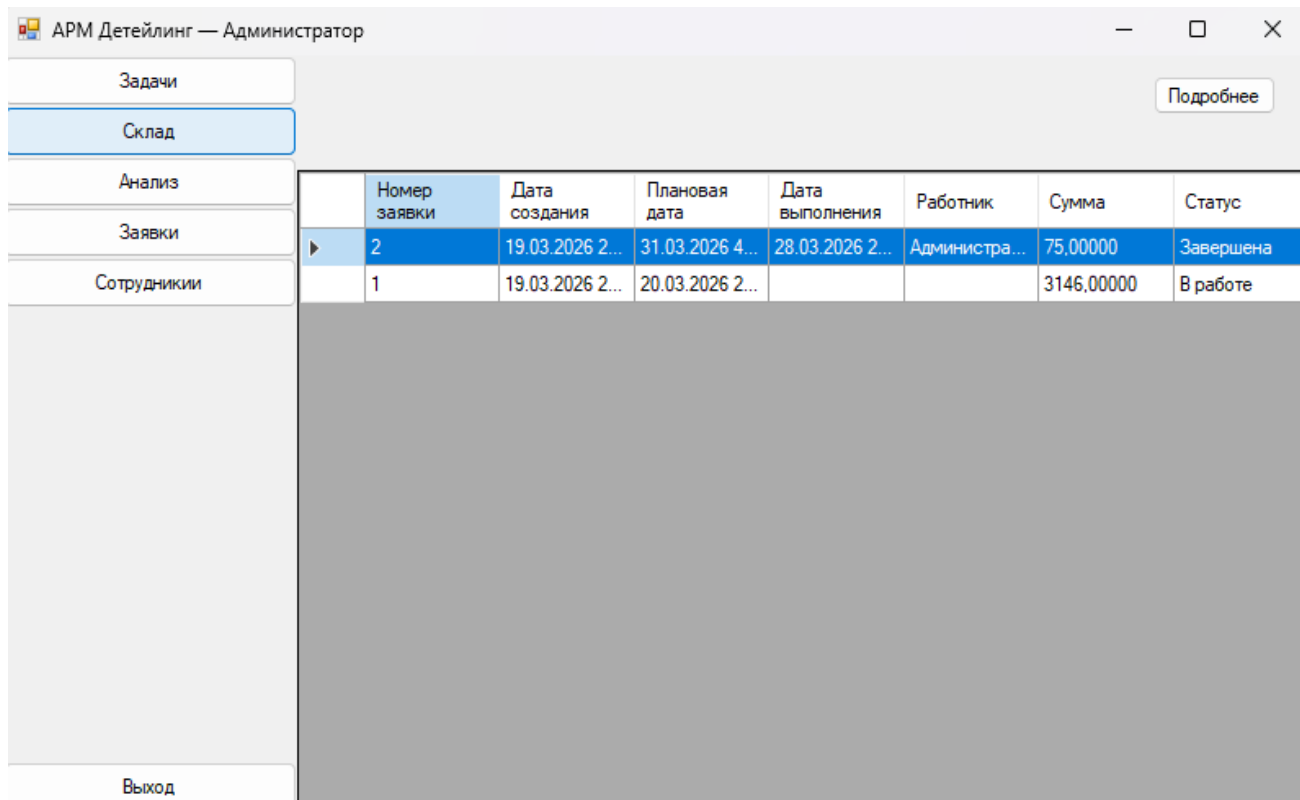


Рисунок 25 – Раздел задач

Интерфейс раздела склада представлен на рисунке 26.

Материал	Ед. изм.	Цена	Мин. остаток	Остаток	Статус
тест	шт	35,00	5,000	3,000	Ниже минимума
Антигравийна...	м²	35,00	5,000	19,000	Ок
Антидождь (со...	мл	0,80	300,000	2000,000	Ок
Глина для очи...	шт	18,00	5,000	50,000	Ок
Грунт (азрозоль)	шт	10,00	6,000	50,000	Ок
Керамика (сос...	мл	2,50	200,000	1950,000	Ок
Пак (азрозоль)	шт	12,00	6,000	50,000	Ок
Микрофибра 4...	шт	2,00	30,000	49,000	Ок
Наждачная бу...	шт	1,50	20,000	50,000	Ок
Обезжиривате...	л	10,00	2,000	5,000	Ок
Очиститель би...	л	12,00	1,000	5,000	Ок
Очиститель на...	л	9,00	1,000	5,000	Ок
Очиститель са...	л	11,00	2,000	5,000	Ок
Полировальна...	л	28,00	1,000	4,000	Ок
Полировальна...	л	28,00	1,000	4,000	Ок
Полировальны...	шт	9,00	10,000	50,000	Ок

Рисунок 26 – Раздел складского учета

При открытии карточки задачи из главного окна выбранная запись передается в форму TaskDetailsForm. После закрытия карточки данные в главном окне обновляются, чтобы пользователь видел актуальное состояние заказов.

Главное окно объединяет основные функции приложения и обеспечивает переход к созданию заявок, обработке заказ-нарядов, складскому учету, управлению сотрудниками и формированию аналитических данных.

3.4 Реализация работы с клиентскими заявками

Работа с клиентскими заявками реализована в форме RequestCreateForm. Данная форма предназначена для создания новой заявки на выполнение работ в детейлинг-центре. В форме указываются данные клиента, автомобиля, описание обращения и фотографии автомобиля.

При открытии формы выполняется подготовка элементов интерфейса, загрузка списка клиентов и настройка области для работы с фотографиями.

Пользователь может выбрать существующего клиента из списка или ввести данные нового клиента.

Интерфейс формы создания клиентской заявки представлен на рисунке 27.

Рисунок 27 – Форма создания клиентской заявки

При выборе клиента из списка приложение загружает его данные из базы данных: ФИО, паспортные данные, телефон и электронную почту. Также загружается список автомобилей, связанных с данным клиентом. Если у клиента есть несколько автомобилей, пользователь может переключаться между ними и выбрать нужный автомобиль для заявки.

Если клиент отсутствует в базе данных, его данные вводятся вручную. При сохранении заявки приложение создает новую запись о клиенте в таблице Client. Если клиент уже существует, его данные при необходимости обновляются.

Для автомобиля указываются марка, модель, год выпуска, VIN и государственный регистрационный номер. Если автомобиль уже был добавлен ранее, приложение использует существующую запись. Если такого автомобиля нет, создается новая запись в таблице Vehicle.

Перед сохранением заявки выполняется проверка данных клиента, автомобиля и описания обращения. При некорректном заполнении заявка не сохраняется, а пользователю выводится сообщение об ошибке.

Сохранение заявки выполняется как единая операция. Сначала приложение определяет статус «Новая», затем сохраняет или обновляет данные клиента и автомобиля. После этого создается запись в таблице Request, где фиксируются клиент, автомобиль, описание обращения, приоритет и статус заявки.

После создания заявки в таблицу RequestStatusHistory добавляется запись о начальном статусе. Это позволяет хранить историю изменения состояния заявки с момента ее создания.

В форме предусмотрено прикрепление фотографий автомобиля. Пользователь выбирает изображения через диалоговое окно. Выбранные фотографии отображаются в списке и могут быть просмотрены в области предварительного просмотра.

Интерфейс списка клиентских заявок представлен на рисунке 28.

№ заявки	Дата	Клиент	Телефон	Марка	Статус	Модель	Год	Госноме	Описани	Приорит
3	20.03.2...	Мария...	799999...	Audi	Новая	A4	2020	EF-456...	тест	1
2	19.03.2...	Иван ...	799999...	BMW	Новая	X5	2018	AB-123...	Царап...	2
1	19.03.2...	Иван ...	799999...	BMW	В работе	X5	2018	AB-123...	Царап...	2

Рисунок 28 – Список клиентских заявок

При сохранении заявки фотографии копируются в отдельную папку, связанную с номером заявки. В базе данных сохраняются не сами изображения, а пути к файлам. Для этого используется таблица RequestPhoto.

Если при сохранении клиента, автомобиля, заявки, истории статуса или фотографий возникает ошибка, операция отменяется. Это позволяет избежать ситуации, когда часть данных была сохранена, а часть потеряна.

После успешного сохранения заявка получает статус «Новая» и становится доступна в разделе заявок главного окна приложения. Далее пользователь может принять заявку в работу, перевести ее в ожидание или отклонить.

3.5 Реализация работы с задачами

Работа с задачами реализована в форме TaskDetailsForm. В приложении задача представляет собой заказ на выполнение работ по клиентской заявке. После принятия заявки в работу для нее создается задача, в которой фиксируются сведения о клиенте, автомобиле, описании обращения, услугах, материалах, сроках выполнения и результате работ.

Интерфейс карточки задачи представлен на рисунке 29.

Карточка заказа-наряда №2

№ заказ-наряда: 2
 Создан: 19.03.2026 22:29
 План. начало: 19.03.2026 22:29
 Факт. окончание: 28.03.2026 23:15
 Исполнитель: Администратор
 Клиент: Иван Петров
 Телефон: 79999999999
 Автомобиль: BMW X5, 2018, AB-123-CD
 Сумма: 75.00
 Описание заявки: Царапины на двери, нужна полировка + керамика (2 слоя).

Статус: Завершена

Установка план. даты окончания
 31.03.2026 04:13 Сохранить

Тип	Наименование	Кол-во	Цена	Итого
Услуга	Антидождь (стекла)	1,000	40,00	40,00000
Материал	Антигравийная плёнка (...)	1,000	35,00	35,00000

Фото
 c1afdd51b8fa4e
 Добавить
 Удалить

Добавить услугу Добавить материал Удалить строку Завершить Осмотр Отчёт Закрыть

Рисунок 29 – Карточка задачи

Карточка задачи открывается из главного окна приложения при выборе записи в разделе задач. В форме отображаются основные сведения по выбранной задаче: номер, статус, данные клиента, автомобиль, описание заявки, ответственный сотрудник, плановая дата завершения и фактическое завершение.

В форме предусмотрена работа с услугами. Пользователь может добавить услугу из справочника, указать количество и сохранить ее в составе задачи. После добавления услуга отображается в таблице услуг. При необходимости позицию можно удалить.

Для выбора услуги используется отдельная форма PartPickForm. В ней отображаются записи из справочника ServiceCatalog. После выбора услуга передается обратно в карточку задачи и добавляется в таблицу WorkOrderService.

Интерфейс выбора услуги представлен на рисунке 30.

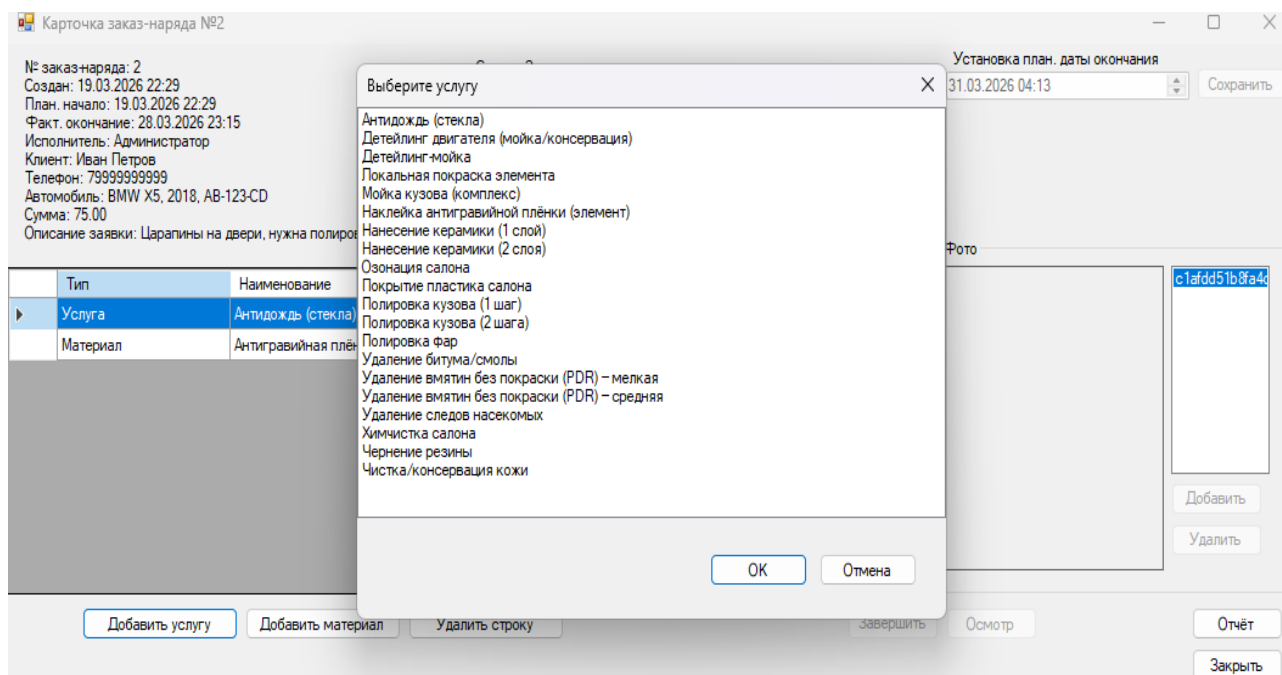


Рисунок 30 – Форма выбора услуги

В карточке задачи также реализована работа с материалами. Пользователь выбирает материал из справочника, указывает количество, после чего материал добавляется в задачу. Одновременно формируется операция списания материала со склада. Это позволяет учитывать расход материалов при выполнении работ.

Для хранения материалов, добавленных в задачу, используется таблица WorkOrderMaterial. Складская операция сохраняется в таблицу StockMovement. Приходы увеличивают остаток материала, а списания уменьшают его.

Интерфейс добавления материалов в задачу представлен на рисунке 31.

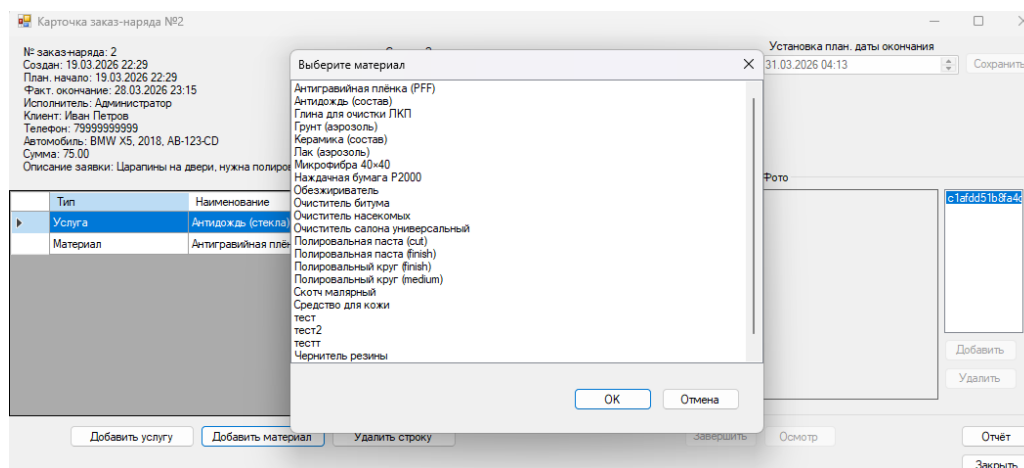


Рисунок 31 – Добавление материалов в задачу

В форме задачи выполняется расчет стоимости. Итоговая сумма складывается из стоимости добавленных услуг и материалов с учетом скидки. После изменения состава услуг или материалов данные в форме обновляются, чтобы пользователь видел актуальную стоимость работ.

Для фиксации результата работ в задаче предусмотрено добавление фотографий. Пользователь выбирает изображение, после чего файл сохраняется в папку приложения, а путь к нему записывается в таблицу WorkOrderPhoto. В форме можно просмотреть добавленные фотографии и удалить ненужные изображения.

Интерфейс работы с фотографиями задачи представлен на рисунке 32.

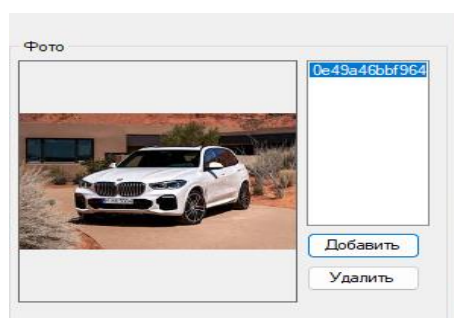


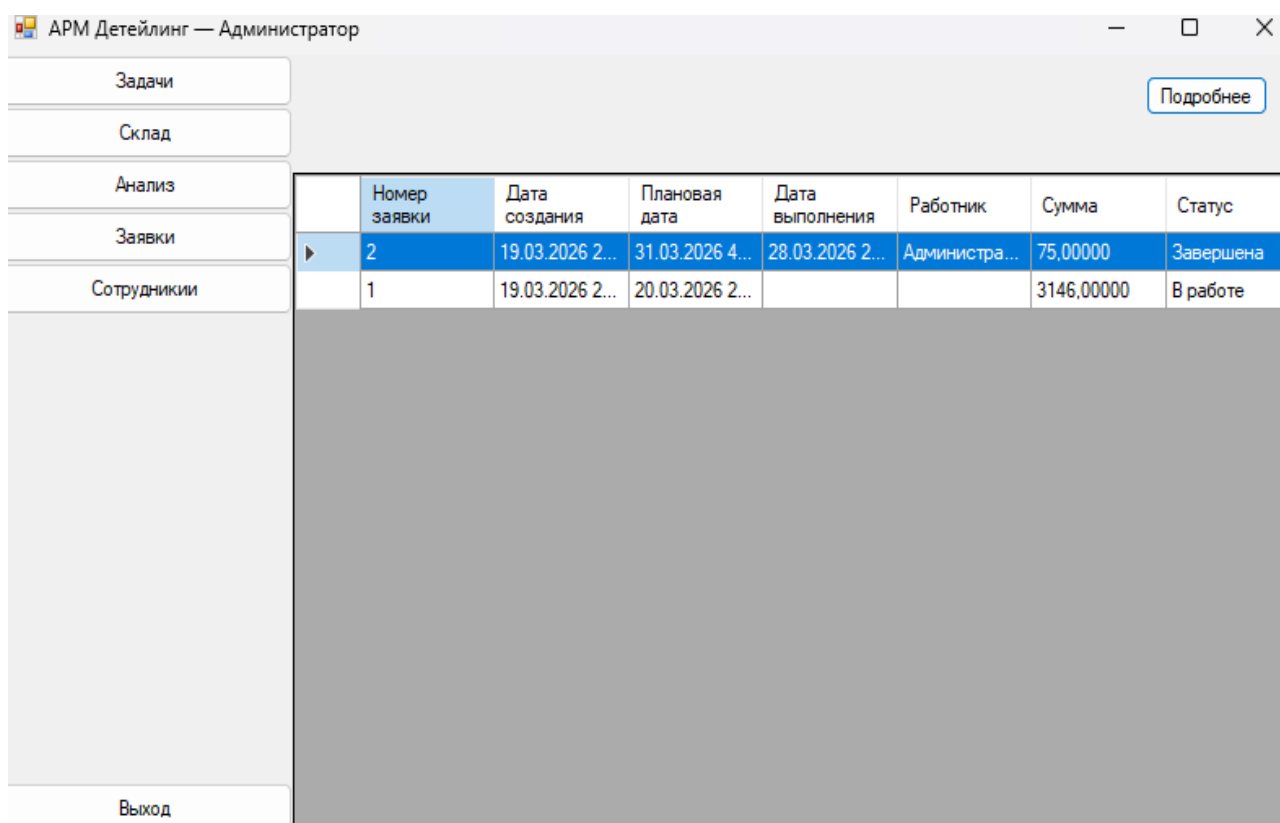
Рисунок 32 – Фотографии результата работ

В карточке задачи пользователь может указать плановую дату завершения. Это позволяет контролировать срок выполнения работ. После сохранения выбранная дата записывается в соответствующее поле задачи.

После выполнения всех работ задача может быть завершена. При завершении задачи приложение записывает фактическую дату окончания работ, меняет статус на «Завершена» и добавляет запись в историю статусов. История изменения статусов хранится в таблице WorkOrderStatusHistory.

В форме также предусмотрено сохранение текстового отчета по задаче. Отчет формируется на основе данных клиента, автомобиля, выполненных услуг, использованных материалов, стоимости и результата работ. Такой отчет может быть использован для передачи клиенту или для внутреннего учета.

Раздел задач в главном окне представлен на рисунке 33.



Номер заявки	Дата создания	Плановая дата	Дата выполнения	Работник	Сумма	Статус
2	19.03.2026 2...	31.03.2026 4...	28.03.2026 2...	Администра...	75,00000	Завершена
1	19.03.2026 2...	20.03.2026 2...			3146,00000	В работе

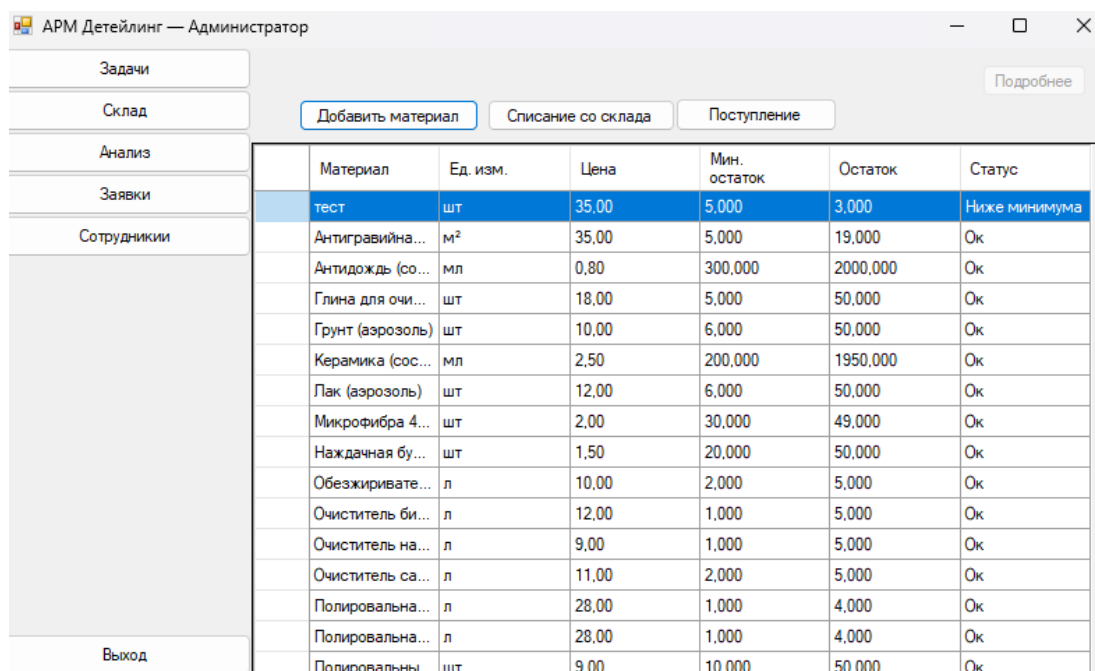
Рисунок 33 – Раздел задач

Реализованная работа с задачами позволяет вести полный цикл выполнения работ: от принятия заявки в работу до завершения задачи, списания материалов, фиксации фотографий результата и формирования отчета.

3.6 Реализация складского учета

Складской учет реализован в главном окне приложения и связан с таблицами MaterialCatalog и StockMovement. Раздел склада предназначен для просмотра материалов, контроля остатков, добавления новых материалов, а также фиксации операций прихода и списания.

Интерфейс раздела складского учета представлен на рисунке 34.



Материал	Ед. изм.	Цена	Мин. остаток	Остаток	Статус
тест	шт	35,00	5,000	3,000	Ниже минимума
Антигравийна...	м²	35,00	5,000	19,000	Ок
Антидождь (со...	мл	0,80	300,000	2000,000	Ок
Глина для очи...	шт	18,00	5,000	50,000	Ок
Грунт (аэрозоль)	шт	10,00	6,000	50,000	Ок
Керамика (сос...	мл	2,50	200,000	1950,000	Ок
Лак (аэрозоль)	шт	12,00	6,000	50,000	Ок
Микрофибра 4...	шт	2,00	30,000	49,000	Ок
Наждачная бу...	шт	1,50	20,000	50,000	Ок
Обезжиривате...	л	10,00	2,000	5,000	Ок
Очиститель би...	л	12,00	1,000	5,000	Ок
Очиститель на...	л	9,00	1,000	5,000	Ок
Очиститель са...	л	11,00	2,000	5,000	Ок
Полировальна...	л	28,00	1,000	4,000	Ок
Полировальна...	л	28,00	1,000	4,000	Ок
Полировальны...	шт	9,00	10,000	50,000	Ок

Рисунок 34 – Раздел складского учета

В разделе склада отображается список материалов. Для каждого материала выводятся наименование, единица измерения, розничная цена, минимальный остаток, текущий остаток и статус. Текущий остаток рассчитывается на основании складских операций. Приход материала увеличивает остаток, а списание уменьшает его.

Для хранения справочника материалов используется таблица MaterialCatalog. В ней фиксируются название материала, единица измерения, розничная цена, текущий остаток и минимальный допустимый остаток. Минимальный остаток используется для контроля нехватки материалов.

Для хранения складских операций используется таблица StockMovement. В данной таблице фиксируются материал, тип операции, количество, цена

единицы, причина операции, связанная задача и дата создания операции. Тип операции определяет, является запись приходом или списанием.

Если текущий остаток материала становится меньше минимального остатка, в разделе склада отображается статус «Ниже минимума». Это позволяет сотруднику склада своевременно заметить нехватку материала и выполнить пополнение.

Добавление нового материала реализовано в форме MaterialCreateForm. В форме пользователь указывает название материала, единицу измерения, розничную цену, минимальный остаток и начальное количество.

Интерфейс формы добавления материала представлен на рисунке 35.

Рисунок 35 – Форма добавления материала

После сохранения нового материала приложение добавляет запись в таблицу MaterialCatalog. Если пользователь указал начальное количество, дополнительно создается складская операция прихода в таблице StockMovement. За счет этого начальный остаток материала также учитывается через складские движения.

В разделе склада предусмотрены операции прихода и списания материала. При выполнении операции пользователь выбирает материал, указывает

количество и причину движения. После подтверждения в таблицу StockMovement добавляется новая запись.

Операция прихода используется при пополнении склада. Операция списания используется при уменьшении остатка материала. Кроме ручного списания, списание также выполняется при добавлении материала в задачу. В этом случае складская операция связывается с конкретной задачей.

Реализованный складской учет обеспечивает хранение справочника материалов, контроль текущих остатков, учет прихода и списания, а также связь расхода материалов с выполняемыми задачами.

3.7 Реализация управления сотрудниками

Управление сотрудниками реализовано в форме EmployeesForm. Данный раздел доступен пользователю с ролью администратора и предназначен для добавления новых сотрудников, редактирования учетных записей и изменения статуса пользователя.

Интерфейс формы управления сотрудниками представлен на рисунке 36.

Рисунок 36 – Форма управления сотрудниками

При открытии формы загружается список сотрудников из таблицы User и список ролей из таблицы Role. В списке сотрудников отображаются ФИО, логин и статус учетной записи. При выборе сотрудника его данные подставляются в поля формы для редактирования.

В форме администратор может указать ФИО сотрудника, логин, роль, пароль и признак увольнения. Если создается новый сотрудник, пароль является обязательным. Если редактируется существующий сотрудник, пароль можно не менять.

Перед сохранением выполняется проверка заполнения обязательных полей. Также проверяется уникальность логина, чтобы в системе не было двух сотрудников с одинаковыми учетными данными.

Сохранение сотрудника выполняется в транзакции. Если создается новый пользователь, в таблицу User добавляется новая запись. Если редактируется существующий пользователь, обновляются его ФИО, логин, роль и признак активности.

Для хранения пароля используется хэширование. При установке нового пароля формируется соль, затем вычисляется хэш пароля с использованием заданного количества итераций. В таблицу User сохраняются хэш пароля, соль и количество итераций. Сам пароль в базе данных не хранится.

Если сотрудник больше не работает в организации, администратор может отметить его как уволенного. В этом случае значение активности учетной записи изменяется, и пользователь не сможет пройти авторизацию в системе.

Реализованный модуль управления сотрудниками позволяет администратору поддерживать актуальный список пользователей системы, назначать роли и ограничивать доступ неактивным учетным записям.

3.8 Реализация формирования отчетности

Формирование отчетности реализовано в форме ReportPeriodForm. Данная форма предназначена для получения отчета по завершенным задачам за выбранный период.

Интерфейс формы формирования отчета представлен на рисунке 37.

№ заказа	Создан	План. окончан	Заверше	Исполни	Клиент	Телефон	Автомоб	Описание	Услуг	Матери	Сумма услуг	Сумма матери	Итого	Дней в работе
2	19.03....	31.03....	28.03....	Админ...	Иван ...	79999...	BMW ...	Царап...	1	1	40,00	35,00	75,00	10

Рисунок 37 – Форма формирования отчета

При открытии формы в качестве начальной даты автоматически устанавливается первый день текущего месяца, а в качестве конечной даты устанавливается текущая дата. Пользователь может изменить период вручную с помощью элементов выбора даты.

После нажатия кнопки формирования отчета выполняется проверка выбранного периода. Если начальная дата больше конечной, отчет не формируется, а пользователю выводится сообщение об ошибке.

Отчет строится на основании данных о завершенных задачах. В выборку попадают только те задачи, у которых заполнена фактическая дата завершения. Для каждой задачи отображаются номер, дата создания, плановая дата окончания, дата завершения, исполнитель, клиент, телефон, автомобиль, описание, количество услуг, количество материалов, сумма услуг, сумма материалов, итоговая стоимость и количество дней в работе.

Основной запрос для формирования отчета представлен в листинге 24.

Листинг 24 – Формирование отчета по завершенным задачам

```

const string sql = @»
SELECT
    wo.Id AS [№ заказ-наряда],
    wo.CreatedAt AS [Создан],
    wo.PlannedFinish AS [План. окончание],
    wo.ActualFinish AS [Завершён],
    u.FullName AS [Исполнитель],
    c.FullName AS [Клиент],
    c.Phone AS [Телефон],
    (v.Brand + N' ' + v.Model + N' (' + v.RegPlate + N')') AS
[Автомобиль],
    req.Description AS [Описание],
    ISNULL(svc.ServiceCount, 0) AS [Услуг],
    ISNULL(mat.MaterialCount, 0) AS [Материалов],
    CAST(ISNULL(svc.ServiceSum, 0) AS decimal(18,2)) AS [Сумма
услуг],
    CAST(ISNULL(mat.MaterialSum, 0) AS decimal(18,2)) AS [Сумма
материалов],
    CAST(ISNULL(svc.ServiceSum, 0) + ISNULL(mat.MaterialSum, 0) -
ISNULL(wo.Discount, 0) AS decimal(18,2)) AS [Итого],
CASE
    WHEN wo.ActualFinish IS NULL THEN NULL
    ELSE DATEDIFF(DAY, CAST(wo.CreatedAt AS date),
CAST(wo.ActualFinish AS date))+1
END AS [Дней в работе]
FROM dbo.WorkOrder wo
JOIN dbo.Request req ON req.Id = wo.RequestId
JOIN dbo.Client c ON c.Id = req.ClientId
JOIN dbo.Vehicle v ON v.Id = req.VehicleId
LEFT JOIN dbo.[User] u ON u.Id = wo.ResponsibleUserId
OUTER APPLY
(
    SELECT
        COUNT(*) AS ServiceCount,
        SUM(wos.Qty * wos.Price) AS ServiceSum
    FROM dbo.WorkOrderService wos
    WHERE wos.WorkOrderId = wo.Id
) svc
OUTER APPLY
(
    SELECT
        COUNT(*) AS MaterialCount,
        SUM(wom.Qty * wom.Price) AS MaterialSum
    FROM dbo.WorkOrderMaterial wom
    WHERE wom.WorkOrderId = wo.Id
) mat
WHERE wo.ActualFinish IS NOT NULL
    AND CAST(wo.ActualFinish AS date) BETWEEN @from AND @to
ORDER BY wo.ActualFinish DESC, wo.Id DESC;»

```

В запросе используются данные из таблиц WorkOrder, Request, Client, Vehicle, User, WorkOrderService и WorkOrderMaterial. Сумма услуг и сумма материалов рассчитываются отдельно, после чего определяется итоговая стоимость задачи с учетом скидки.

После загрузки данных отчет отображается в таблице DataGridView. Пользователь может просмотреть список завершенных задач и оценить основные показатели за выбранный период.

Кроме основной таблицы, в форме рассчитываются итоговые показатели: количество завершенных задач, общая выручка и средний чек. Эти данные отображаются в нижней части формы.

Сформированный отчет можно сохранить в файл CSV. При сохранении в файл записываются название отчета, итоговые показатели, заголовки столбцов и строки отчета. Данный файл можно открыть в табличном редакторе и использовать для дальнейшего анализа.

Интерфейс сформированного отчета представлен на рисунке 38.

C20															
	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
1	Отчет за период 01.03.2026 - 30.05.2026														
2	Завершено заказов: 1 Выручка: 75,00 Средний чек: 75,00														
3															
4	№ заказ-	Создан	План. окончание	Завершён	Исполнитель	Клиент	Телефон	Автомоби	Описани	Услуг	Материалов	Сумма услуг	Сумма материалов	Итого	Дней в работе
5	2	19.03.2026 22:29	31.03.2026 4:13	28.03.2026 23:15	Администратор	Иван Петров	79999999999	BMW X5 (4	Царапин	1	1	40		35	75
6															10
7															
8															
9															

Рисунок 38 – Сформированный отчет за выбранный период

Реализованная отчетность позволяет получать сведения о выполненных задачах, контролировать выручку за период и анализировать результаты работы дилерского центра.

Фрагменты исходного кода приложения приведены в приложении Б.

Полный исходный код информационной системы размещен в удаленном репозитории GitHub, ссылка на который приведена в списке использованных источников [20].

4 Тестирование системы

4.1 Методика ручного тестирования

Тестирование информационной системы детейлинг-центра выполнялось вручную методом черного ящика. При данном подходе проверялась работа функций приложения с точки зрения пользователя, без написания автоматизированных тестов. Метод черного ящика основан на проверке соответствия работы программного продукта предъявляемым требованиям и спецификации. В процессе тестирования внимание уделялось корректности обработки входных данных, правильности выполнения пользовательских действий и соответствию получаемых результатов ожидаемому поведению системы.

В ходе тестирования проверялись основные сценарии работы системы:

- а) авторизация пользователя;
- б) разграничение доступа по ролям;
- в) создание клиентской заявки;
- г) прикрепление фотографий к заявке;
- д) принятие заявки в работу;
- е) открытие и редактирование задачи;
- ж) добавление услуг в задачу;
- з) добавление материалов в задачу;
- и) списание материалов со склада;
- к) завершение задачи;
- л) добавление нового материала;
- м) управление сотрудниками;
- н) формирование отчета за выбранный период;
- о) сохранение отчета в файл.

Для тестирования использовалась база данных AutoDetailing, заполненная тестовыми данными о клиентах, автомобилях, услугах, материалах, пользователях, ролях и статусах. Проверка выполнялась путем

последовательного выполнения пользовательских действий в интерфейсе приложения и сравнения фактического результата с ожидаемым.

Особое внимание уделялось проверке корректности сохранения данных в базе данных, отображению записей в таблицах приложения, обработке ошибок ввода и сохранению связей между заявками, задачами, клиентами, автомобилями, услугами и материалами.

4.2 Функциональное тестирование

Функциональное тестирование проводилось для проверки основных возможностей разработанной информационной системы. Результаты тестирования представлены в таблице 15.

Таблица 15 – Результаты функционального тестирования информационной системы

№	Проверяемая функция	Действия пользователя	Ожидаемый результат	Фактический результат	Статус
1	Авторизация с корректными данными	Ввести существующий логин и пароль, нажать кнопку входа	Открывается главное окно приложения	Главное окно открывается	Пройден
2	Авторизация с неверным паролем	Ввести существующий логин и неверный пароль	Отображается сообщение об ошибке входа	Сообщение отображается	Пройден
3	Проверка пустых полей авторизации	Не заполнить логин или пароль и нажать кнопку входа	Отображается предупреждение о необходимости ввода данных	Предупреждение отображается	Пройден
4	Ограничение доступа по роли	Войти под пользователем без прав администратора	Недоступные разделы не отображаются	Интерфейс изменяется согласно роли	Пройден
5	Создание новой заявки	Заполнить данные клиента, автомобиля и описание обращения	Заявка сохраняется со статусом «Новая»	Заявка создана и отображается в списке	Пройден
6	Проверка обязательных полей заявки	Оставить обязательные поля пустыми и сохранить заявку	Сохранение не выполняется, отображается ошибка	Ошибка отображается	Пройден

Продолжение таблицы 15

№	Проверяемая функция	Действия пользователя	Ожидаемый результат	Фактический результат	Статус
7	Прикрепление фотографии к заявке	Добавить изображение при создании заявки	Файл сохраняется, путь записывается в базу данных	Фотография сохраняется и отображается	Пройден
8	Принятие заявки в работу	Выбрать заявку и нажать кнопку принятия в работу	Создается задача, заявка меняет статус	Задача создается, статус обновляется	Пройден
9	Открытие карточки задачи	Выбрать задачу в списке и открыть ее	Открывается форма с данными задачи	Карточка задачи открывается	Пройден
10	Добавление услуги в задачу	Выбрать услугу из справочника и указать количество	Услуга добавляется в задачу, сумма пересчитывается	Услуга добавлена, сумма обновлена	Пройден
11	Добавление материала в задачу	Выбрать материал и указать количество	Материал добавляется в задачу, выполняется списание со склада	Материал добавлен, складское движение создано	Пройден
12	Добавление нового материала	Заполнить форму материала и сохранить	Материал появляется в справочнике склада	Материал добавлен в список	Пройден
13	Контроль минимального остатка	Уменьшить остаток материала ниже минимума	В разделе склада отображается статус «Ниже минимума»	Статус отображается	Пройден
14	Завершение задачи	Нажать кнопку завершения задачи	У задачи заполняется фактическая дата завершения, статус меняется на «Завершена»	Задача завершена	Пройден
15	Управление сотрудниками	Добавить нового сотрудника с ролью	Пользователь сохраняется и отображается в списке	Сотрудник добавлен	Пройден
16	Деактивация сотрудника	Отметить сотрудника как неактивного	Пользователь не может пройти авторизацию	Неактивная учетная запись не входит в систему	Пройден
17	Формирование отчета	Выбрать период и сформировать отчет	Отображается список завершенных задач за период	Отчет сформирован	Пройден

Продолжение таблицы 15

№	Проверяемая функция	Действия пользователя	Ожидаемый результат	Фактический результат	Статус
18	Проверка некорректного периода отчета	Указать начальную дату позже конечной	Отчет не формируется, отображается ошибка	Ошибка отображается	Пройден
19	Сохранение отчета в файл	Сформировать отчет и сохранить его в CSV	Файл отчета создается	Файл сохраняется	Пройден
20	Обновление данных после операций	Выполнить создание или изменение записи	Список данных обновляется после операции	Данные обновляются	Пройден

По результатам функционального тестирования основные функции информационной системы работают корректно. Ошибок, препятствующих выполнению основных пользовательских сценариев, выявлено не было.

4.3 Тестирование обработки ошибок

При ручном тестировании также проверялась реакция приложения на некорректные действия пользователя. Проверялись пустые поля, неверный формат данных, некорректный период отчета, попытка входа с неверными учетными данными и работа с неактивной учетной записью.

Результаты проверки обработки ошибок представлены в таблице 16.

Таблица 16 – Проверка обработки ошибок

№	Ситуация	Ожидаемая реакция системы	Результат
1	Не введен логин или пароль	Вывод предупреждения	Предупреждение выводится
2	Введен неверный пароль	Отказ во входе в систему	Вход не выполняется
3	Учетная запись неактивна	Отказ во входе в систему	Вход не выполняется
4	Не заполнено описание заявки	Заявка не сохраняется	Сохранение блокируется
5	Некорректный телефон клиента	Вывод сообщения об ошибке	Ошибка выводится
6	Некорректный VIN автомобиля	Вывод сообщения об ошибке	Ошибка выводится
7	Начальная дата отчета больше конечной	Отчет не формируется	Формирование блокируется
8	Ошибка подключения к базе данных	Вывод сообщения об ошибке	Сообщение выводится

Обработка ошибок реализована через проверки входных данных и вывод сообщений пользователю.

Это позволяет снизить вероятность сохранения некорректных данных в базе данных.

Фрагменты исходного кода модульных тестов приведены в приложении В.

4.4 Метрики качества кода

Для оценки качества программного кода были рассмотрены количественные метрики исходных файлов приложения. Анализ выполнялся по основным файлам форм без учета автоматически сгенерированных файлов Designer.

Результаты оценки представлены в таблице 17.

Таблица 17 – Метрики исходного кода основных модулей

Файл	Назначение	Количество значимых строк кода	Оценка цикломатической сложности
Program.cs	Запуск приложения	18	1
LoginForm.cs	Авторизация пользователя	111	21
MainForm.cs	Главное окно приложения	626	86
RequestCreateForm.cs	Создание клиентской заявки	737	88
TaskDetailsForm.cs	Работа с задачей	867	102
MaterialCreateForm.cs	Добавление материала	134	18
PartPickForm.cs	Выбор услуги или материала	190	11
EmployeesForm.cs	Управление сотрудниками	322	28
ReportPeriodForm.cs	Формирование отчетности	336	51
Итого	Основные программные модули	3341	—

Наибольший объем кода имеют формы TaskDetailsForm, RequestCreateForm и MainForm. Это связано с тем, что данные формы реализуют основные бизнес-процессы системы: работу с задачами, создание заявок и навигацию по разделам приложения.

Цикломатическая сложность выше у модулей, где присутствует больше условий, обработчиков событий и вариантов пользовательских действий. Для снижения сложности код разделен по отдельным формам, каждая из которых отвечает за конкретную часть функциональности.

Код приложения имеет модульную структуру. Основные функции распределены между отдельными формами: авторизация, главное окно, заявки,

задачи, склад, сотрудники и отчетность. Такое разделение упрощает сопровождение приложения и позволяет изменять отдельные части системы без переработки всего проекта.

Заключение

В ходе выполнения выпускной квалификационной работы была разработана информационная система детейлинг-центра. Разработанное приложение предназначено для автоматизации учета клиентских заявок, автомобилей, задач, выполняемых услуг, используемых материалов, сотрудников и отчетности.

В первой части работы был выполнен анализ предметной области. Были рассмотрены особенности работы детейлинг-центра, определены основные участники процесса и выявлены задачи, которые требуют автоматизации. На основании анализа были сформулированы требования к разрабатываемой информационной системе.

Во второй части работы было выполнено проектирование системы. Были определены основные функции приложения, разработана структура базы данных, описаны сущности предметной области и связи между ними. Также были спроектированы программные модули системы и пользовательский интерфейс.

В третьей части работы была рассмотрена реализация программных средств. Была создана база данных AutoDetailing в Microsoft SQL Server, реализованы формы авторизации, главного окна, создания клиентской заявки, работы с задачами, складского учета, управления сотрудниками и формирования отчетности. Приложение было разработано на языке C# с использованием технологии Windows Forms.

В четвертой части работы было выполнено ручное тестирование системы. Проверялись основные пользовательские сценарии: вход в систему, создание заявки, принятие заявки в работу, работа с задачами, добавление услуг и материалов, списание материалов со склада, завершение задачи, управление сотрудниками и формирование отчета за выбранный период. По результатам тестирования основные функции системы работают корректно.

Разработанная информационная система позволяет сократить объем ручной работы сотрудников детейлинг-центра, упростить учет заявок и задач, обеспечить контроль складских остатков и получать отчетные данные за выбранный период.

Поставленная цель выпускной квалификационной работы достигнута. Задачи, связанные с анализом предметной области, проектированием, реализацией и тестированием информационной системы детейлинг-центра, выполнены.

Список использованных источников

- 1 Общероссийский классификатор видов экономической деятельности ОК 029-2014 (КДЕС Ред. 2). Код ОКВЭД 45.20.3. Мойка автотранспортных средств, полирование и предоставление аналогичных услуг [Электронный ресурс]. – URL: <https://classinform.ru/okved2/kod-45.20.3.html> (дата обращения: 09.05.2026).
- 2 Закон РФ от 07.02.1992 № 2300-1 «О защите прав потребителей». Статья 27. Сроки выполнения работ (оказания услуг) [Электронный ресурс]. – URL: https://www.consultant.ru/document/cons_doc_LAW_305/50865dba506be37ff44fa8df0819a8c598b25282/ (дата обращения: 09.05.2026).
- 3 Федеральный закон от 27.07.2006 № 152-ФЗ «О персональных данных». Статья 3. Основные понятия, используемые в настоящем Федеральном законе [Электронный ресурс]. – URL: https://legalacts.ru/doc/152_FZ-o-personalnyh-dannyh/glava-1/statja-3/ (дата обращения: 09.05.2026).
- 4 LiveSklad – CRM система для автосервиса и автоматизации СТО [Электронный ресурс]. – URL: <https://livesklad.com/crm-system-car-service> (дата обращения: 09.05.2026).
- 5 LiveSklad – CRM программа для учета клиентов в автосервисе [Электронный ресурс]. – URL: <https://livesklad.com/program-account-service-center> (дата обращения: 09.05.2026).
- 6 Clean-Control. Программа контроля и автоматизации для автомойки, детейлинг-центра, шиномонтажа, СТО [Электронный ресурс]. – URL: <https://clean-control.ru/> (дата обращения: 09.05.2026).
- 7 YCLIENTS. Программа для онлайн-записи и автоматизации автосервиса [Электронный ресурс]. – URL: <https://www.yclients.com/auto-service> (дата обращения: 09.05.2026).

8 1С:Альфа-Авто. Программа для автосервиса и учета запчастей [Электронный ресурс]. – URL: <https://1c-alfa-avto.ru/> (дата обращения: 09.05.2026).

9 CRM4Auto. Облачная CRM для автосервиса [Электронный ресурс]. – URL: <https://crm4auto.ru/> (дата обращения: 09.05.2026).

10 UML Resource Page. Unified Modeling Language (UML) [Электронный ресурс]. – Режим доступа: <https://www.uml.org/> (дата обращения: 10.05.2026).

11 Microsoft Learn. C# Programming Guide [Электронный ресурс]. – Режим доступа: <https://learn.microsoft.com/dotnet/csharp/programming-guide/> (дата обращения: 10.05.2026).

12 Microsoft Learn. Документация по языку C# [Электронный ресурс]. – URL: <https://learn.microsoft.com/ru-ru/dotnet/csharp/> (дата обращения: 10.05.2026).

13 Microsoft Learn. Документация по Windows Forms [Электронный ресурс]. – URL: <https://learn.microsoft.com/ru-ru/dotnet/desktop/winforms/> (дата обращения: 10.05.2026).

14 Microsoft Learn. ADO.NET [Электронный ресурс]. – URL: <https://learn.microsoft.com/ru-ru/dotnet/framework/data/adonet/> (дата обращения: 10.05.2026).

15 Microsoft Learn. Техническая документация SQL Server [Электронный ресурс]. – URL: <https://learn.microsoft.com/ru-ru/sql/sql-server/> (дата обращения: 11.05.2026).

16 Microsoft Learn. Справочник по Transact-SQL [Электронный ресурс]. – URL: <https://learn.microsoft.com/ru-ru/sql/t-sql/language-reference> (дата обращения: 11.05.2026).

17 Microsoft Learn. SQL Server Management Studio [Электронный ресурс]. – URL: <https://learn.microsoft.com/ru-ru/ssms/> (дата обращения: 11.05.2026).

18 Microsoft Learn. Документация по Visual Studio [Электронный ресурс]. – URL: <https://learn.microsoft.com/ru-ru/visualstudio/windows/> (дата обращения: 11.05.2026).

19 draw.io. Documentation [Электронный ресурс]. – URL: <https://www.drawio.com/doc/> (дата обращения: 12.05.2026).

20 Репозиторий исходного кода информационной системы детейлинг-центра [Электронный ресурс]. – URL: <https://github.com/daniilkrasnopeev/diplom/blob/ba5f79c9ccb6cd3291a363f47a2cddf b83eacbeb/AutoDetailingApp> (дата обращения: 12.06.2026).

Приложение А
(справочное)
Глоссарий

Детейлинг – комплекс работ по профессиональному уходу за автомобилем, направленный на восстановление и сохранение его внешнего вида и технического состояния.

Детейлинг-центр – организация, оказывающая услуги по уходу, очистке, полировке и защите автомобиля.

Клиент – лицо, обращающееся в детейлинг-центр для получения услуг.

Заказ – заявка клиента на выполнение одной или нескольких услуг детейлинга.

Услуга – отдельный вид работ, выполняемых сотрудниками детейлинг-центра.

Сотрудник – работник организации, осуществляющий выполнение услуг или управление деятельностью предприятия.

Администратор – сотрудник, осуществляющий регистрацию клиентов, оформление заказов и контроль выполнения работ.

База данных – организованная совокупность взаимосвязанных данных, используемая для хранения информации о клиентах, заказах и услугах.

Информационная система – программное обеспечение, предназначенное для автоматизации деятельности детейлинг-центра.

Авторизация – процесс проверки учетных данных пользователя для получения доступа к системе.

Учетная запись – совокупность данных, необходимых для идентификации пользователя в системе.

СУБД – система управления базами данных, обеспечивающая создание, хранение и обработку информации.

SQL Server – система управления реляционными базами данных, используемая для хранения данных информационной системы.

Entity Framework Core – технология объектно-реляционного отображения для взаимодействия приложения с базой данных.

WPF – платформа разработки настольных приложений для операционной системы Windows.

Приложение Б
(обязательное)
Исходный код приложения

Листинг Б.1 – Фрагмент кода формы авторизации LoginForm.cs

```

private void btnLogin_Click(object sender, EventArgs e)
{
    string login = txtLogin.Text.Trim();
    string password = txtPassword.Text; // не Trim()

    if (string.IsNullOrEmpty(login) ||
string.IsNullOrEmpty(password))
    {
        MessageBox.Show("Введите логин и пароль.", "Вход",
            MessageBoxButtons.OK, MessageBoxIcon.Warning);
        return;
    }

    const string sql = @"
        SELECT TOP 1
            u.Id,
            u.FullName,
            u.RoleId,
            u.PasswordHash,
            u.PasswordSalt,
            u.PasswordIterations
        FROM dbo.[User] u
        WHERE u.Login = @login AND u.IsActive = 1;";

    try
    {
        using (var con = new SqlConnection(ConnStr))
        using (var cmd = new SqlCommand(sql, con))
        {
            cmd.Parameters.AddWithValue("@login", login);
            con.Open();

            using (var reader = cmd.ExecuteReader())
            {
                if (!reader.Read())
                {
                    MessageBox.Show("Неверный логин или пароль.",
"Вход",
                        MessageBoxButtons.OK,
MessageBoxIcon.Error);
                    return;
                }

                int userId = Convert.ToInt32(reader["Id"]);
                int roleId = Convert.ToInt32(reader["RoleId"]);
                string fio = reader["FullName"].ToString();

                string hashB64 =
reader["PasswordHash"]?.ToString();

```

```

        byte[] salt = reader["PasswordSalt"] ==
DBNull.Value ? null : (byte[])reader["PasswordSalt"];
        int iterations = reader["PasswordIterations"] ==
DBNull.Value ? 0 : Convert.ToInt32(reader["PasswordIterations"]);

        if (string.IsNullOrEmpty(hashB64) || salt ==
null || salt.Length == 0 || iterations <= 0)
        {
            MessageBox.Show("Учетная запись не настроена.
Обратитесь к администратору.",
                "Вход", MessageBoxButtons.OK,
MessageBoxIcon.Error);
            return;
        }

        if (!VerifyPasswordPBKDF2(password, hashB64,
salt, iterations))
        {
            MessageBox.Show("Неверный логин или пароль.",
"Вход",
                MessageBoxButtons.OK,
MessageBoxIcon.Error);
            return;
        }

        var main = new MainForm(userId, roleId, fio);
        main.Show();
        this.Hide();
    }
}
catch (Exception ex)
{
    MessageBox.Show("Ошибка подключения к базе данных:\n" +
ex.Message,
        "Ошибка", MessageBoxButtons.OK,
MessageBoxIcon.Error);
}
}

```

Листинг Б.2 – Фрагмент кода главного окна MainForm.cs

```

private void ApplyRoleUi()
{
    if (btnSupplies != null)
    {
        btnSupplies.Visible = (_role == 1 || _role == 4);
        btnSupplies.Text = "Склад";
    }

    if (btnReqToWork != null) btnReqToWork.Visible = false;
    if (btnReqDone != null) btnReqDone.Visible = false;
}

```

```
}
```

Листинг Б.3 – Фрагмент кода формы создания клиентской заявки

RequestCreateForm.cs

```
using (var con = new SqlConnection(ConnStr))
{
    con.Open();

    using (var tx = con.BeginTransaction())
    {
        try
        {
            int statusNewId = GetStatusIdByName(con, tx, "Новая");

            int clientId = GetOrCreateOrUpdateClient(con, tx, fio,
passport, phone, email);
            int vehicleId = GetOrCreateOrUpdateVehicle(con, tx,
clientId, brand, model, year, vin, reg);

            int requestId = InsertRequest(con, tx, clientId,
vehicleId, description, priority, statusNewId);
            InsertRequestStatusHistory(con, tx, requestId,
statusNewId);
            SaveRequestPhotos(con, tx, requestId);

            tx.Commit();

            CreatedRequestId = requestId;
        }
        catch
        {
            tx.Rollback();
            throw;
        }
    }
}
```

Листинг Б.4 – Фрагмент кода формы формирования отчета ReportPeriodForm.cs

```
private void btnShow_Click(object sender, EventArgs e)
{
    try
    {
        DateTime from = dtFrom.Value.Date;
        DateTime to = dtTo.Value.Date;

        if (from > to)
        {
            MessageBox.Show("Дата 'с' не может быть позже даты
'по'.");
            return;
        }
    }
}
```

```
    }

    LoadReport(from, to);
    LoadTotals(from, to);
}
catch (Exception ex)
{
    MessageBox.Show("Ошибка формирования отчёта:\n" +
ex.Message,
    "Ошибка", MessageBoxButtons.OK, MessageBoxIcon.Error);
}
}
```

Приложение В
(рекомендуемое)
Исходный код модульного тестирования

```

// Тесты для сервиса работы с материалами
public class MaterialServiceMockTests
{
    // Мок-объект репозитория (поддельная БД для тестов)
    private readonly Mock<IMaterialRepository> _mockRepo;

    // Тестируемый сервис
    private readonly MaterialService _service;

    // Конструктор тестового класса
    public MaterialServiceMockTests()
    {
        // Создаем мок репозитория
        _mockRepo = new Mock<IMaterialRepository>();

        // Передаем мок в сервис
        _service = new MaterialService(_mockRepo.Object);
    }

    // Проверка: если имя пустое - должно выбрасываться
    // исключение
    [Fact]
    public void AddMaterial_ShouldThrow_WhenNameEmpty() =>
        Assert.Throws<ArgumentException>(() =>
            _service.AddMaterial("", "kg", "10", "1", "5"));

    // Проверка: если единица измерения пустая - ошибка
    [Fact]
    public void AddMaterial_ShouldThrow_WhenUnitEmpty() =>
        Assert.Throws<ArgumentException>(() =>
            _service.AddMaterial("Sugar", "", "10", "1", "5"));

    // Проверка: цена не может быть отрицательной
    [Fact]
    public void AddMaterial_ShouldThrow_WhenPriceInvalid() =>
        Assert.Throws<ArgumentException>(() =>
            _service.AddMaterial("Sugar", "kg", "-1", "1",
"5"));

    // Проверка: минимальный остаток не может быть отрицательным
    [Fact]
    public void AddMaterial_ShouldThrow_WhenMinStockInvalid() =>
        Assert.Throws<ArgumentException>(() =>
            _service.AddMaterial("Sugar", "kg", "10", "-1",
"5"));

    // Проверка: стартовое количество не может быть
    // отрицательным
    [Fact]
    public void AddMaterial_ShouldThrow_WhenStartQtyInvalid() =>
        Assert.Throws<ArgumentException>(() =>

```

```

        _service.AddMaterial("Sugar", "kg", "10", "1", "-
5"));

    // Проверка корректного добавления материала
    [Fact]
    public void AddMaterial_ShouldCallRepo_WhenValidData()
    {
        // Настраиваем мок:
        // материал не существует
        _mockRepo.Setup(r =>
r.MaterialExists(It.IsAny<string>()))
            .Returns(false);

        // При добавлении материала вернуть ID = 1
        _mockRepo.Setup(r => r.InsertMaterial(
            It.IsAny<string>(),
            It.IsAny<string>(),
            It.IsAny<decimal>(),
            It.IsAny<decimal>()))
            .Returns(1);

        // Вызываем метод сервиса
        var id = _service.AddMaterial("Flour", "kg", "20", "5",
"10");

        // Проверяем ID
        Assert.Equal(1, id);

        // Проверяем, что был вызван метод проверки
        существования
        _mockRepo.Verify(r => r.MaterialExists("Flour"),
Times.Once);

        // Проверяем, что материал добавился в БД
        _mockRepo.Verify(r => r.InsertMaterial("Flour", "kg",
20m, 5m), Times.Once);

        // Проверяем, что был добавлен остаток на склад
        _mockRepo.Verify(r => r.InsertStock(
            1,
            10m,
            20m,
            It.IsAny<string>()),
            Times.Once);
    }

    // Проверка: если стартовое количество = 0,
    // то остаток на склад не создается
    [Fact]
    public void
AddMaterial_ShouldNotInsertStock_WhenStartQtyZero()
    {

```

```

        // Материал не существует
        _mockRepo.Setup(r =>
            r.MaterialExists(It.IsAny<string>()))
            .Returns(false);

        // Вернуть ID = 2
        _mockRepo.Setup(r => r.InsertMaterial(
            It.IsAny<string>(),
            It.IsAny<string>(),
            It.IsAny<decimal>(),
            It.IsAny<decimal>()))
            .Returns(2);

        // Добавляем материал
        var id = _service.AddMaterial("Salt", "kg", "5", "1",
"0");

        // Проверяем ID
        Assert.Equal(2, id);

        // Проверяем, что InsertStock не вызывался
        _mockRepo.Verify(r => r.InsertStock(
            It.IsAny<int>(),
            It.IsAny<decimal>(),
            It.IsAny<decimal>(),
            It.IsAny<string>()),
            Times.Never);
    }

    // Проверка: если материал уже существует - ошибка
    [Fact]
    public void AddMaterial_ShouldThrow_WhenMaterialExists()
    {
        // Настраиваем мок:
        // материал уже есть в БД
        _mockRepo.Setup(r => r.MaterialExists("Sugar"))
            .Returns(true);

        // Проверяем выброс исключения
        Assert.Throws<InvalidOperationException>(() =>
            _service.AddMaterial("Sugar", "kg", "10", "1",
"0"));
    }

    // Проверка обработки строк с пробелами
    [Fact]
    public void AddMaterial_ShouldTrimInputs()
    {
        // Материал не существует
        _mockRepo.Setup(r =>
            r.MaterialExists(It.IsAny<string>()))
            .Returns(false);

```

```

// Вернуть ID = 3
_mockRepo.Setup(r => r.InsertMaterial(
    It.IsAny<string>(),
    It.IsAny<string>(),
    It.IsAny<decimal>(),
    It.IsAny<decimal>()))
    .Returns(3);

// Передаем строки с пробелами
var id = _service.AddMaterial(
    " Honey ",
    " kg ",
    " 50 ",
    " 2 ",
    " 1 ");

// Проверяем результат
Assert.Equal(3, id);

// Проверяем вызов метода добавления
_mockRepo.Verify(r => r.InsertMaterial(
    " Honey ",
    " kg ",
    50m,
    2m),
    Times.Once);
}
}

```