Пермский филиал федерального государственного автономного
образовательного учреждения высшего образования
«Национальный исследовательский университет
«Высшая школа экономики»

Факультет социально-экономических и компьютерных наук

Кривов Андрей Викторович

РАЗРАБОТКА ИНФОРМАЦИОННОЙ СИСТЕМЫ ДЛЯ ПЕРСОНАЛА МАГАЗИНА КОМПЬЮТЕРНЫХ КОМПЛЕКТУЮЩИХ

Курсовая работа

студента образовательной программы «Программная инженерия»

по направлению подготовки *09.03.04 Программная инженерия*

Руководитель

преподаватель кафедры
информационных технологий в
бизнесе

_____

Г. И. Рустамханова

Пермь, 2023 год

Аннотация

Автор: студент НИУ ВШЭ Кривов Андрей Викторович, кафедра информационных технологий в бизнесе.

Тема работы: Разработка информационной системы для персонала магазина компьютерных комплектующих.

Актуальность работы обусловлена растущей потребности компаний в автоматизации бизнес-процессов и оптимизации управления персоналом. Информационная система для персонала магазина компьютерных комплектующих позволит значительно повысить эффективность работы магазина, упростить управление заказами, складами и персоналом, а также улучшить обслуживание клиентов.

В первой главе рассматривается анализ задачи и разработка требований. Вторая глава посвящена проектированию базы данных, отчетов и интерфейса приложения. В третьей главе описана реализация информационной системы.

Работа состоит из 3 глав, включает 59 страниц основного текста, 49 иллюстраций, 8 таблиц и 3 приложения. Библиографический список состоит из 3 источников.

ОГЛАВЛЕНИЕ

Введение 4
Глава 1 Анализ задачи и разработка требований для информационной системы для персонала магазина компьютерных комплектующих 5
1.1 Постановка задачи 5
1.2 Анализ предметной области 5
1.3 Анализ существующих решений 7
1.4 Анализ требований к системе 10
1.5 Выводы по первой главе 14
Глава 2 Проектирование информационной системы для персонала магазина компьютерных комплектующих 15
2.1 Проектирование базы данных 15
2.1.1 Описание атрибутов 15
2.1.2 Установление функциональных зависимостей 18
2.1.3 Нормализация базы данных 19
2.2 Проектирование интерфейса приложения 26
2.3 Проектирование отчетов 28
2.4 Выводы по второй главе 29
Глава 3 Реализация информационной системы для персонала магазина компьютерных комплектующих 30
3.1 Выбор инструментальных средств 30
3.2 Создание моделей 30
3.2.1 Создание модели «Пользователь» 31
3.2.2 Создание модели «Склад» 32
3.2.3 Создание модели «Поставщик» 32
3.2.4 Создание модели «Категория комплектующих» 33
3.2.5 Создание модели «Комплектующее» 33
3.2.6 Создание модели «Тип характеристик» 34
3.2.7 Создание модели «Характеристика» 34
3.2.8 Создание модели «Учет комплектующих на складах» 35
3.2.9 Создание модели «Поставка» 36
3.2.10 Создание модели «Заказ» 37
3.2.11 Создание модели составов заказов 38
3.2.12 Создание модели конфигураций 39
3.3 Подключение приложения к базе данных 40
3.4 Разработка запросов 40
3.5 Реализация интерфейса 41
3.5.1 Окно авторизации 41
3.5.2 Главное меню системы 42
3.6 Выводы по третьей главе 63
Заключение 64
Библиографический список 65
ПРИЛОЖЕНИЕ А. Код программы 66
ПРИЛОЖЕНИЕ Б. Техническое задание 171
ПРИЛОЖЕНИЕ В. Руководство оператора 187

Введение

В настоящее время сфера торговли компьютерными комплектующими активно развивается, и управление таким бизнесом требует использования современных информационных технологий. В связи с этим тема разработки информационной системы для персонала магазина компьютерных комплектующих является актуальной. Система должна упростить и ускорить процессы управления продажами, учета товара, а также повысить качество обслуживания покупателей.

Объектом исследования является информационная система для персонала магазина компьютерных комплектующих, а предметом исследования – процессы управления персоналом, складами, заказами, продажами, учета товаров и обслуживания покупателей.

Цель данной курсовой работы – разработка информационной системы, которая позволит автоматизировать основные процессы магазина компьютерных комплектующих. Для достижения данной цели должны быть выполнены следующие задачи:

1. Провести анализ требований к информационной системе.
2. Спроектировать базу данных.
3. Создать пользовательский интерфейс.
4. Разработать функционал для управления продажами и учета товаров.

Данная проблема имеет достаточно широкую разработанность, и в настоящее время существуют информационные системы, разработанные для автоматизации процессов в магазинах. Однако существующие решения не всегда удовлетворяют требованиям конкретных магазинах и могут быть слишком сложными в использовании.

В данной работе будут использованы современные методы и технологии разработки информационных систем. Будут использованы инструменты разработки программного обеспечения на языке программирования C# с использованием UI фреймворка WPF и ORM библиотеки Entity Framework.

Результатом работы будет информационная система, которая удовлетворит заявленным требованиям, повысит эффективность работы персонала и упростит управление бизнесом.

Глава 1 Анализ задачи и разработка требований для информационной системы для персонала магазина компьютерных комплектующих

Данная глава обосновывает актуальность проекта, представляет обзор существующих решений информационной системы для магазина компьютерных комплектующих, формирует функциональные и нефункциональные требования.

1.1 Постановка задачи

Целью курсовой работы является разработка информационной системы для персонала магазина компьютерных комплектующих с целью упрощения процессов управления продажами и инвентаризацией товаров, оптимизации времени и повышения эффективности работы магазина и персонала. Для достижения этой цели необходимо выполнить следующие задачи:

1. Проанализировать предметную область.
2. Создать модель данных.
3. Спроектировать приложение.
4. Разработать запросы к данным.
5. Спроектировать отчеты.
6. Реализовать информационную систему.

1.2 Анализ предметной области

Магазин компьютерных комплектующих – место, где продаются комплектующие для сборки компьютера: процессоры, материнские платы, видеокарты, оперативная память, жесткие диски, корпуса, блоки питания. Возможная организация работы магазина: онлайн, оффлайн или комбинированный вариант.

В магазине могут работать консультанты, аналитики, менеджеры по закупкам, складские работники (при наличии складов), администраторы. У каждого своя роль. Например, консультанты помогают покупателям выбрать нужные комплектующие, а также могут оказывать услуги по сборке и настройке компьютеров. Аналитики проводят анализ полученной информации и проверяют гипотезы, менеджеры по закупкам осуществляют закупки комплектующих у поставщиков, складские

работники оформляют поставки и проводят учет товаров, администраторы управляют персоналом.

Описание процессов оформления заказа, закупки комплектующих, учет на складе представлено в виде диаграммы активностей в нотации UML на рисунке 1.1

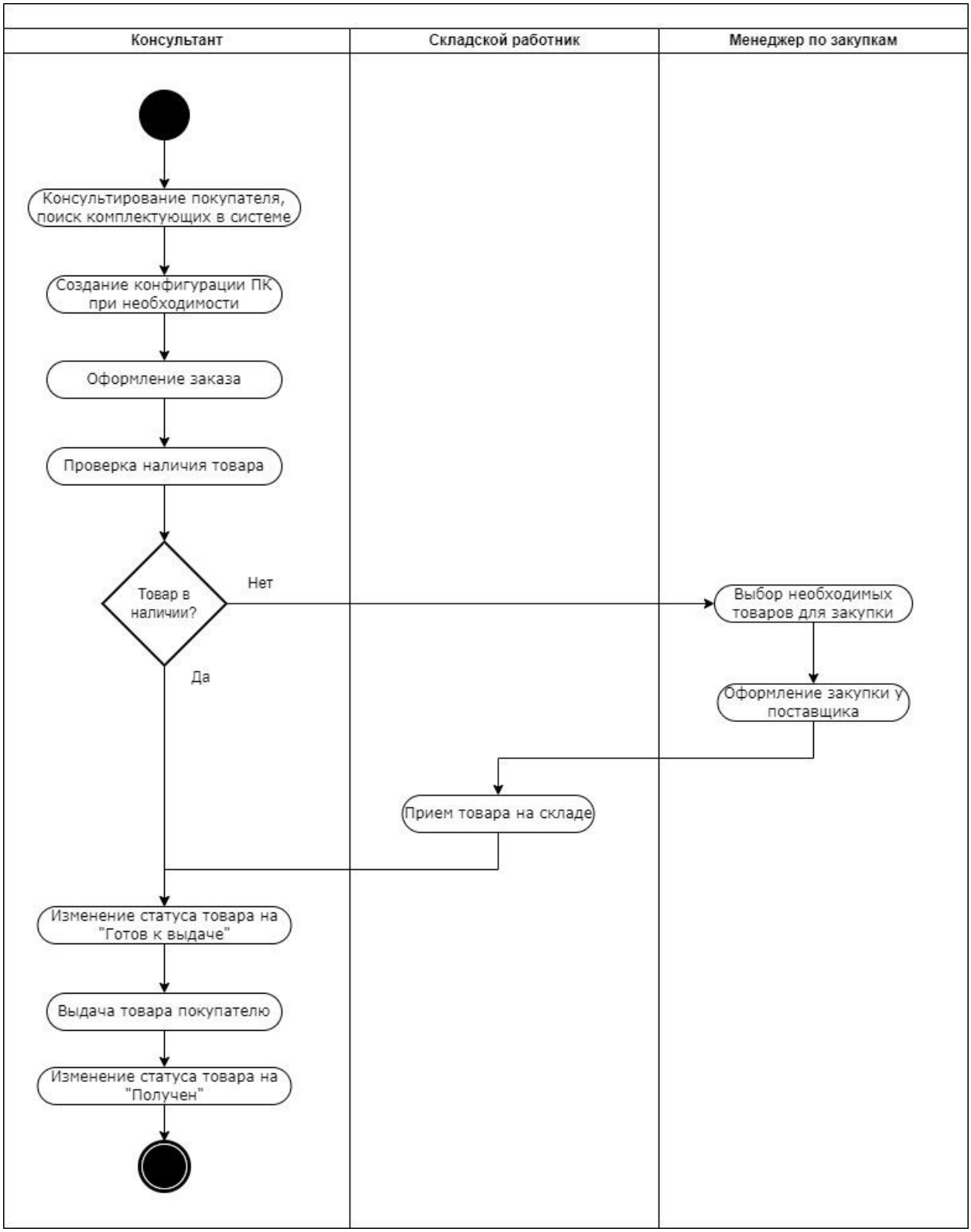


Рисунок 1.1 – Диаграмма активностей

1.3 Анализ существующих решений

На сегодняшний день существует множество приложений для управления магазином и учета товаров. Но данные решения, как правило, являются универсальными. Информация система, которая должна быть разработана в рамках данной курсовой работы, является узконаправленной. Она охватывает магазины, продающие только компьютерные комплектующие.

1С:Предприятие – одна из платформ, наиболее распространенных на российском рынке. Для этой платформы существует решение «1С:Розница 8. Магазин бытовой техники и средств связи» для работы со специфическим ассортиментом (бытовая техника, сотовые телефоны, компьютеры и комплектующие) и бизнес-процессами (включая гарантийное обслуживание и обмен по схеме trade-in).

Основные функциональные возможности данного решения:

1. Заказ товаров поставщику.
2. Возврат товаров поставщику.
3. Возврат товаров от покупателей.
4. Сборка товаров.
5. Поступление товаров от контрагента на склад магазина.
6. Перемещение товаров между магазинами.
7. Перемещение товаров между складами магазина.
8. Пересчет товаров (инвентаризация).
9. Списание.
10. Реализация товаров и услуг.
11. Работа с эквайринговыми системами и с банковскими кредитами.
12. Оформление чеков продажи и сводного отчета по ККМ.
13. Доставка товаров.
14. Перемещение денежных средств между кассами магазина.
15. Заказ товаров от покупателя.
16. Поступление денежных средств от покупателей.
17. Прием товара в ремонт, выдача из ремонта.
18. Передача товара в сторонний сервисный центр, возврат товара из стороннего сервисного центра.

На рисунке 1.2 изображено окно оформления поставки, в котором необходимо указать поставщика, магазин, склад, организацию и другие данные. При добавлении товара заполняется артикул, номенклатура, количество товара в выбранной единице измерения, закупочная цена. НДС учитывается.

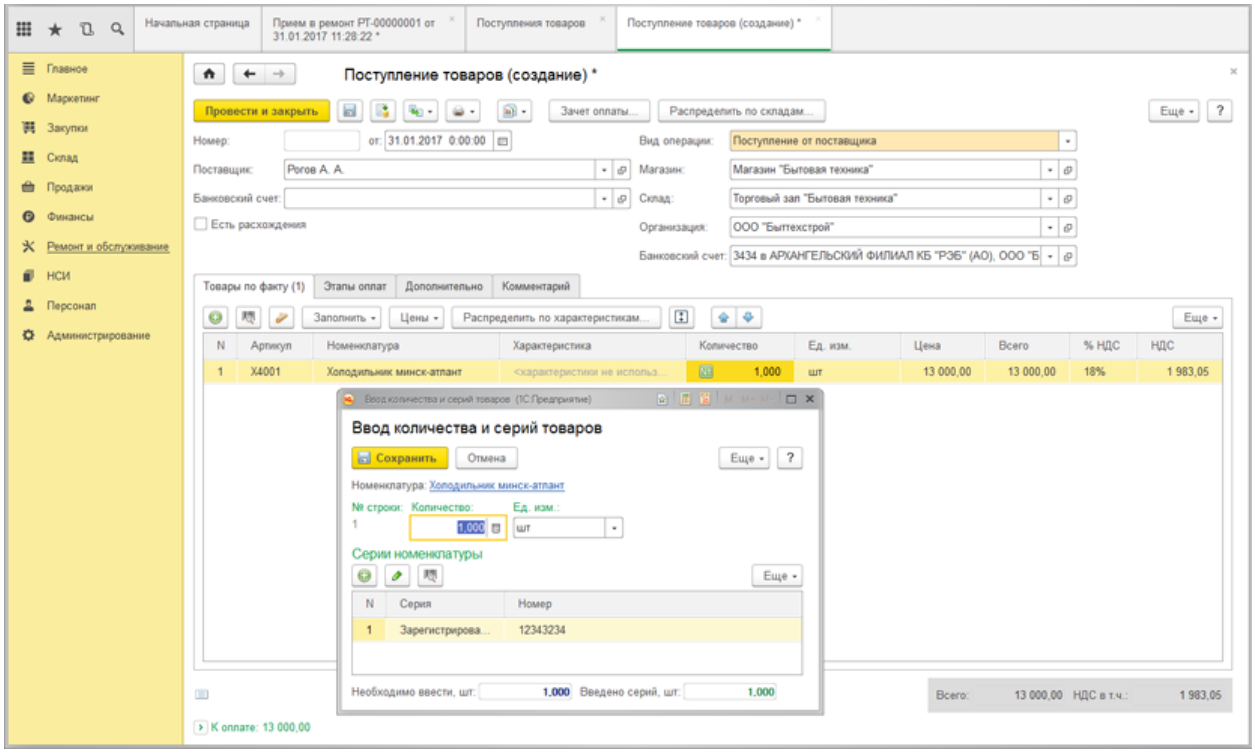


Рисунок 1.2 – Оформление поступления товара в 1С:Предприятие

Приложение позволяет генерировать отчеты по продажам, включая визуализацию в виде столбчатой диаграммы по годам (рисунок 1.3).

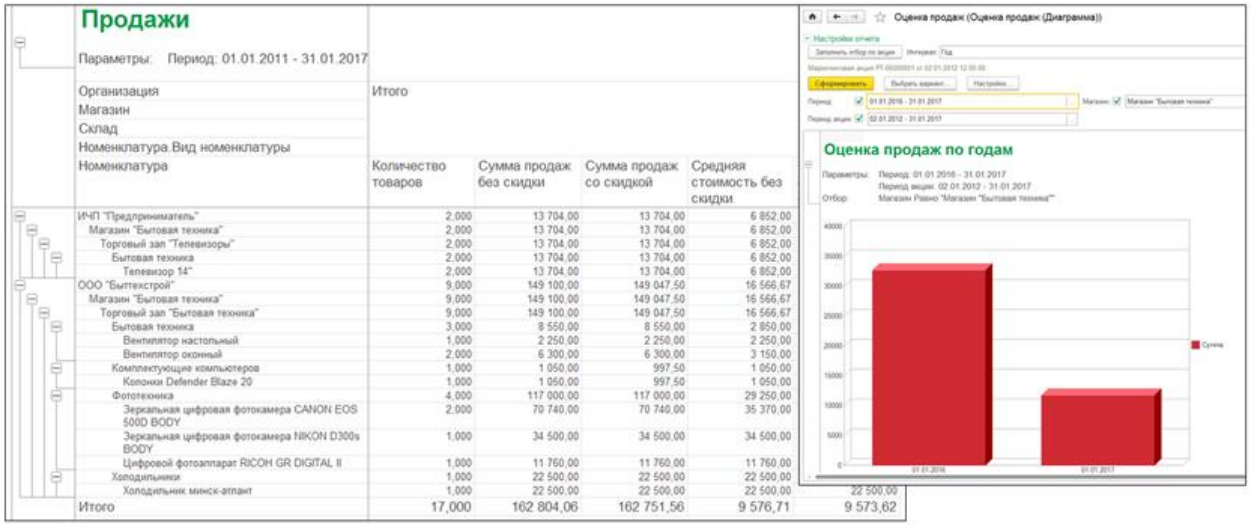


Рисунок 1.3 – Окно продаж в 1С:Предприятие

Система учета товаров позволяет осуществлять прием, передачу в сторонний сервисный центр и выдачу подмены товаров, находящихся в ремонте. Используется как для товаров, принадлежащих клиентам, так и для товаров, принадлежащих

компании. Таким образом, система обеспечивает полный и точный учет товаров в процессе ремонта (рисунок 1.4).

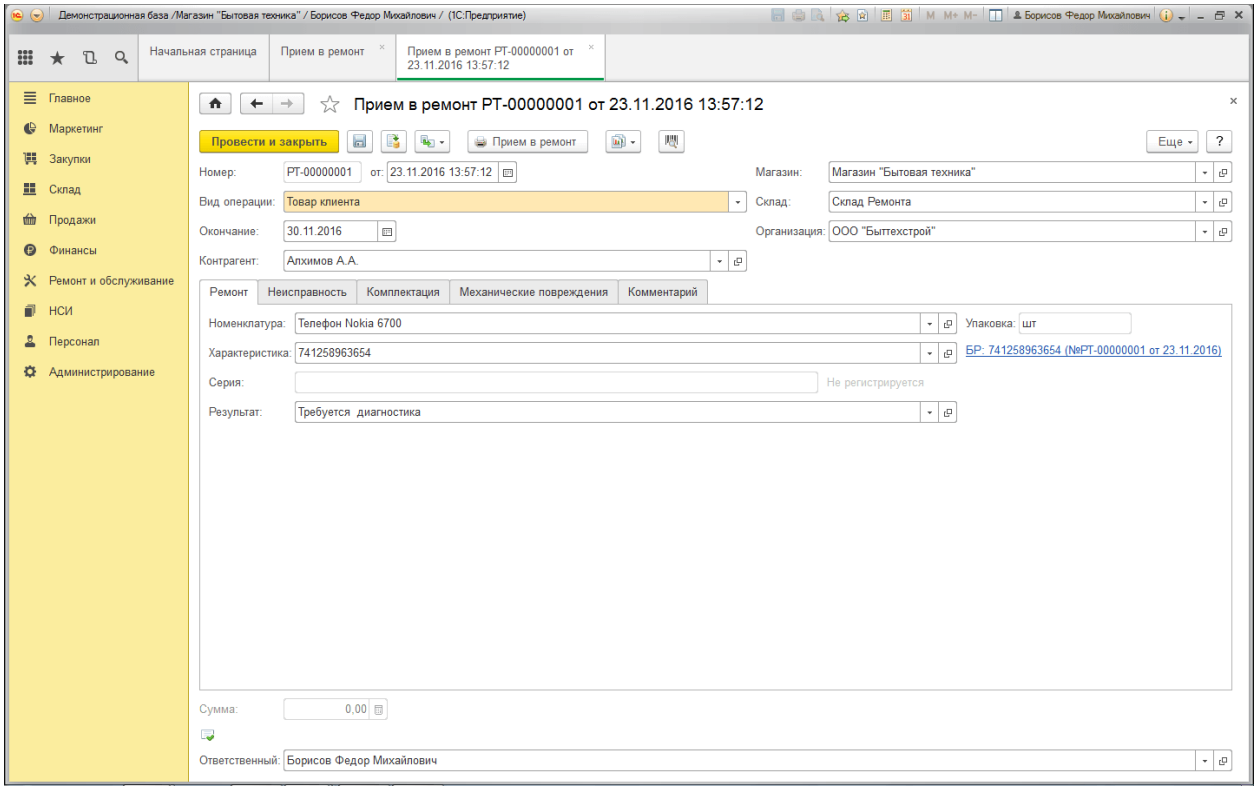


Рисунок 1.4 – Окно приема товара в ремонт

В приложении также доступны две схемы перемещения, реализации и поступления товаров на склады магазина: ордерная и безордерная. Ордерная схема включает перемещение товаров в буферный список, а фактическая операция с товарным остатком производится при помощи расходного или приходного ордера. Для перемещения товаров между магазинами используется безордерная схема, при которой списание товаров с остатком в одном магазине и оприходование их в другом происходит в момент проведения документа «Перемещение товаров». В данном случае дополнительное оформление складских ордеров не требуется. На рисунке 1.5 изображено окно перемещения товаров.

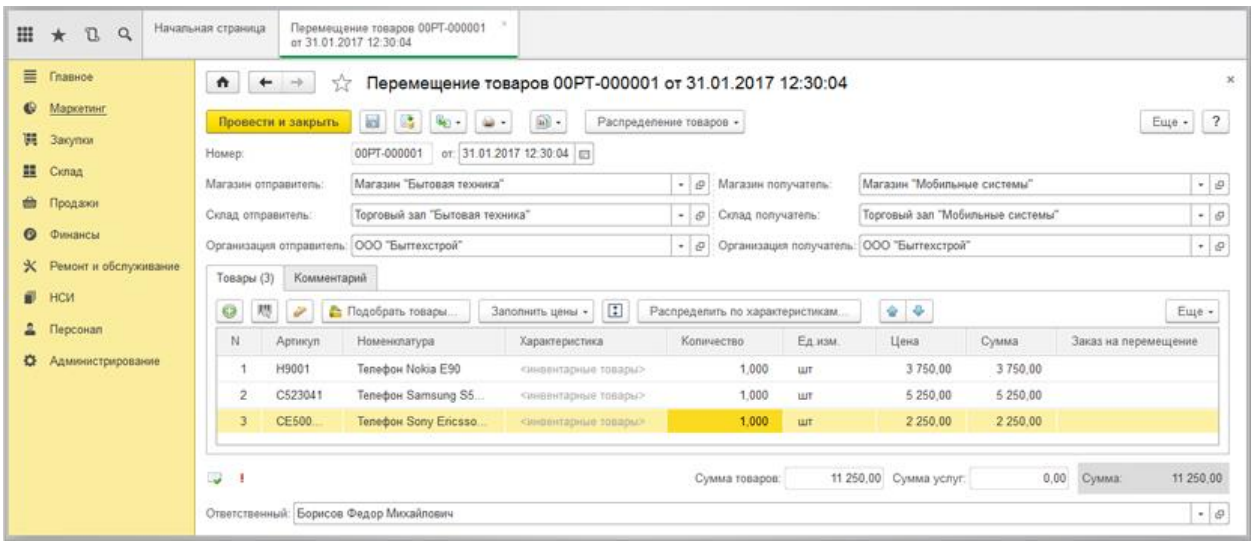


Рисунок 1.5 – Окно перемещения товаров

Описанная система представляет собой универсальное решение, предназначенное для управления широким спектром операций, связанных с продажей компьютерных комплектующих. Она обладает множеством функций, необходимых для полноценной и успешной деятельности магазина, таких как управление складами, продажами, финансами и отчетностью. Благодаря многофункциональности данной системы, магазин может эффективно управлять всеми аспектами своей деятельности.

1.4 Анализ требований к системе

В системе можно выделить два типа требований: функциональные и нефункциональные. Функциональные требования определяют задачи, которые должна выполнять система. Нефункциональные требования, в свою очередь, определяют свойства и ограничения системы, не связанные с её поведением и не являющиеся задачами, которые система должна выполнять.

Следует выделить следующие функциональные требования:

1. Добавление, просмотр, изменение и удаление данных пользователей, складов, поставщиков, комплектующих, категорий комплектующих, типов характеристик, характеристик, учета комплектующих, поставок, конфигураций, заказов и их содержимое.
2. Генерирование отчетов для любой информации в формате xlsx.
3. Визуализация долей категорий комплектующих в виде круговой диаграммы, числа продаж и числа поставок за последние 30 дней в виде столбчатой диаграммы.

4. Система должна быть многопользовательской.

5. Каждый пользователь имеет группу прав, которая определяет доступ к определенным ресурсам и операциям.

Нефункциональные требования:

1. Система должна быть устойчивой при вводе некорректных данных.
2. Пароли должны храниться в захэшированном виде.
3. База данных должна быть нормализована до 3НФ.

С учетом рассмотренных требований к информационной системе составлена диаграмма вариантов использования (рисунок 1.6). Описание прецедентов приведено в таблицах 1.1-1.7.

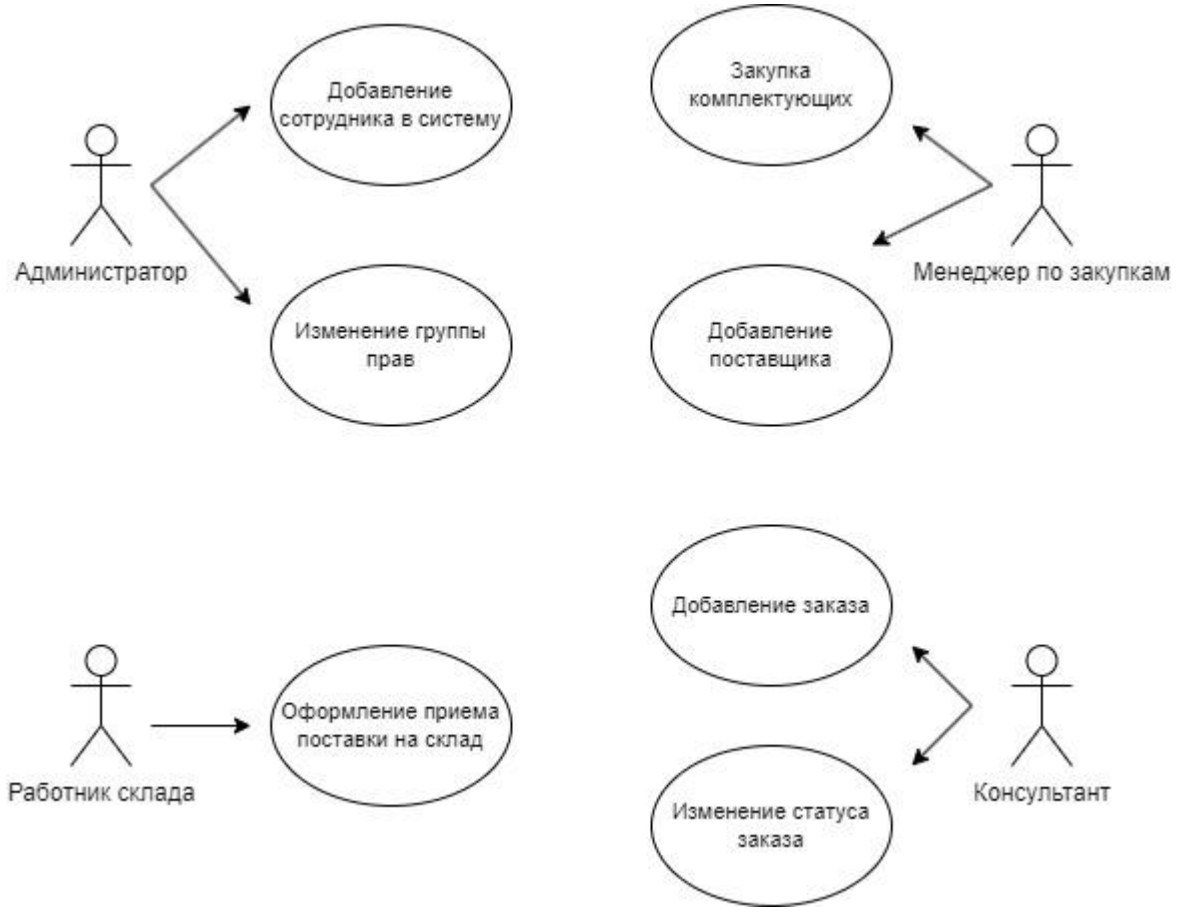


Рисунок 1.6 – Диаграмма вариантов

Таблица 1.1 – Прецедент «Добавление сотрудника в систему»

ID	1
Название	Добавление сотрудника в систему
Акторы	Администратор
Краткое описание	Магазин нанимает нового сотрудника, которому необходимо создать учетную запись в системе
Триггер	Найм нового сотрудника

Основной поток	1. Администратор авторизовывается в систему. 2. Открывает форму добавления нового пользователя, вводит пароль и выбирает группу прав.
Альтернативный поток	Система выдает ошибку о вводе неверных данных

Таблица 1.2 – Прецедент «Изменение группы прав»

ID	2
Название	Изменение группы прав
Акторы	Администратор
Краткое описание	Сотрудник захотел сменить должность, поэтому ему необходимо обновить права доступа в системе
Триггер	Смена должности
Основной поток	1. Администратор авторизовывается в систему. 2. Открывает форму изменения данных о пользователе. 3. Вводит в форму идентификатор сотрудника и изменяет группу прав.
Альтернативный поток	Сотрудник с заданным идентификатором не найден

Таблица 1.3 – Прецедент «Закупка комплектующих»

ID	3
Название	Закупка комплектующих
Акторы	Менеджер по закупкам
Краткое описание	На складе закончилась модель комплектующего, которая пользуется большим спросом или в данный момент её хочет приобрести покупатель
Триггер	Закончилась модель комплектующего
Основной поток	1. Менеджер по закупкам авторизовывается в систему. 2. Делает заказ в системе на закупку определенного количества комплектующего.
Альтернативный поток	

Таблица 1.4 – Прецедент «Добавление поставщика»

ID	4
Название	Добавление поставщика
Акторы	Менеджер по закупкам
Краткое описание	У других поставщиков нет нужного комплектующего
Триггер	Нет поставщика, у которого можно было бы заказать комплектующее

Основной поток	1. Менеджер по закупкам авторизовывается в систему. 2. Добавляет нового поставщика, указывает название организации.
Альтернативный поток	

Таблица 1.5 – Прецедент «Оформление приема поставки на склад»

ID	5
Название	Оформление приема поставки на склад
Акторы	Работник склада
Краткое описание	На склад доставлены комплектующие от поставщика, необходимо обновить учет
Триггер	На склад доставлены комплектующие
Основной поток	1. Работник склада авторизовывается в систему. 2. Обновляет количество комплектующего для данного склада
Альтернативный поток	

Таблица 1.6 – Прецедент «Изменение статуса заказа»

ID	6
Название	Изменение статуса заказа
Акторы	Консультант
Краткое описание	Заказ готов к выдаче, покупатель его забирает
Триггер	Заказ готов к выдаче / Заказ получен
Основной поток	1. Консультант авторизовывается в систему. 2. Обновляет таблицу заказов в приложении.
Альтернативный поток	

Таблица 1.7 – Прецедент «Добавление заказа»

ID	7
Название	Добавление заказа
Акторы	Консультант
Краткое описание	Покупатель собирается приобрести комплектующие
Триггер	Покупатель собирается приобрести комплектующие
Основной поток	1. Консультант авторизовывается в систему. 2. Добавляет новый заказ, указывает комплектующие, стоимость заказа.
Альтернативный поток	Комплектующего нет на складе

1.5 Выводы по первой главе

По итогам выполнения этапа анализа были получены следующие результаты:

1. Выполнена постановка задачи.
2. Проанализирована предметная область.
3. Проанализировано существующее решение «1С:Розница 8. Магазин бытовой техники и средств связи» на платформе 1С:Предприятие.
4. Установлены функциональные и нефункциональные требования, составлена диаграмма вариантов использования с расширенным описанием всех прецедентов.

Глава 2 Проектирование информационной системы для персонала магазина компьютерных комплектующих

В процессе проектирования информационной системы для персонала магазина компьютерных комплектующих необходимо решить следующие задачи:

1. Спроектировать базу данных.
2. Выполнить проектирование интерфейса приложения.
3. Спроектировать отчеты.

2.1 Проектирование базы данных

В данном разделе приведено детальное описание процесса проектирования базы данных для магазина компьютерных комплектующих. Процесс включает следующие этапы:

1. Описание атрибутов, которые будут храниться в базе данных.
2. Установление функциональных зависимостей.
3. Нормализация базы данных до третьей нормальной формы.

2.1.1 Описание атрибутов

Для выполнения описанных на этапе анализа операций по автоматизации учета компьютерных комплектующих необходимо, чтобы в базе данных хранилась информация, описанная в таблице 2.1.

Таблица 2.1 – Атрибуты базы данных

Имя атрибута	**Тип данных**	**Ограничение на значение атрибута**	**Комментарий**
Код пользователя	Целое число	Уникально, обязательно для заполнения	
Пароль	Строка символов	Обязательно для заполнения	Хранится в захэшированном (SHA256) виде
Дата регистрации	Целое число	Обязательно для заполнения	В формате Unix Timestamp, значение по умолчанию – текущая дата
Дата последнего входа	Целое число		В формате Unix Timestamp
Код группы прав	Целое число	Обязательно для заполнения	5 групп прав (от 0 до 4)
Код склада	Целое число	Уникально, обязательно для заполнения	

Имя атрибута	Тип данных	Ограничение на значение атрибута	Комментарий
Город	Строка символов	Обязательно для заполнения, максимальная длина – 64 символа	
Улица	Строка символов	Обязательно для заполнения, максимальная длина – 64 символа	
Дом	Строка символов	Обязательно для заполнения, максимальная длина – 8 символов	
Индекс	Целое число	Обязательно для заполнения	Маска - 000000
Код поставщика	Целое число	Уникально, обязательно для заполнения	
Название организации поставщика	Строка символов	Уникально, обязательно для заполнения, максимальная длина – 128 символов	
Код категории комплектующих	Целое число	Уникально, обязательно для заполнения	
Название категории комплектующих	Строка символов	Уникально, обязательно для заполнения, максимальная длина – 64 символа	
Код комплектующего	Целое число	Уникально, обязательно для заполнения	
Название модели комплектующего	Строка символов	Уникально, обязательно для заполнения, максимальная длина – 256 символов	
Гарантия комплектующего	Целое число	Обязательно для заполнения, не может быть меньше 1	

Имя атрибута	Тип данных	Ограничение на значение атрибута	Комментарий
Код типа характеристики	Целое число	Уникально, обязательно для заполнения	
Название типа характеристики	Строка символов	Уникально, обязательно для заполнения, максимальная длина – 64 символа	
Код характеристики	Целое число	Уникально, обязательно для заполнения	
Значение характеристики	Строка символов	Обязательно для заполнения, максимальная длина – 128 символов	
Код хранения комплектующего на складе	Целое число	Уникально, обязательно для заполнения	
Цена комплектующего	Вещественное число	Обязательно для заполнения, не может быть равно или меньше 0	
Количество комплектующего	Целое число	Обязательно для заполнения, не может быть меньше 1	
Код поставки	Целое число	Уникально, обязательно для заполнения	
Закупочная цена комплектующего	Вещественное число	Обязательно для заполнения, не может быть равно или меньше 0	
Дата поставки	Целое число	Обязательно для заполнения	В формате Unix Timestamp
Код заказа	Целое число	Уникально, обязательно для заполнения	
Дата заказа	Целое число	Обязательно для заполнения	В формате Unix Timestamp
Статус заказа	Целое число	Обязательно для заполнения	5 статусов (от 0 до 4)
Код состава заказа	Целое число	Уникально, обязательно для заполнения	
Часть конфигурации	Логический тип	Обязательно для заполнения	

Имя атрибута	Тип данных	Ограничение на значение атрибута	Комментарий
Код конфигурации	Целое число	Уникально, обязательно для заполнения	
Код части конфигурации	Целое число	Уникально, обязательно для заполнения	

2.1.2 Установление функциональных зависимостей

Рассмотрим следующие функциональные зависимости между атрибутами в каждом отношении:

1. Код пользователя однозначно определяет пароль, дату регистрации, дату последнего входа, группу прав.
2. Код склада однозначно определяет город, улицу, дом, индекс.
3. Код поставщика однозначно определяет название организации поставщика.
4. Код категории комплектующих однозначно определяет название категории комплектующих.
5. Код комплектующего однозначно определяет название модели комплектующего, гарантию комплектующего, код категории комплектующего.
6. Код типа характеристики однозначно определяет название типа характеристики.
7. Код характеристики однозначно определяет код типа характеристики, код комплектующего, значение характеристики.
8. Код хранения комплектующего на складе однозначно определяет цену комплектующего, количество комплектующего, код склада, код комплектующего.
9. Коду поставки однозначно определяет закупочную цену комплектующего, количество комплектующего, дату поставки, код поставщика, код склада, код комплектующего.
10. Код заказа однозначно определяет дату заказа, статус заказа.

11. Код состава заказа однозначно определяет цену комплектующего, количество комплектующего, является ли комплектующее частью конфигурации, код заказа, код склада, код комплектующего.

12. Код части конфигурации однозначно определяет код конфигурации, количество комплектующего, код комплектующего.

2.1.3 Нормализация базы данных

В данном разделе описывается процесс нормализации базы данных с целью приведения её до третьей нормальной формы. Процесс включает в себя выделение отдельных таблиц и установление связей между ними, что позволяет устранить аномалии ввода, редактирования и удаления.

2.1.3.1 Приведение к 1НФ

Отношение находится в первой нормальной форме, когда все его атрибуты являются атомарными, то есть имеют единственное значение, и все кортежи уникальны, так как содержат первичный ключ.

В исходном универсальном отношении содержатся указанные ниже атрибуты:

1. Код пользователя.
2. Пароль.
3. Дата регистрации.
4. Дата последнего входа.
5. Код группы прав.
6. Код склада.
7. Город.
8. Улица.
9. Дом.
10. Индекс.
11. Код поставщика.
12. Название организации поставщика.
13. Код категории комплектующих.
14. Название категории комплектующих.
15. Код комплектующего.
16. Название модели комплектующего.

17. Гарантия комплектующего.
18. Код типа характеристики.
19. Название типа характеристики.
20. Код характеристики.
21. Значение характеристики.
22. Код хранения комплектующего на складе.
23. Цена комплектующего.
24. Количество комплектующего.
25. Код поставки.
26. Закупочная цена комплектующего.
27. Дата поставки.
28. Код заказа.
29. Дата заказа.
30. Статус заказа.
31. Код состава заказа.
32. Часть конфигурации.
33. Код конфигурации.
34. Код части конфигурации.

Согласно функциональным зависимостям, описанным в разделе 2.1.2, формируется первичный ключ для отношения, включающий следующие атрибуты:

1. Код пользователя.
2. Код склада.
3. Код поставщика.
4. Код категории комплектующих.
5. Код комплектующего.
6. Код типа характеристики.
7. Код характеристики.
8. Код хранения комплектующего на складе.
9. Код поставки.
10. Код заказа.
11. Код состава заказа.
12. Код части конфигурации.

Комбинация этих атрибутов уникальна для каждой строки, что позволяет однозначно идентифицировать запись.

Для обеспечения атомарности требуется, чтобы в каждой строке таблицы содержались все значения указанных атрибутов без группировки данных. Однако это может привести к избыточности данных и нежелательным аномалиям при выполнении операций:

1. Аномалии ввода:
   - невозможно ввести данные о заказах, если нет информации о комплектующих;
   - невозможно ввести данные о поставках, если нет информации о складах;
   - невозможно ввести данные о комплектующих, если нет информации о категориях комплектующих.
2. Аномалии редактирования:
   - при изменении информации о комплектующем, потребуется продублировать изменения во всех строках, в которых хранится информация о данном комплектующем;
   - при изменении информации о заказе, потребуется продублировать изменения во всех строках, в которых хранится информация о данном заказе.
3. Аномалии удаления:
   - при удалении информации о поставках одного поставщика все данные о поставщике могут быть утеряны.

Для их устранения необходимо выполнить декомпозицию отношения в соответствии с алгоритмом нормализации.

2.1.3.2 Приведение ко 2НФ

Отношение находится во второй нормальной форме, если оно удовлетворяет критериям первой нормальной формы и не имеет частичной функциональной зависимости не ключевых атрибутов от ключа, то есть не должно быть не ключевых полей, зависящих от части первичного ключа.

В соответствии с описанными выше зависимостями можно сделать вывод, что отношение уже находится во второй нормальной форме.

2.1.3.3 Приведение к 3НФ

Отношение находится в третьей нормальной форме, если оно удовлетворяет критериям второй нормальной формы, отсутствует функциональная зависимость между не ключевыми атрибутами, то есть не ключевые атрибуты взаимно независимы, и каждый не ключевой атрибут не транзитивно зависит от ключа.

Текущее отношение не находится в третьей нормальной форме, так как по атрибуту «Код пользователя» можно однозначно определить следующие атрибуты:

1. Пароль.
2. Дата регистрации.
3. Дата последнего входа.
4. Группа прав.

Атрибут «Код склада» позволяет однозначно определить следующие атрибуты:

1. Город.
2. Улица.
3. Дом.
4. Индекс.

Атрибут «Код поставщика» позволяет однозначно определить атрибут «Название организации поставщика».

Атрибут «Код категории» позволяет однозначно определить атрибут «Название категории комплектующих».

Атрибут «Код комплектующего» позволяет однозначно определить следующие атрибуты:

1. Название модели комплектующего.
2. Гарантия комплектующего.
3. Код категории комплектующего.

Атрибут «Код типа характеристики» позволяет однозначно определить атрибут «Название типа характеристики».

Атрибут «Код характеристики» позволяет однозначно определить следующие атрибуты:

1. Код типа характеристики.
2. Код комплектующего.

3. Значение характеристики.

Атрибут «Код хранения комплектующего на складе» позволяет однозначно определить следующие атрибуты:

1. Цена комплектующего.
2. Количество комплектующего.
3. Код склада.
4. Код комплектующего.

Атрибут «Код поставки» позволяет однозначно определить следующие атрибуты:

1. Цена комплектующего.
2. Количество комплектующего.
3. Дата поставки.
4. Код поставщика.
5. Код склада.
6. Код комплектующего.

Атрибут «Код заказа» позволяет однозначно определить следующие атрибуты:

1. Дата заказа.
2. Статус заказа.

Атрибут «Код состава заказа» позволяет однозначно определить следующие атрибуты:

1. Цена комплектующего.
2. Количество комплектующего.
3. Часть конфигурации.
4. Код заказа.
5. Код склада.
6. Код комплектующего.

Атрибут «Код части конфигурации» позволяет однозначно определить следующие атрибуты:

1. Код конфигурации.
2. Количество комплектующего.
3. Код комплектующего.

Таким образом, в результате приведения универсального отношения к третьей нормальной форме, должны быть выделены следующие таблицы (подчеркиванием выделены первичные ключи, штриховым подчеркиванием – внешние ключи):

1. Таблица «Пользователи»:
   - код пользователя;
   - пароль;
   - дата регистрации;
   - дата последнего входа;
   - группа прав.
2. Таблица «Склады»:
   - код склада;
   - город;
   - улица;
   - дом;
   - индекс.
3. Таблица «Поставщики»:
   - код поставщика;
   - название организации поставщика.
4. Таблица «Категории комплектующих»:
   - код категории комплектующих;
   - название категории комплектующих.
5. Таблица «Комплектующие»:
   - код комплектующего;
   - название модели комплектующего;
   - гарантия комплектующего;
   - код категории комплектующего.
6. Таблица «Типы характеристик»:
   - код типа характеристики;
   - название типа характеристики.
7. Таблица «Характеристики»:
   - код характеристики;
   - код типа характеристики;

- код комплектующего;
- значение характеристики.

8. Таблица «Учет комплектующих на складах»:
   - код хранения комплектующего на складе;
   - цена комплектующего;
   - количество комплектующего;
   - код склада;
   - код комплектующего.
9. Таблица «Поставки»:
   - код поставки;
   - закупочная цена комплектующего;
   - количество комплектующего;
   - дата поставки;
   - код поставщика;
   - код склада;
   - код комплектующего.
10. Таблица «Заказы»:
   - код заказа;
   - дата заказа;
   - статус заказа.
11. Таблица «Состав заказов»:
   - код состава заказа;
   - цена комплектующего;
   - количество комплектующего;
   - часть конфигурации;
   - код заказа;
   - код склада;
   - код комплектующего.
12. Таблица «Конфигурации»:
   - код части конфигурации;
   - код конфигурации;
   - количество комплектующего;

- код комплектующего.

На рисунке 2.1 представлена схема базы данных, полученная в результате её проектирования и приведения к третьей нормальной форме.

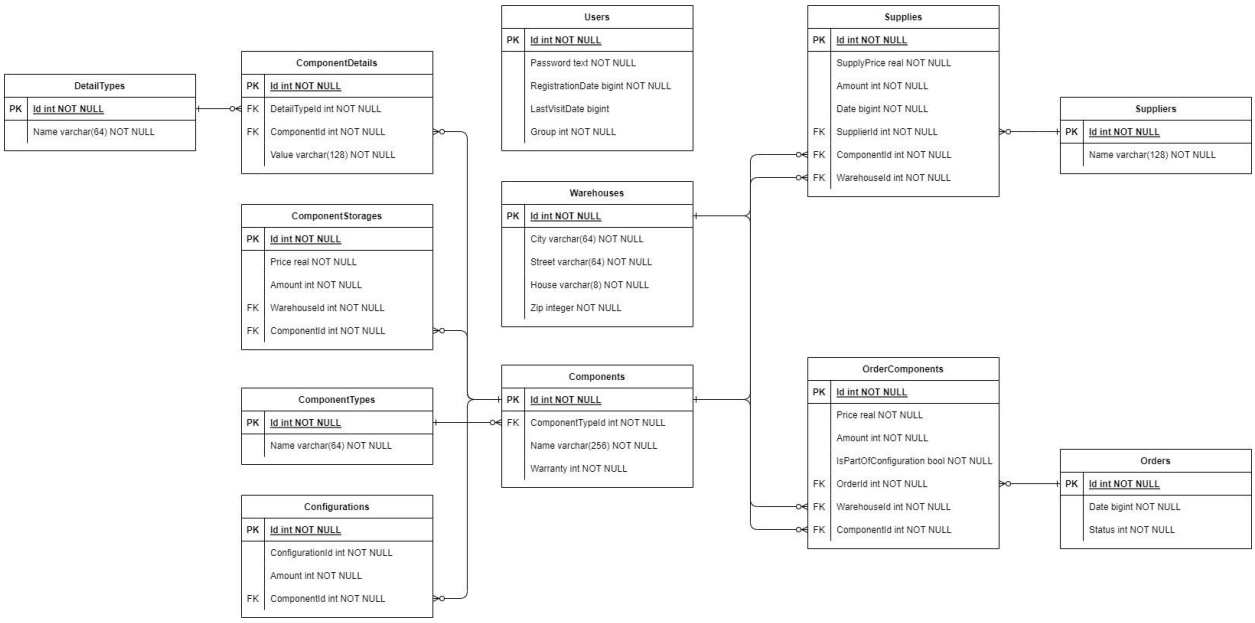


Рисунок 2.1 – Схема базы данных

2.2 Проектирование интерфейса приложения

Информация система разрабатывается для персонала, следовательно, в приложении должно быть окно авторизации, включающее в себя следующие элементы пользовательского интерфейса:

1. Поле для ввода идентификатора сотрудника.
2. Поле для ввода пароля.
3. Кнопка авторизации.

В случае ввода некорректных данных интерфейс должен сообщить об этом пользователю и предоставить возможность ввести их заново. Если данные соответствуют действительности, то должно открыться главное окно, а окно авторизации закрыться. Макет окна авторизации представлен на рисунке 2.2.

Рисунок 2.2 – Макет окна авторизации

В главном окне должны быть расположены вкладки, соответствующие таблицам из базы данных (всего 12 таблиц). Взаимодействие (в том числе выполнение CRUD операций) будет происходить через таблицы. В случае с таблицей «Пользователи» добавление и редактирование записей должно осуществляться через диалоговое окно, так как это необходимо для ввода пароля. На каждой вкладке требуется наличие кнопки создания отчета и кнопки фильтрации, если в таблице присутствует больше 1 столбца (помимо столбца с уникальным идентификатором). Макет главного окна представлен на рисунке 2.3.

Комплектующие Склады Заказы Содержимое заказов Поставщики Поставки Пользователи Конфигурации
Характеристики комплектующих Учет комплектующих Категории комплектующих Типы характеристик комплектующих
Обновить Сохранить Фильтр График Экспорт

ИД	Модель	Категория	Гарантия (мес)
2	Intel Core i5-12400F BOX	Процессоры	12

Рисунок 2.3 – Макет главного окна

2.3 Проектирование отчетов

Информационная система предлагает возможность генерирования 12 отчетов, столько же, сколько таблиц в базе данных. Отчеты повторяют структуру соответствующих таблиц. Формат отчетов – Excel (расширение .xlsx). Ниже приведен пример - структура отчета «Комплектующие»:

1. ИД.
2. Модель.
3. ИД категории.
4. Название категории.
5. Гарантия (мес).

В структуру некоторых отчетах также входят дополнительные столбцы для предоставления более детальной информации. Например, в отчет комплектующих добавлен столбец «Название категории», в самой таблице комплектующих в базе данных его нет.

2.4 Выводы по второй главе

По итогам выполнения этапа проектирования были получены следующие результаты:

1. Выполнено проектирование и нормализация базы данных, в итоге получено 12 таблиц.
2. Спроектирован интерфейс приложения, который состоит из окна авторизации и главного окна, которое включает в себя 12 вкладок для взаимодействия с таблицами из базы данных.
3. Выполнено проектирование отчетов, их общее число равно 12, что соответствует количеству таблиц в базе данных.

Глава 3 Реализация информационной системы для персонала магазина компьютерных комплектующих

В данной главе приведено подробное описание процесса реализации информационной системы для персонала магазина компьютерных комплектующих. Глава включает следующие разделы:

1. Выбор инструментальных средств.
2. Создание моделей.
3. Подключение приложения к базе данных.
4. Разработка запросов.
5. Реализация интерфейса.

3.1 Выбор инструментальных средств

В качестве языка программирования выбран C#, один из самых популярных и наиболее часто используемых в разработке прикладных приложений. Интерфейс построен при помощи UI фреймворка WPF (Windows Presentation Foundation).

База данных хранится в СУБД PostgreSQL, что является отличным решением для малых и средних проектов с низкими минимальными требованиями к производительности и возможностям масштабирования. Помимо этого, PostgreSQL предоставляет большой набор функций и возможностей для работы с данными, что позволяет легко управлять базой данных. Для доступа к таблицам и атрибутам используется ORM фреймворк – Entity Framework Core 7.

3.2 Создание моделей

В данном разделе описывается процесс создания классов на языке программирования C#, которые будут соответствовать таблицам из базы данных. Создание таких классов позволит обращаться к базе данных без написания SQL-кода, что упростит разработку и облегчит поддержку системы.

Каждый класс создается с использованием фреймворка Entity Framework Core 7 и подхода «Data Annotations», что позволяет задавать метаданные для модели напрямую в коде класса. Такой подход облегчает процесс создания модели данных и позволяет легко изменять её в соответствии с требованиями системы.

3.2.1 Создание модели «Пользователь»

Атрибуты таблицы «Пользователи»:

- код пользователя;
- пароль;
- дата регистрации;
- дата последнего входа;
- группа прав.

Для создания модели необходимо написать класс, повторяющий структуру таблицы. Атрибут «Not mapped» означает, что свойство не требуется сохранять в базе данных. На рисунке 3.1 приведен код модели на C#.

```
Ссылок: 9
public class User
{
    Ссылок: 8
    public int Id { get; set; }
    Ссылок: 3
    public required string Password { get; set; }
    Ссылок: 2
    public long RegistrationDate { get; set; }
    Ссылок: 3
    public long? LastVisitDate { get; set; }
    Ссылок: 5
    public Group Group { get; set; }

    [NotMapped]
    Ссылок: 1
    public string RegistrationDateText
    {
        get
        {
            DateTime dateTime = Utilities.UnixTimeStampToDateTime(RegistrationDate);
            return dateTime.ToString("dd-MM-yyyy HH:mm");
        }
    }

    [NotMapped]
    Ссылок: 1
    public string LastVisitDateText
    {
        get
        {
            if (LastVisitDate == null)
                return "";

            DateTime dateTime = Utilities.UnixTimeStampToDateTime((long)LastVisitDate);
            return dateTime.ToString("dd-MM-yyyy HH:mm");
        }
    }
}
```

Рисунок 3.1 – Класс «User»

В класс также добавлены свойства RegistrationDateText и LastVisitDateText, которые используются для вывода даты регистрации и даты последнего входа в текстовом формате.

3.2.2 Создание модели «Склад»

Атрибуты таблицы «Склады»:

- код склада;
- город;
- улица;
- дом;
- индекс.

Аналогично создаем класс «Warehouse» (см. рисунок 3.2).

```
[Index(nameof(City), nameof(Street), nameof(House), IsUnique = true)]
Ссылок: 6
public class Warehouse
{
    Ссылок: 2
    public int Id { get; set; }

    [StringLength(64)]
    Ссылок: 3
    public required string City { get; set; }

    [StringLength(64)]
    Ссылок: 3
    public required string Street { get; set; }

    [StringLength(8)]
    Ссылок: 3
    public required string House { get; set; }

    Ссылок: 2
    public int Zip { get; set; }
    Ссылок: 0
    public virtual ICollection<ComponentStorage> ComponentStorages { get; private set; } = new ObservableCollection<ComponentStorage>();
    Ссылок: 0
    public virtual ICollection<OrderComponent> OrderComponents { get; private set; } = new ObservableCollection<OrderComponent>();
    Ссылок: 0
    public virtual ICollection<Supply> Supplies { get; private set; } = new ObservableCollection<Supply>();
}
```

Рисунок 3.2 – Класс «Warehouse»

Для модели дополнительно указывается индекс, состоящий из трех столбцов: City, Street, House. При этом индекс должен быть уникальным, что исключает возможность дублирования складов с одинаковым адресом. Устанавливается максимальная длина для City (64 символа), Street (64 символа), House (8 символов).

3.2.3 Создание модели «Поставщик»

Атрибуты таблицы «Поставщики»:

- <u>код поставщика</u>;
- название организации поставщика.

Представим таблицу поставщиков в виде класса «Supplier» на C# (см. рисунок 3.3).

```
[Index(nameof(Name), IsUnique = true)]
Ссылок: 3
public class Supplier
{
    Ссылок: 1
    public int Id { get; set; }

    [StringLength(128)]
    Ссылок: 2
    public required string Name { get; set; }
    Ссылок: 0
    public virtual ICollection<Supply> Supplies { get; private set; } = new ObservableCollection<Supply>();
}
```

Рисунок 3.3 – Класс «Supplier»

На свойство «Name» наложено ограничение: максимальная длина равна 128. Кроме того, наименование организации должно быть уникальным.

3.2.4 Создание модели «Категория комплектующих»

Атрибуты таблицы «Категории комплектующих»:

- код категории комплектующих;
- название категории комплектующих.

Представим таблицу категорий комплектующих в виде класса «ComponentType» на C# (см. рисунок 3.4).

```
[Index(nameof(Name), IsUnique = true)]
Ссылок: 5
public class ComponentType
{
    Ссылок: 2
    public int Id { get; set; }

    [StringLength(64)]
    Ссылок: 5
    public required string Name { get; set; }
    Ссылок: 0
    public virtual ICollection<Component> Components { get; private set; } = new ObservableCollection<Component>();
}
```

Рисунок 3.4 – Класс «ComponentType»

На свойство «Name» наложено ограничение: максимальная длина равна 64. Кроме того, название категории должно быть уникальным.

3.2.5 Создание модели «Комплектующее»

Атрибуты таблицы «Комплектующие»:

- код комплектующего;
- название модели комплектующего;
- гарантия комплектующего;
- код категории комплектующего.

Представим таблицу комплектующих в виде класса «Component» на C# (см. рисунок 3.5).

```
Ссылок: 11
public class Component
{
    Ссылок: 2
    public int Id { get; set; }

    Ссылок: 3
    public int ComponentTypeId { get; set; }
    Ссылок: 1
    public virtual ComponentType ComponentType { get; set; } = null!;

    [StringLength(256)]
    Ссылок: 2
    public required string Name { get; set; }
    Ссылок: 5
    public int Warranty { get; set; }

    Ссылок: 0
    public virtual ICollection<ComponentDetail> ComponentDetails { get; private set; } = new ObservableCollection<ComponentDetail>();
    Ссылок: 0
    public virtual ICollection<ComponentStorage> ComponentStorages { get; private set; } = new ObservableCollection<ComponentStorage>();
    Ссылок: 0
    public virtual ICollection<OrderComponent> OrderComponents { get; private set; } = new ObservableCollection<OrderComponent>();
    Ссылок: 0
    public virtual ICollection<Configuration> Configurations { get; private set; } = new ObservableCollection<Configuration>();
    Ссылок: 0
    public virtual ICollection<Supply> Supplies { get; private set; } = new ObservableCollection<Supply>();
}
```

Рисунок 3.5 – Класс «Component»

На свойство «Name» наложено ограничение на максимальное число символов – 256. Кроме того, из данного класса можно обратиться к характеристикам, заказам, конфигурациям, поставкам и информации о хранении заданного комплектующего.

3.2.6 Создание модели «Тип характеристик»

Атрибуты таблицы «Типы характеристик»:

- код типа характеристики;
- название типа характеристики.

Представим таблицу типов характеристик в виде класса «DetailType» на C# (см. рисунок 3.6).

```
[Index(nameof(Name), IsUnique = true)]

public class DetailType
{
    public int Id { get; set; }

    [StringLength(64)]
    public required string Name { get; set; }
    Ссылок: 0
    public virtual ICollection<ComponentDetail> ComponentDetails { get; private set; } = new ObservableCollection<ComponentDetail>();
}
```

Рисунок 3.6 – Класс «DetailType»

На «Name» наложено ограничение – значения данного свойства должны быть уникальными.

3.2.7 Создание модели «Характеристика»

Атрибуты таблицы «Характеристики»:

- код характеристики;
- код типа характеристики;
- код комплектующего;
- значение характеристики.

Представим таблицу характеристик в виде класса «ComponentDetail» на C# (см. рисунок 3.7).

```
[Index(nameof(DetailTypeId), nameof(ComponentId), IsUnique = true)]
Ссылок: 7
public class ComponentDetail
{
    Ссылок: 2
    public int Id { get; set; }

    Ссылок: 3
    public int DetailTypeId { get; set; }
    Ссылок: 1
    public virtual DetailType DetailType { get; set; } = null!;

    Ссылок: 3
    public int ComponentId { get; set; }
    Ссылок: 0
    public virtual Component Component { get; set; } = null!;

    [StringLength(128)]
    Ссылок: 2
    public required string Value { get; set; }
}
```

Рисунок 3.7 – Класс «ComponentDetail»

На значения данного класса наложено два ограничения. Первое ограничение – пара значений «DetailTypeId» и «ComponentId» должна быть уникальной, второе – максимальная длина значения характеристики не должна превышать 128 символов.

3.2.8 Создание модели «Учет комплектующих на складах»

Атрибуты таблицы «Учет комплектующих на складах»:

- код хранения комплектующего на складе;
- цена комплектующего;
- количество комплектующего;
- код склада;
- код комплектующего.

Представим таблицу учета комплектующих на складах в виде класса «ComponentStorage» на C# (см. рисунок 3.8).

```
public class ComponentStorage
{

    public int Id { get; set; }

    public float Price { get; set; }

    public int Amount { get; set; }


    public int WarehouseId { get; set; }

    public virtual Warehouse Warehouse { get; set; } = null!;


    public int ComponentId { get; set; }

    public virtual Component Component { get; set; } = null!;
}
```

Рис 3.8 – Класс «ComponentStorage»

Из данного класса можно обратиться к комплектующему и складу, на котором он хранится.

3.2.9 Создание модели «Поставка»

Атрибуты таблицы «Поставки»:

- код поставки;
- закупочная цена комплектующего;
- количество комплектующего;
- дата поставки;
- код поставщика;
- код склада;
- код комплектующего.

Представим таблицу поставок в виде класса «Supply» на C# (см. рисунок 3.9).

```
Ссылок: 13
public class Supply
{
    Ссылок: 2
    public int Id { get; set; }
    Ссылок: 5
    public float SupplyPrice { get; set; }
    Ссылок: 5
    public int Amount { get; set; }
    Ссылок: 13
    public long Date { get; set; }

    [NotMapped]
    Ссылок: 0
    public DateTime DateHandler
    {
        get
        {
            return Utilities.UnixTimeStampToDateTime(Date);
        }
        set
        {
            Date = Utilities.DateTimeToUnixTimeStamp(value);
        }
    }

    Ссылок: 2
    public int SupplierId { get; set; }
    Ссылок: 0
    public virtual Supplier Supplier { get; set; } = null!;

    Ссылок: 2
    public int ComponentId { get; set; }
    Ссылок: 0
    public virtual Component Component { get; set; } = null!;

    Ссылок: 2
    public int WarehouseId { get; set; }
    Ссылок: 0
    public virtual Warehouse Warehouse { get; set; } = null!;
}
```

Рисунок 3.9 – Класс «Supply»

Таблица «Поставки» связана с таблицами «Поставщики», «Комплектующие», «Склады» отношением многие-к-одному. Поэтому к ним можно обратиться напрямую из класса «Supply» через свойства «Supplier», «Component» и «Warehouse».

3.2.10 Создание модели «Заказ»

Атрибуты таблицы «Заказы»:

- <u>код заказа</u>;
- дата заказа;

- статус заказа.

Представим таблицу заказов в виде класса «Order» на C# (см. рисунок 3.10).

```
Ссылок: 6
public class Order
{
    Ссылок: 2
    public int Id { get; set; }
    Ссылок: 13
    public long Date { get; set; }

    [NotMapped]
    Ссылок: 0
    public DateTime DateHandler
    {
        get
        {
            return Utilities.UnixTimeStampToDateTime(Date);
        }
        set
        {
            Date = Utilities.DateTimeToUnixTimeStamp(value);
        }
    }

    Ссылок: 2
    public OrderStatus Status { get; set; }

    Ссылок: 0
    public virtual ICollection<OrderComponent> OrderComponents { get; private set; } = new ObservableCollection<OrderComponent>();
}
```

Рисунок 3.10 – Класс «Order»

К составу заказа можно обратиться через свойство «OrderComponents», так как таблицы «Order» и «OrderComponents» связаны отношением один-ко-многим.

3.2.11 Создание модели составов заказов

Атрибуты таблицы «Состав заказов»:

- код состава заказа;
- цена комплектующего;
- количество комплектующего;
- часть конфигурации;
- код заказа;
- код склада;
- код комплектующего.

Представим таблицу состава заказов в виде класса «OrderComponent» на C# (см. рисунок 3.11).

```
public class OrderComponent
{

    public int Id { get; set; }

    public float Price { get; set; }

    public int Amount { get; set; }

    public bool IsPartOfConfiguration { get; set; }


    public int OrderId { get; set; }

    public virtual Order Order { get; set; } = null!;


    public int WarehouseId { get; set; }
    Ссылок: 0
    public virtual Warehouse Warehouse { get; set; } = null!;

    Ссылок: 2
    public int ComponentId { get; set; }
    Ссылок: 0
    public virtual Component Component { get; set; } = null!;
}
```

Рисунок 3.11 – Класс «OrderComponent»

Через данный класс можно получить информацию о комплектующем, выбранном складе, заказе через соответствующие свойства «Component», «Warehouse» и «Order».

3.2.12 Создание модели конфигураций

Атрибуты таблицы «Конфигурации»:

- код части конфигурации;
- код конфигурации;
- количество комплектующего;
- код комплектующего.

Представим таблицу конфигураций в виде класса «Configuration» на C# (см. рисунок 3.12).

```
[Index(nameof(ConfigurationId), nameof(ComponentId), IsUnique = true)]
Ссылок: 6
public class Configuration
{
    Ссылок: 2
    public int Id { get; set; }
    Ссылок: 3
    public int ConfigurationId { get; set; }
    Ссылок: 5
    public int Amount { get; set; }

    Ссылок: 3
    public int ComponentId { get; set; }
    Ссылок: 0
    public virtual Component Component { get; set; } = null!;
}
```

Рисунок 3.12 – Класс «Configuration»

В классе «Configuration» пара значений «ConfigurationId» и «ComponentId» должна быть уникальной.

3.3 Подключение приложения к базе данных

Взаимодействие программы с базой данных реализуется через библиотеку Entity Framework. В разделе 3.2 были рассмотрены модели, которые представлены в виде таблиц в СУБД, но для возможности работы с ними необходимо определить класс «StoreContext», который наследуется от «DbContext».

Каждая коллекция имеет тип DbSet<T>, где T – название класса модели (например, User или Component). В конструкторе вызывается метод «EnsureCreated», который используется для создания базы данных, связанных таблиц и других объектов хранения. В методе «OnConfiguring» вызывается функция «UseNpgsql» с указанием данных для подключения к СУБД. Код класса «StoreContext» приведен в листинге А.15 ПРИЛОЖЕНИЯ А.

3.4 Разработка запросов

В разработанной программе используется технология ORM Entity Framework, поэтому в коде программы отсутствует явное применение SQL запросов. Также реализация CRUD-операций (за исключением реализации CRUD для таблицы пользователей), фильтра, группировки данных не требует вызова специальных методов Entity Framework (Where, Add, Remove и других), они реализуются за счет встроенных механизмов WPF и Entity Framework.

Кроме того, Entity Framework применяется при визуализации данных, выполнении CRUD-операций для таблицы пользователей и при экспортировании данных в Excel.

Для получения количества комплектующих каждой категории используется метод «GroupBy», вызов выглядит следующим образом: _context.Components.GroupBy(c => c.ComponentTypeId).Select(c => new { categoryTypeId = c.Key, count = c.Count() }).ToList(). Этот код можно записать на SQL: “SELECT id, COUNT(*) FROM Components GROUP BY ComponentTypeId”.

Чтобы получить список заказов за последние 30 дней по возрастанию даты для построения графика, можно сделать следующий вызов: _context.Orders.Where(o => timeStampLastMoth <= o.Date && o.Date <= Utilities.GetCurrentUnixTimeStamp()).OrderBy(o => o.Date).ToList(). Аналогично для числа продаж, если заменить «Orders» на «Supplies». Функция «GetCurrentUnixTimeStamp» возвращает текущее время в формате unix timestamp.

Выполнение добавления пользователя выглядит следующим образом:

1. Создается объект класса «User», заполняется данными.
2. Вызывается метод Add(user), где user – созданный объект.

В случае изменения данных пользователя – производится поиск с помощью метода Where, после старые данные заменяются новыми. Для удаления необходимо вызвать метод Remove с указанием нужного объекта.

Результаты запросов проиллюстрированы в следующем разделе.

3.5 Реализация интерфейса

В данном разделе рассматривается процесс реализации интерфейса, для которого был разработан макет в ходе проектирования, для настольного приложения на языке программирования C# с использованием библиотеки WPF.

3.5.1 Окно авторизации

При каждом запуске программы пользователю системы необходимо авторизоваться, для чего открывается соответствующее окно (рисунок 3.13).

Окно авторизации включает поля для ввода логина и пароля и кнопку «Войти». Логином является идентификатор сотрудника магазина. В случае ввода некорректных данных пользователь получает сообщение об ошибке авторизации.

Если логин и пароль верны, то окно авторизации закрывается и открывается главное меню.

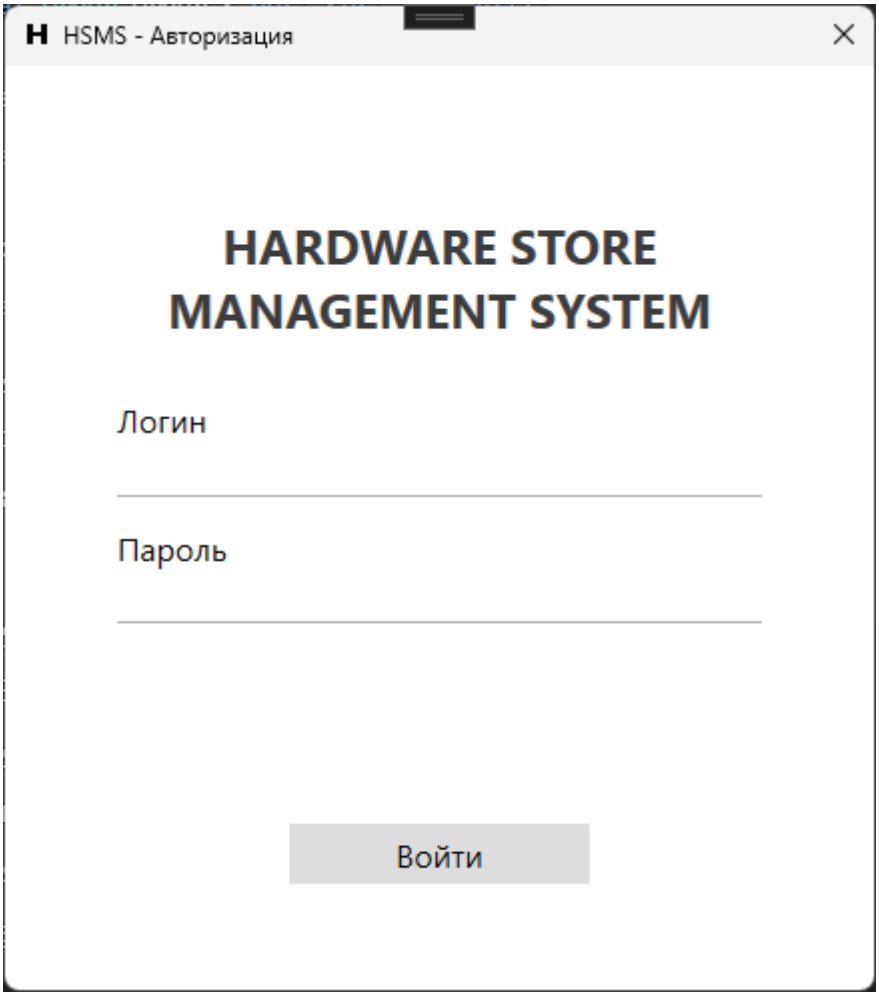


Рисунок 3.13 – Окно авторизации

3.5.2 Главное меню системы

После авторизации пользователю открывается главное меню (рисунок 3.14), в котором происходит взаимодействие с таблицами из базы данных.

Главное меню состоит из трех частей:

1. Вкладки.
2. Панель инструментов.
3. Таблица данных.

Количество вкладок соответствует количеству таблиц в базе данных. Общее их число равно 12. Некоторые вкладки могут быть недоступны в зависимости от группы прав пользователя.

Набор кнопок в панели инструментов зависит от выбранной вкладки. Это будет рассмотрено в данной главе.

Таблица данных выбирается через нажатие на определенную вкладку. Например, при нажатии на вкладку «Комплектующие» выведется таблица комплектующих и соответствующая панель инструментов.

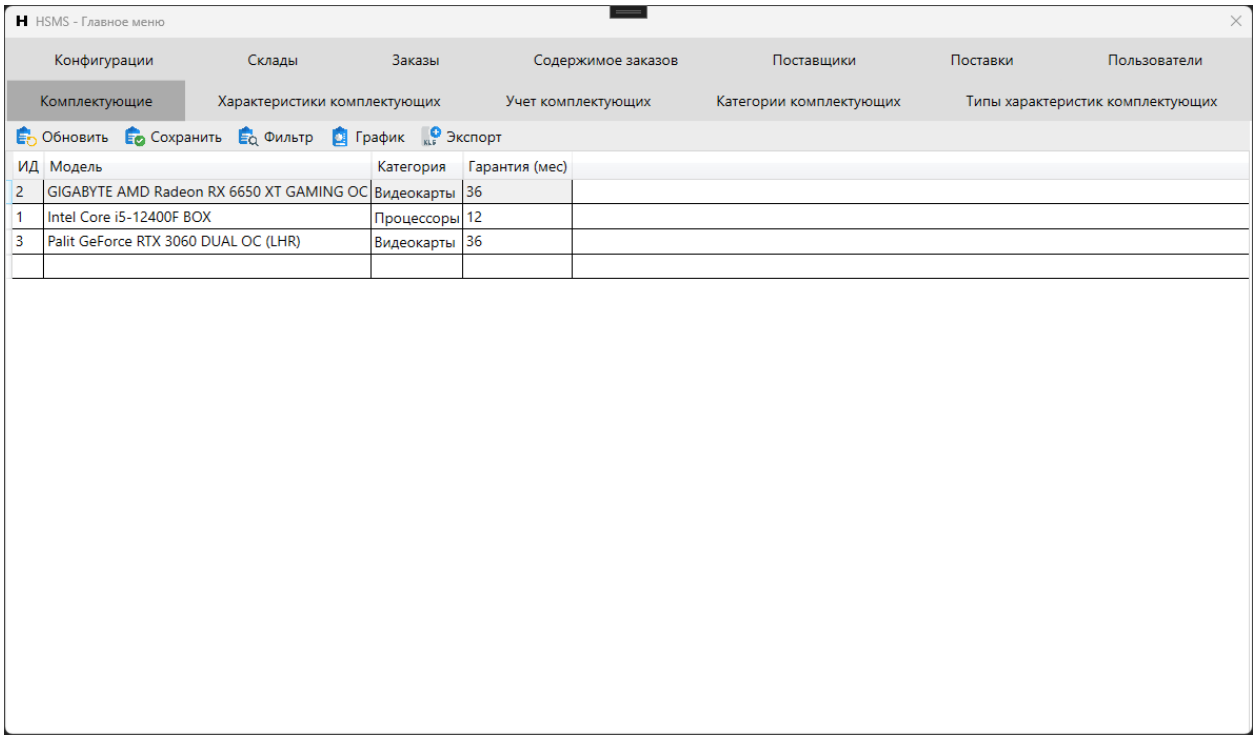


Рисунок 3.14 – Главное меню

Все таблицы в программе реализованы по единому принципу. В фреймворке WPF имеется элемент управления – DataGrid, позволяющий создавать таблицы данных. С помощью DataGrid можно быстро и эффективно организовать выполнение CRUD-операций, и также связать его с базой данных. Для этого с помощью Entity Framework выполняется загрузка определенной таблицы в представление. Далее с помощью свойства DataGrid «ItemsSource» можно указать ссылку на представление. После этого таблица будет связана с базой данных.

3.4.2.1 Вкладка «Пользователи»

На рисунке 3.15 изображена таблица пользователей. Для взаимодействия с ней доступны следующие инструменты:

1. Обновить (загрузить данные из базы данных заново).
2. Сохранить изменения, сделанные в таблице.
3. Создать нового пользователя.
4. Изменить данные существующего пользователя.
5. Удалить полностью информацию о пользователе.
6. Экспортировать данные обо всех пользователях в файл формата .xlsx.

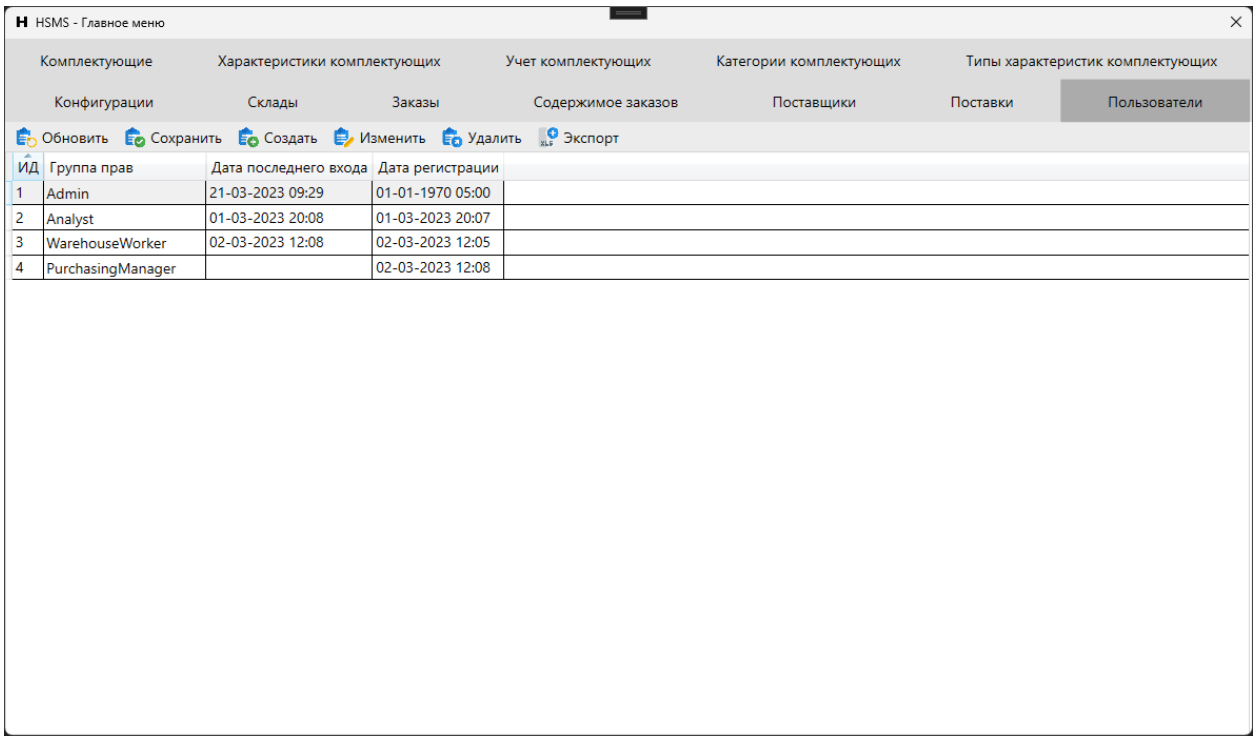


Рисунок 3.15 – Вкладка «Пользователи»

При нажатии на кнопку «Создать» открывается окно «HSMS – Добавить пользователя» (рисунок 3.16). Для добавления пользователя необходимо указать пароль и его группу прав. Его идентификатор будет показан после нажатия на кнопку «ОК».

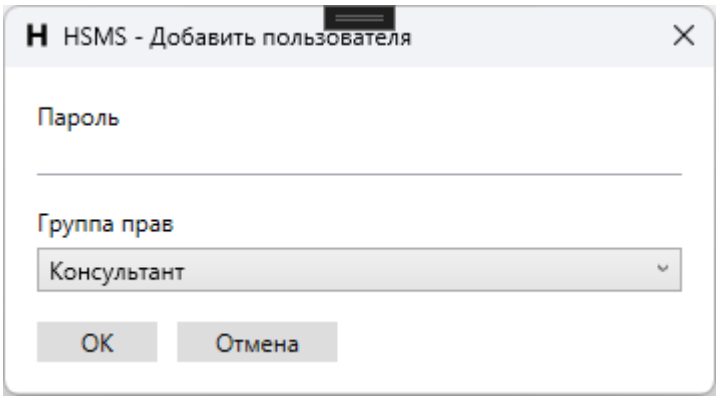


Рисунок 3.16 – Окно добавления пользователя

При нажатии на кнопку «Изменить» открывается окно «HSMS – Изменить данные пользователя» (рисунок 3.17). Для изменения данных необходимо ввести ИД пользователя и заполнить нужные поля. Если пароль не требуется изменять, то поле следует оставить пустым.

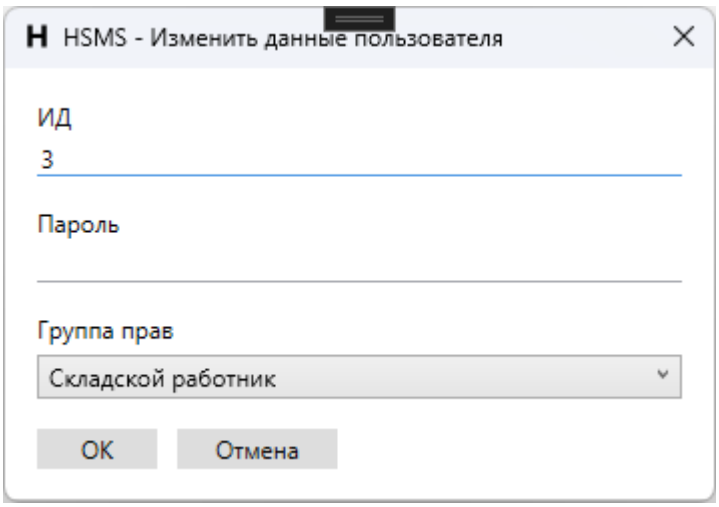


Рисунок 3.17 – Окно изменения данных пользователя

При нажатии на кнопку «Удалить» открывается окно «HSMS – Удалить пользователя», в котором необходимо указать ИД пользователя и нажать на кнопку «ОК» для выполнения удаления.

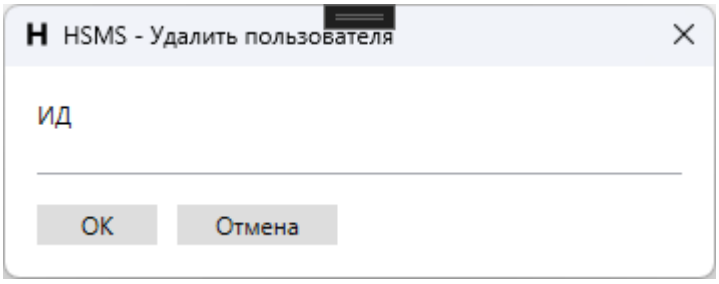


Рисунок 3.18 – Окно удаления пользователя

3.4.2.2 Вкладка «Склады»

На рисунке 3.19 изображена таблица складов. Для взаимодействия с ней доступны следующие инструменты:

1. Обновить (загрузить данные из базы данных заново).
2. Сохранить изменения, сделанные в таблице.
3. Отфильтровать данные по разным параметрам.
4. Экспортировать данные обо всех складах в файл формата .xlsx.

Выполнение CRUD-операций происходит только через таблицу.

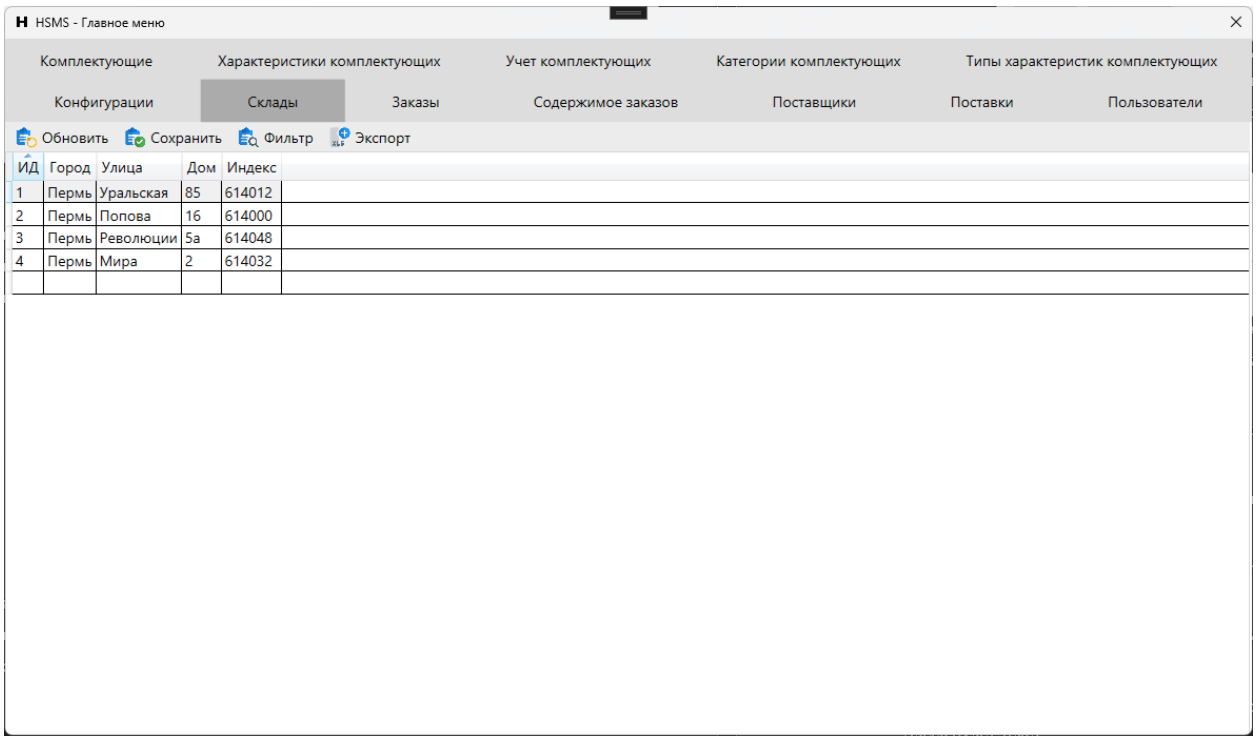


Рисунок 3.19 – Вкладка «Склады»

При нажатии на кнопку «Фильтр» открывается окно «HSMS – Фильтр складов» (рисунок 3.20). Необходимо заполнить те поля, по которым должен осуществляться фильтр данных. Остальные нужно оставить пустыми.

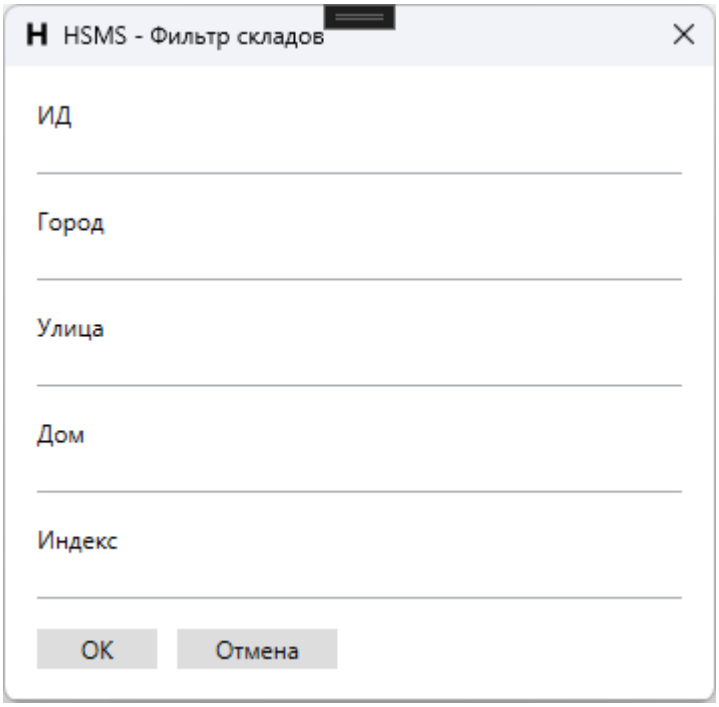


Рисунок 3.20 – Окно фильтра складов

3.4.2.3 Вкладка «Поставщики»

На рисунке 3.21 изображена таблица поставщиков. Для взаимодействия с ней доступны следующие инструменты:

1. Обновить (загрузить данные из базы данных заново).
2. Сохранить изменения, сделанные в таблице.
3. Экспортировать данные обо всех поставщиках в файл формата .xlsx.

Выполнение CRUD-операций происходит только через таблицу.

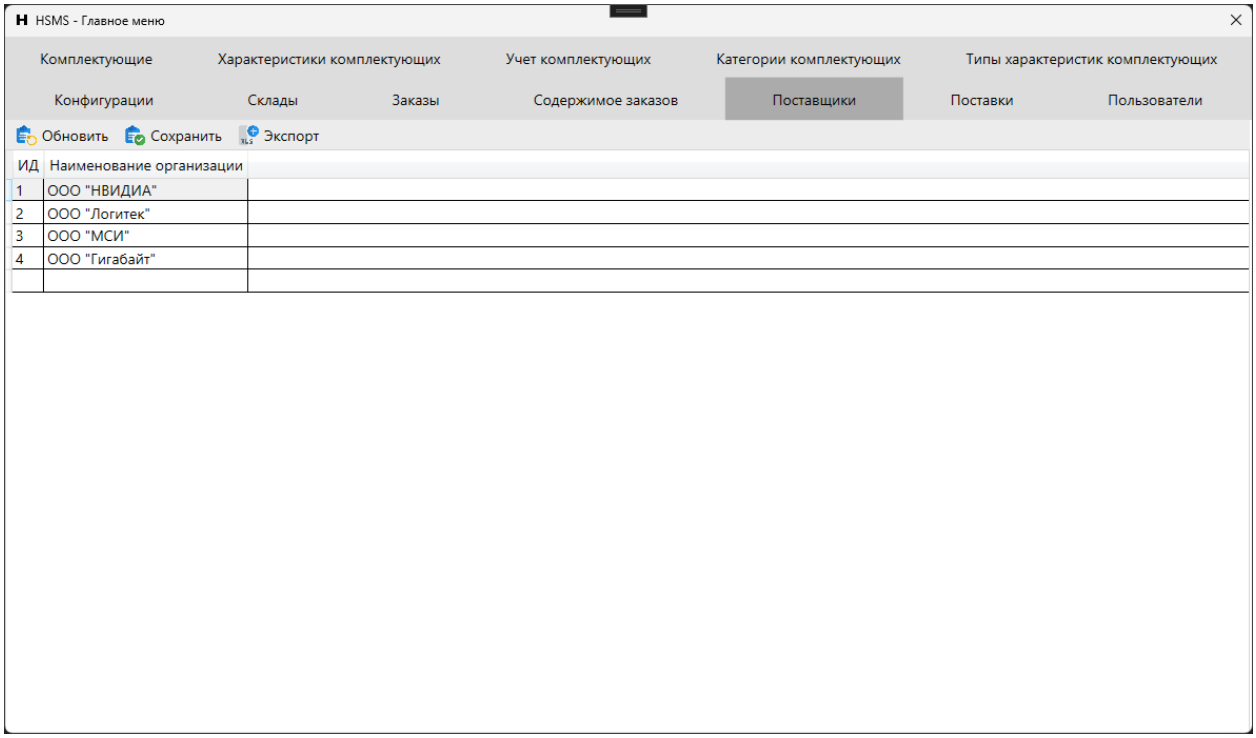


Рисунок 3.21 – Вкладка «Поставщики»

3.4.2.4 Вкладка «Категории комплектующих»

На рисунке 3.22 изображена таблица категорий комплектующих. Для взаимодействия с ней доступны следующие инструменты:

1. Обновить (загрузить данные из базы данных заново).
2. Сохранить изменения, сделанные в таблице.
3. Экспортировать данные обо всех категориях комплектующих в файл формата .xlsx.

Выполнение CRUD-операций происходит только через таблицу.

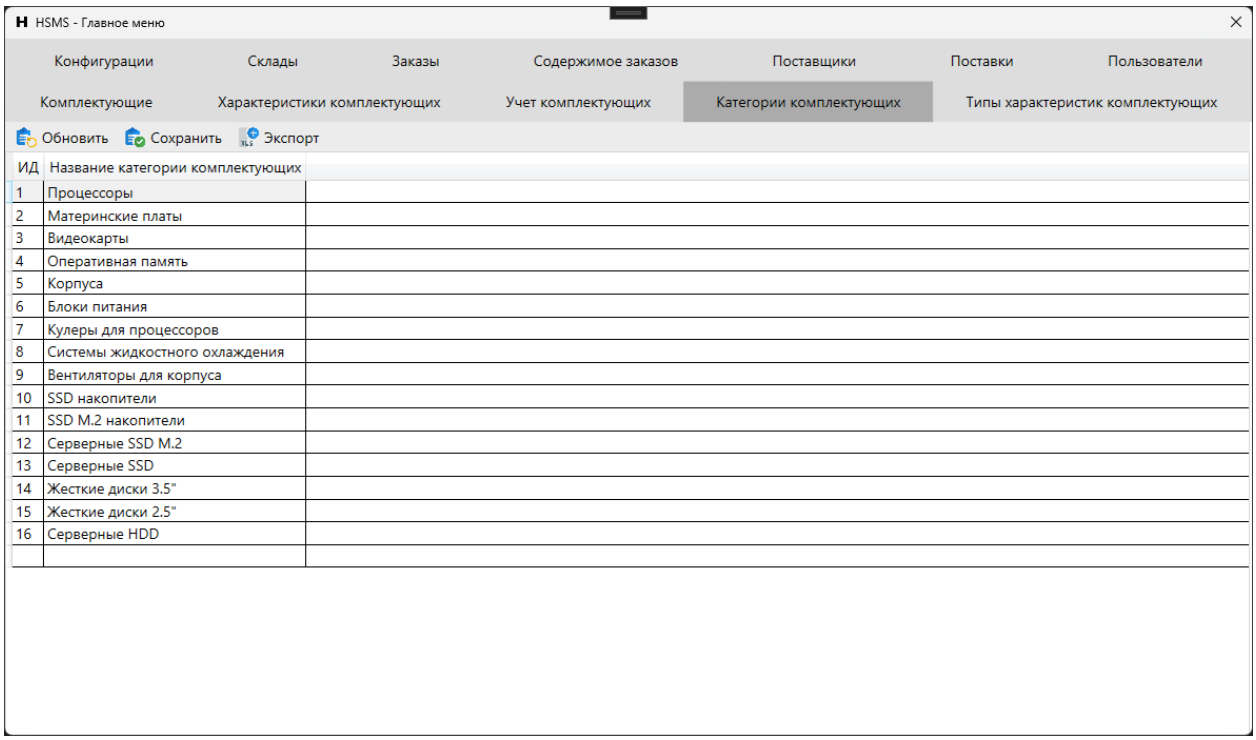


Рисунок 3.22 – Вкладка «Категории комплектующих»

3.4.2.5 Вкладка «Комплектующие»

На рисунке 3.23 изображена таблица комплектующих. Для взаимодействия с ней доступны следующие инструменты:

1. Обновить (загрузить данные из базы данных заново).
2. Сохранить изменения, сделанные в таблице.
3. Отфильтровать данные по разным параметрам.
4. Построить график.
5. Экспортировать данные обо всех комплектующих в файл формата .xlsx.

Выполнение CRUD-операций происходит только через таблицу.

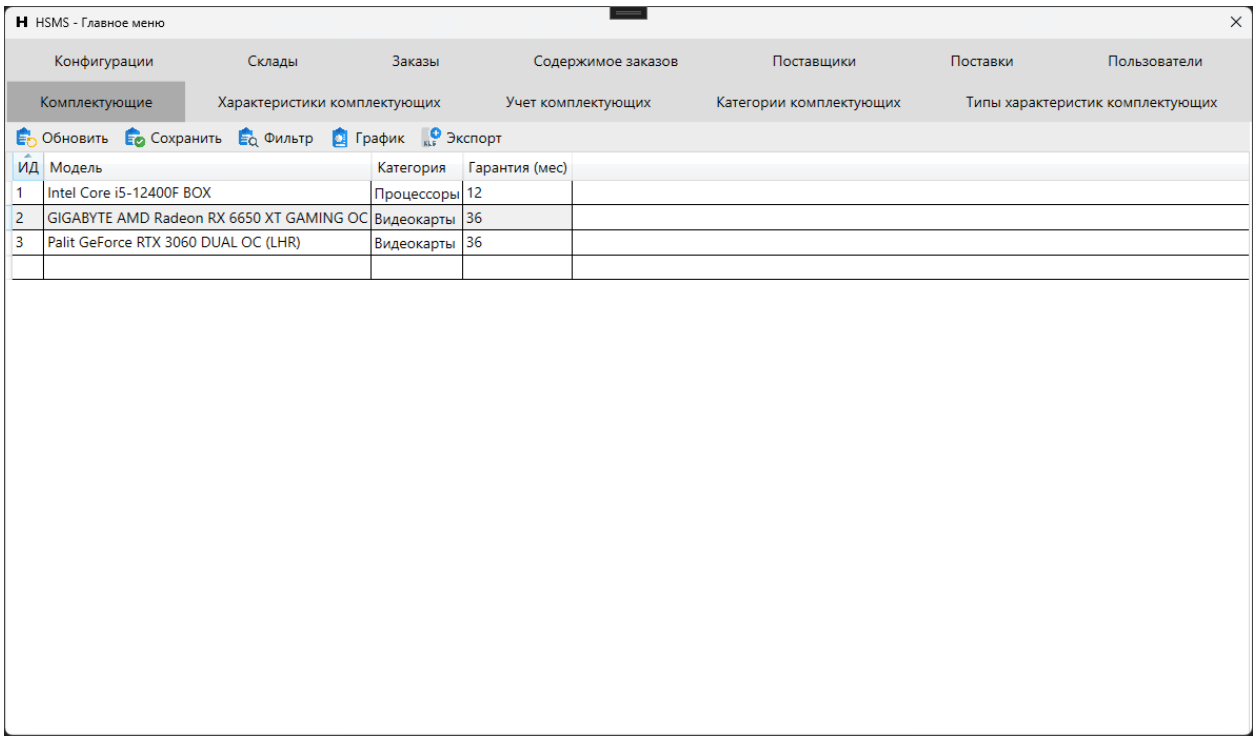


Рисунок 3.23 – Вкладка «Комплектующие»

При нажатии на кнопку «Фильтр» открывается окно «HSMS – Фильтр комплектующих» (рисунок 3.24). Необходимо заполнить те поля, по которым должен осуществляться фильтр данных. Остальные нужно оставить пустыми.

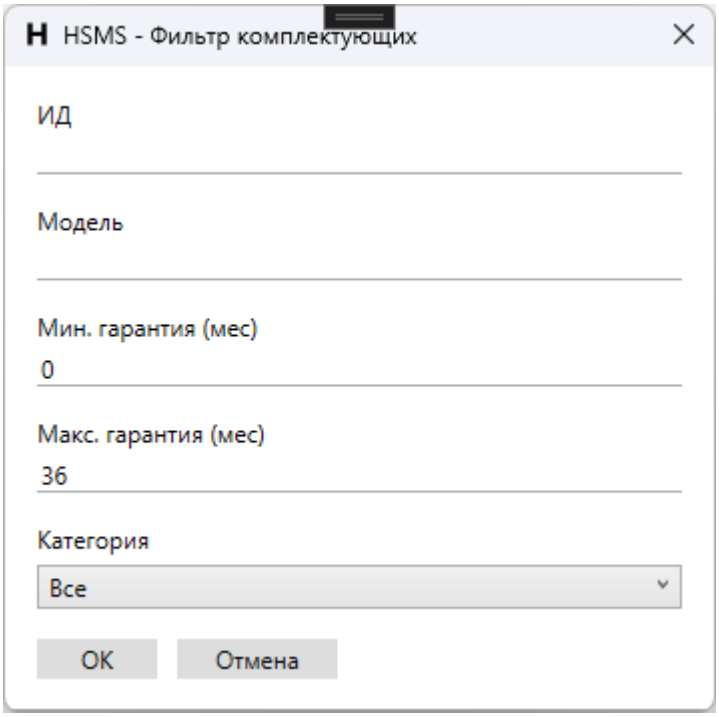


Рисунок 3.24 – Окно фильтра комплектующих

При нажатии на кнопку «График» открывается окно с круговой диаграммой, на которой изображены доли категорий комплектующих (рисунок 3.25).

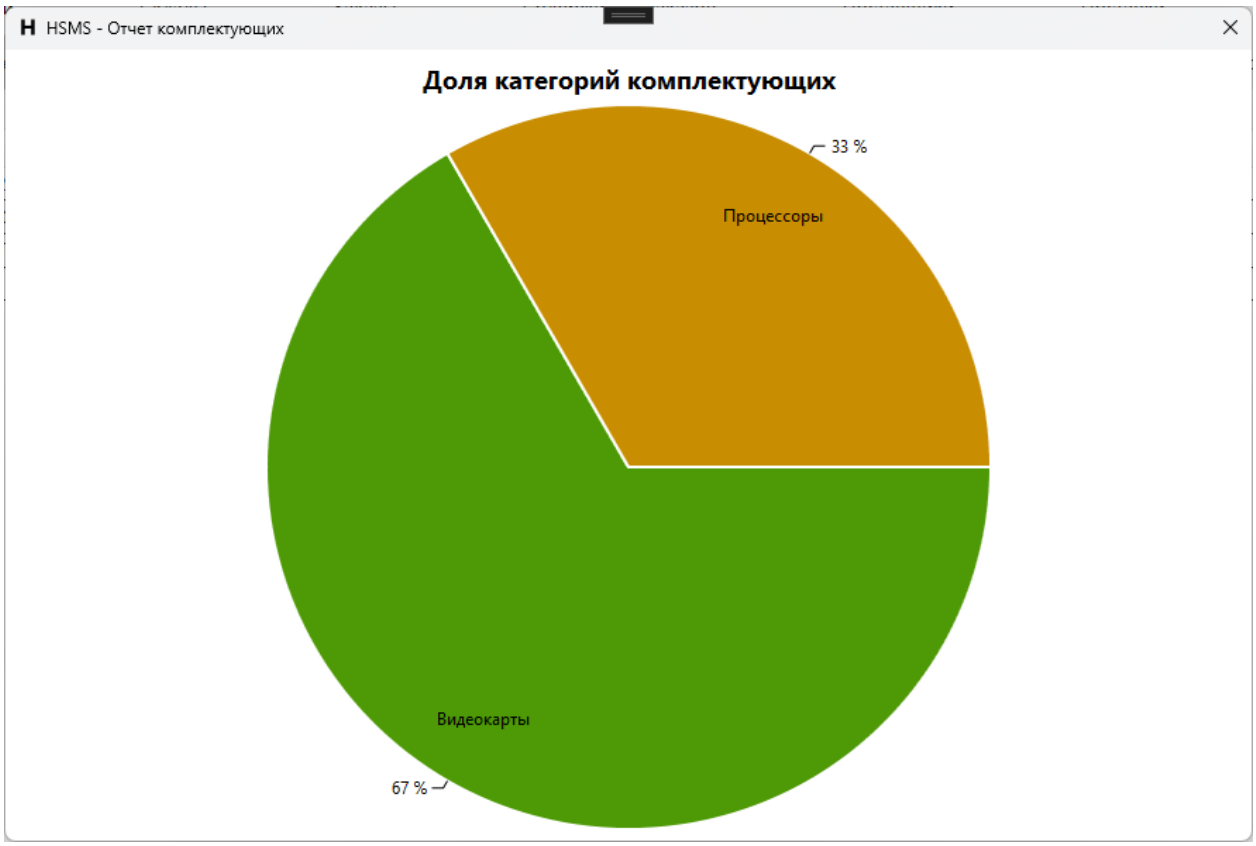


Рисунок 3.25 – График доли категорий комплектующих

3.4.2.6 Вкладка «Типы характеристик комплектующих»

На рисунке 3.26 изображена таблица типов характеристик комплектующих. Для взаимодействия с ней доступны следующие инструменты:

1. Обновить (загрузить данные из базы данных заново).
2. Сохранить изменения, сделанные в таблице.
3. Экспортировать данные обо всех типах характеристик в файл формата .xlsx.

Выполнение CRUD-операций происходит только через таблицу.

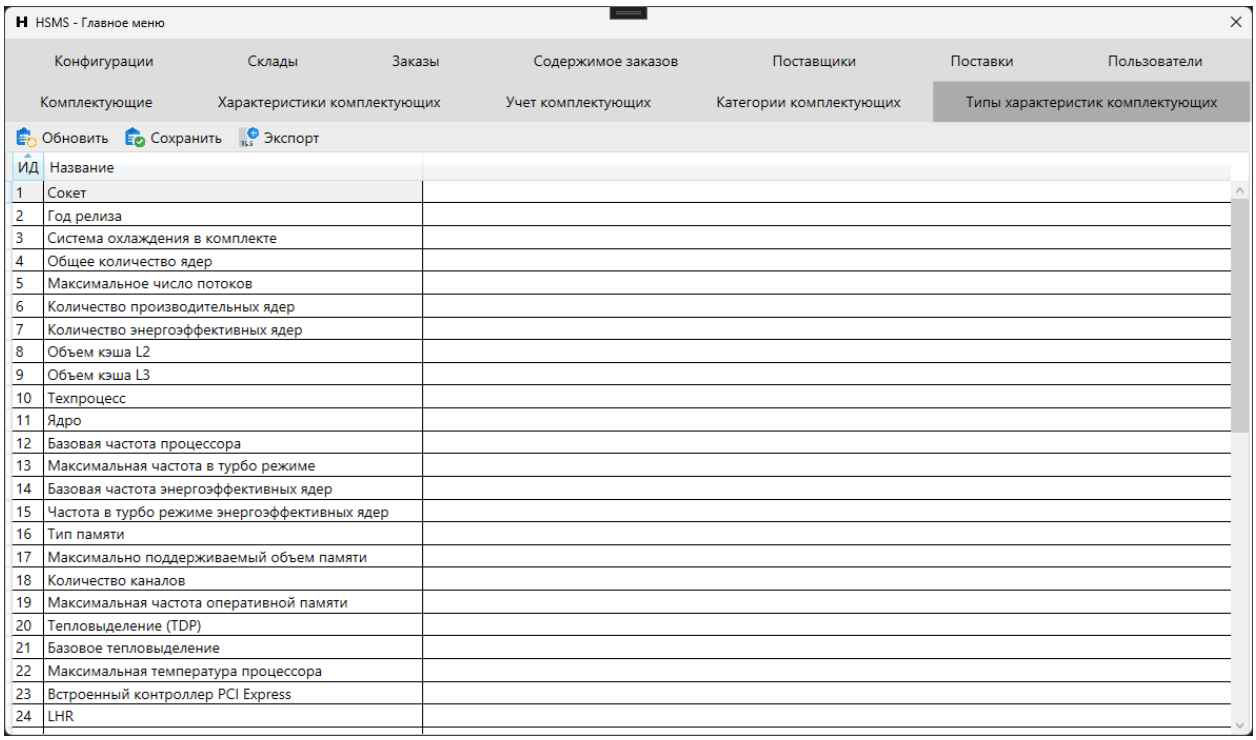


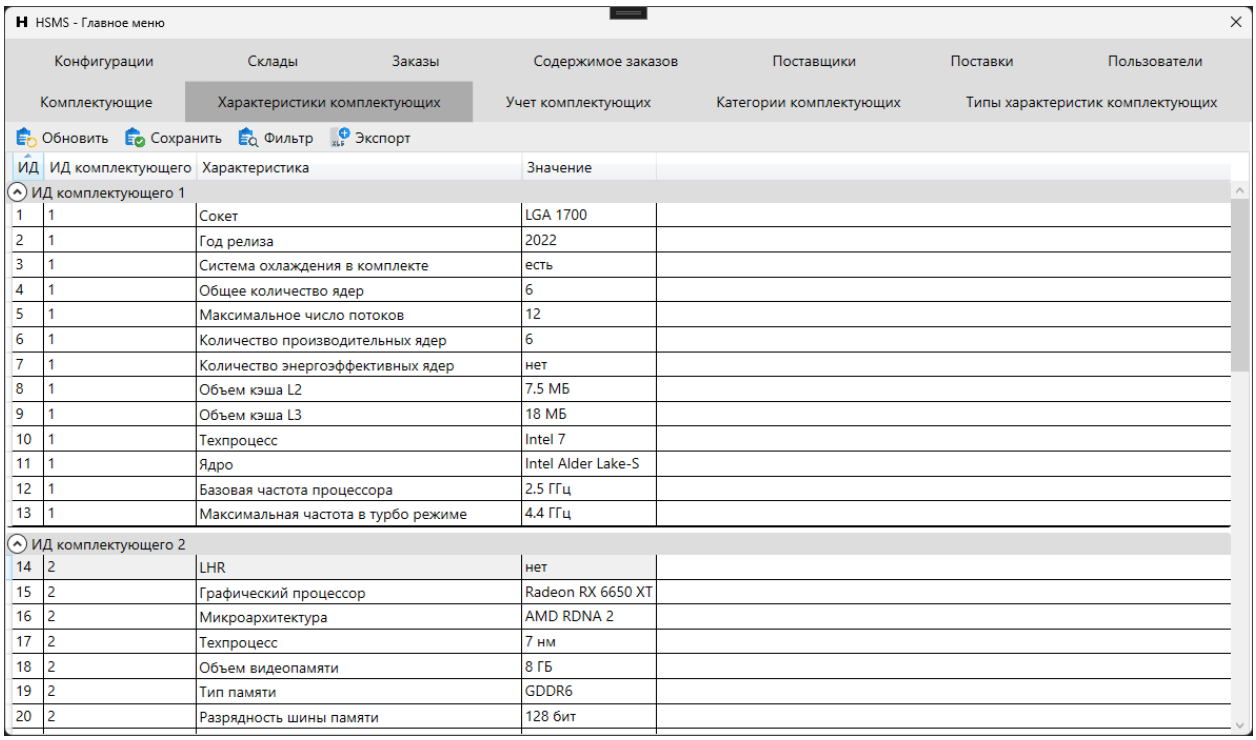
Рисунок 3.26 – Вкладка «Типы характеристик комплектующих»

3.4.2.7 Вкладка «Характеристики комплектующих»

На рисунке 3.27 изображена таблица характеристик комплектующих. Для взаимодействия с ней доступны следующие инструменты:

1. Обновить (загрузить данные из базы данных заново).
2. Сохранить изменения, сделанные в таблице.
3. Отфильтровать данные по разным параметрам.
4. Экспортировать данные обо всех характеристиках комплектующих в файл формата .xlsx.

Выполнение CRUD-операций происходит только через таблицу.



ИД	ИД комплектующего	Характеристика	Значение
ИД комплектующего 1			
1	1	Сокет	LGA 1700
2	1	Год релиза	2022
3	1	Система охлаждения в комплекте	есть
4	1	Общее количество ядер	6
5	1	Максимальное число потоков	12
6	1	Количество производительных ядер	6
7	1	Количество энергоэффективных ядер	нет
8	1	Объем кэша L2	7.5 МБ
9	1	Объем кэша L3	18 МБ
10	1	Техпроцесс	Intel 7
11	1	Ядро	Intel Alder Lake-S
12	1	Базовая частота процессора	2.5 ГГц
13	1	Максимальная частота в турбо режиме	4.4 ГГц
ИД комплектующего 2			
14	2	LHR	нет
15	2	Графический процессор	Radeon RX 6650 XT
16	2	Микроархитектура	AMD RDNA 2
17	2	Техпроцесс	7 нм
18	2	Объем видеопамяти	8 ГБ
19	2	Тип памяти	GDDR6
20	2	Разрядность шины памяти	128 бит

Рисунок 3.27 – Вкладка «Характеристики комплектующих»

При нажатии на кнопку «Фильтр» открывается окно «HSMS – Фильтр характеристик комплектующих» (рисунок 3.28). Необходимо заполнить те поля, по которым должен осуществляться фильтр данных. Остальные нужно оставить пустыми.

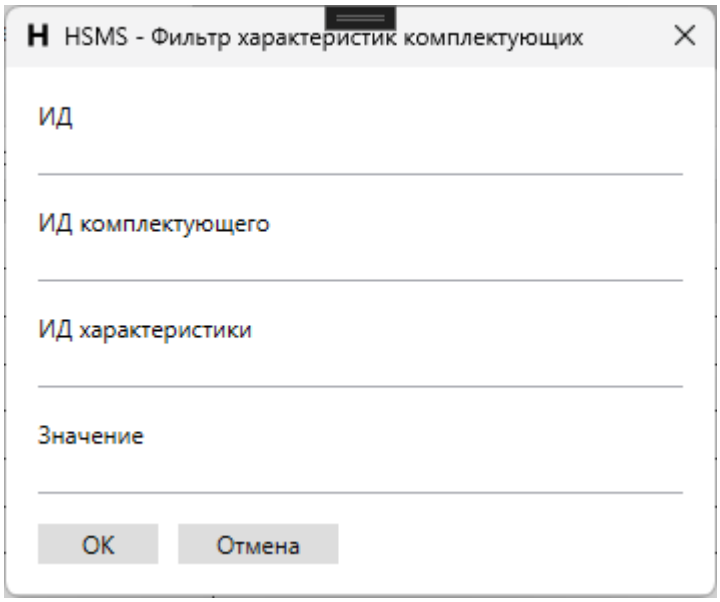


Рисунок 3.28 – Окно фильтра характеристик комплектующих

3.4.2.8 Вкладка «Учет комплектующих»

На рисунке 3.29 изображена таблица учета комплектующих. Для взаимодействия с ней доступны следующие инструменты:

1. Обновить (загрузить данные из базы данных заново).

2. Сохранить изменения, сделанные в таблице.
3. Отфильтровать данные по разным параметрам.
4. Экспортировать данные об учете комплектующих в файл формата .xlsx.

Выполнение CRUD-операций происходит только через таблицу.

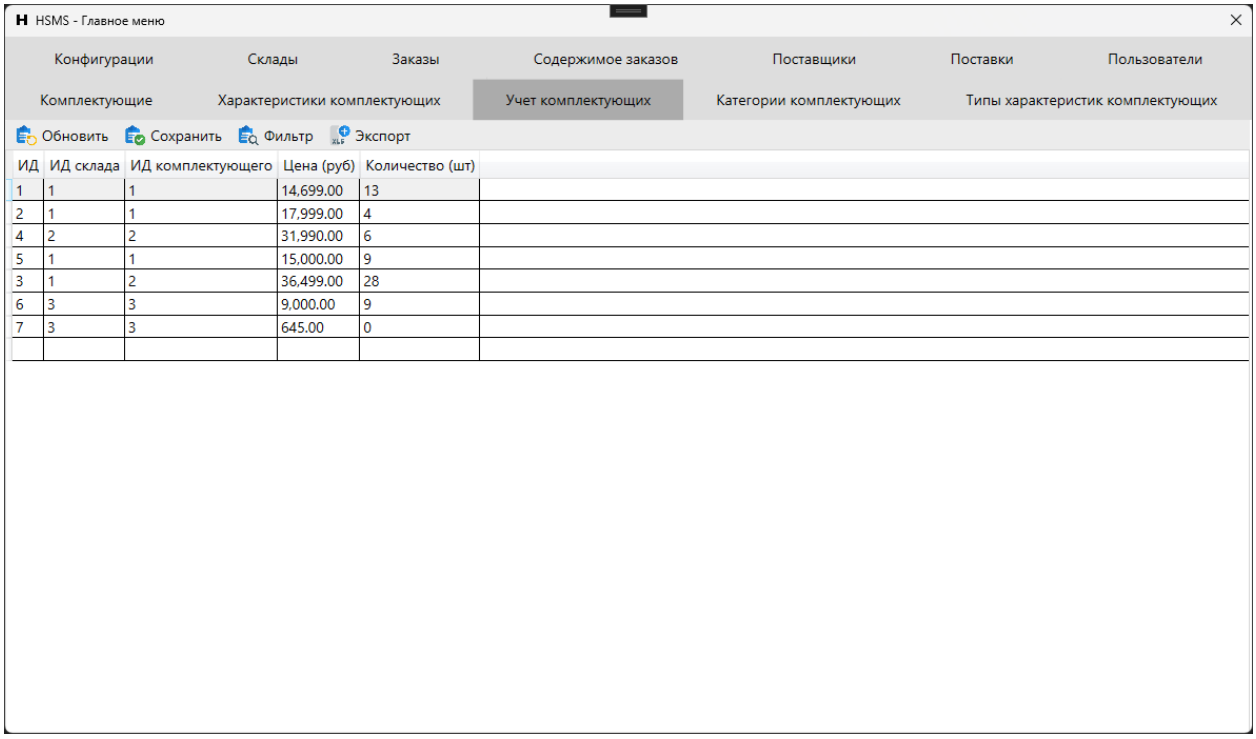


Рисунок 3.29 – Вкладка «Учет комплектующих»

При нажатии на кнопку «Фильтр» открывается окно «HSMS – Фильтр учета комплектующих на складах» (рисунок 3.30). Необходимо заполнить те поля, по которым должен осуществляться фильтр данных. Остальные нужно оставить пустыми.

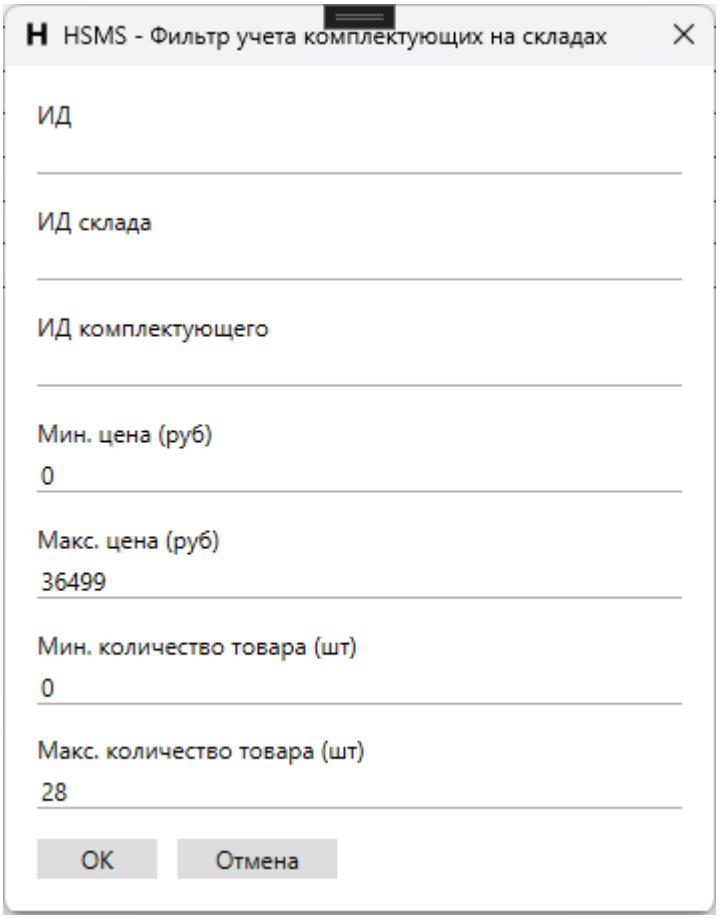


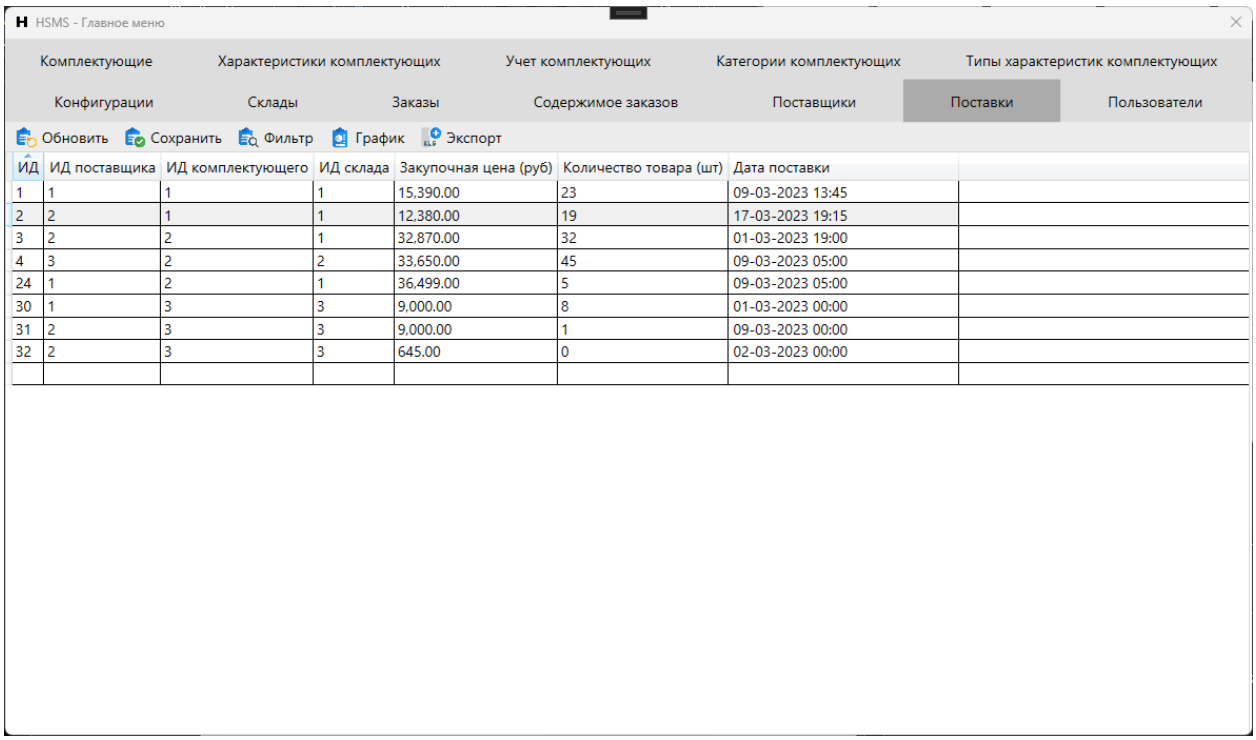
Рисунок 3.30 – Окно фильтра учета комплектующих на складах

3.4.2.9 Вкладка «Поставки»

На рисунке 3.31 изображена таблица поставок. Для взаимодействия с ней доступны следующие инструменты:

1. Обновить (загрузить данные из базы данных заново).
2. Сохранить изменения, сделанные в таблице.
3. Отфильтровать данные по разным параметрам.
4. Построить график.
5. Экспортировать данные обо всех поставках в файл формата .xlsx.

Выполнение CRUD-операций происходит только через таблицу.



ИД	ИД поставщика	ИД комплектующего	ИД склада	Закупочная цена (руб)	Количество товара (шт)	Дата поставки
1	1	1	1	15,390.00	23	09-03-2023 13:45
2	2	1	1	12,380.00	19	17-03-2023 19:15
3	2	2	1	32,870.00	32	01-03-2023 19:00
4	3	2	2	33,650.00	45	09-03-2023 05:00
24	1	2	1	36,499.00	5	09-03-2023 05:00
30	1	3	3	9,000.00	8	01-03-2023 00:00
31	2	3	3	9,000.00	1	09-03-2023 00:00
32	2	3	3	645.00	0	02-03-2023 00:00

Рисунок 3.31 – Вкладка «Поставки»

При нажатии на кнопку «Фильтр» открывается окно «HSMS – Фильтр поставок» (рисунок 3.32). Необходимо заполнить те поля, по которым должен осуществляться фильтр данных. Остальные нужно оставить пустыми.

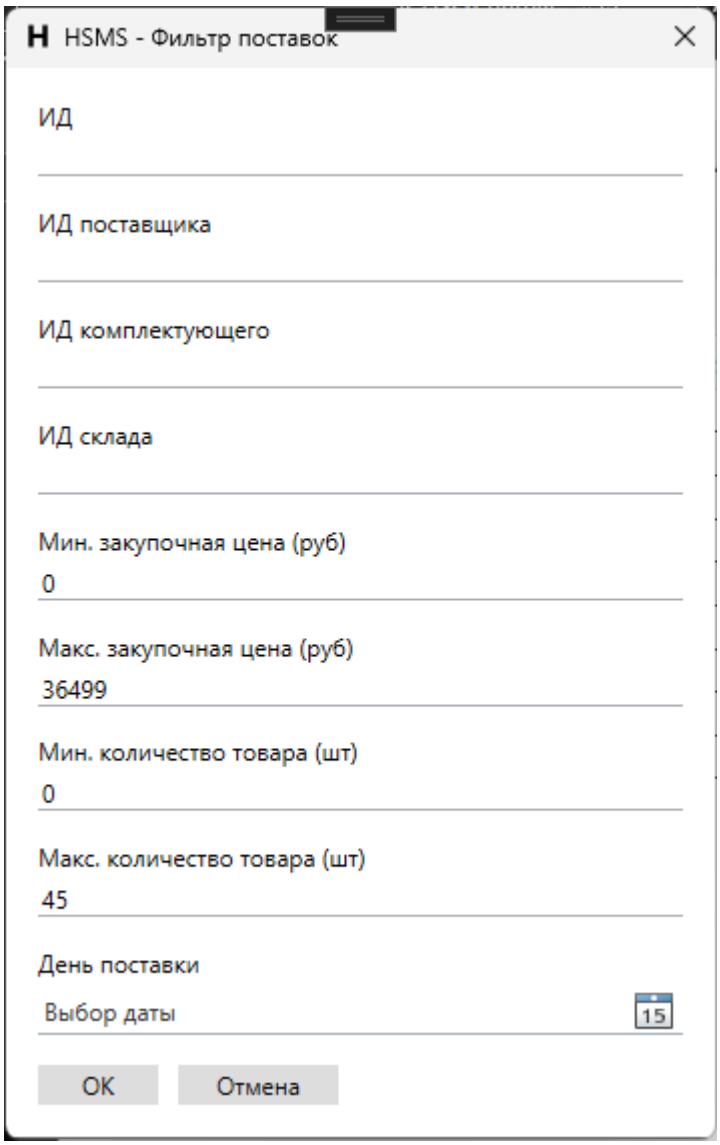


Рисунок 3.32 – Окно фильтра поставок

При нажатии на кнопку «График» открывается окно со столбчатой диаграммой, на которой изображено число поставок за последние 30 дней (рисунок 3.33).

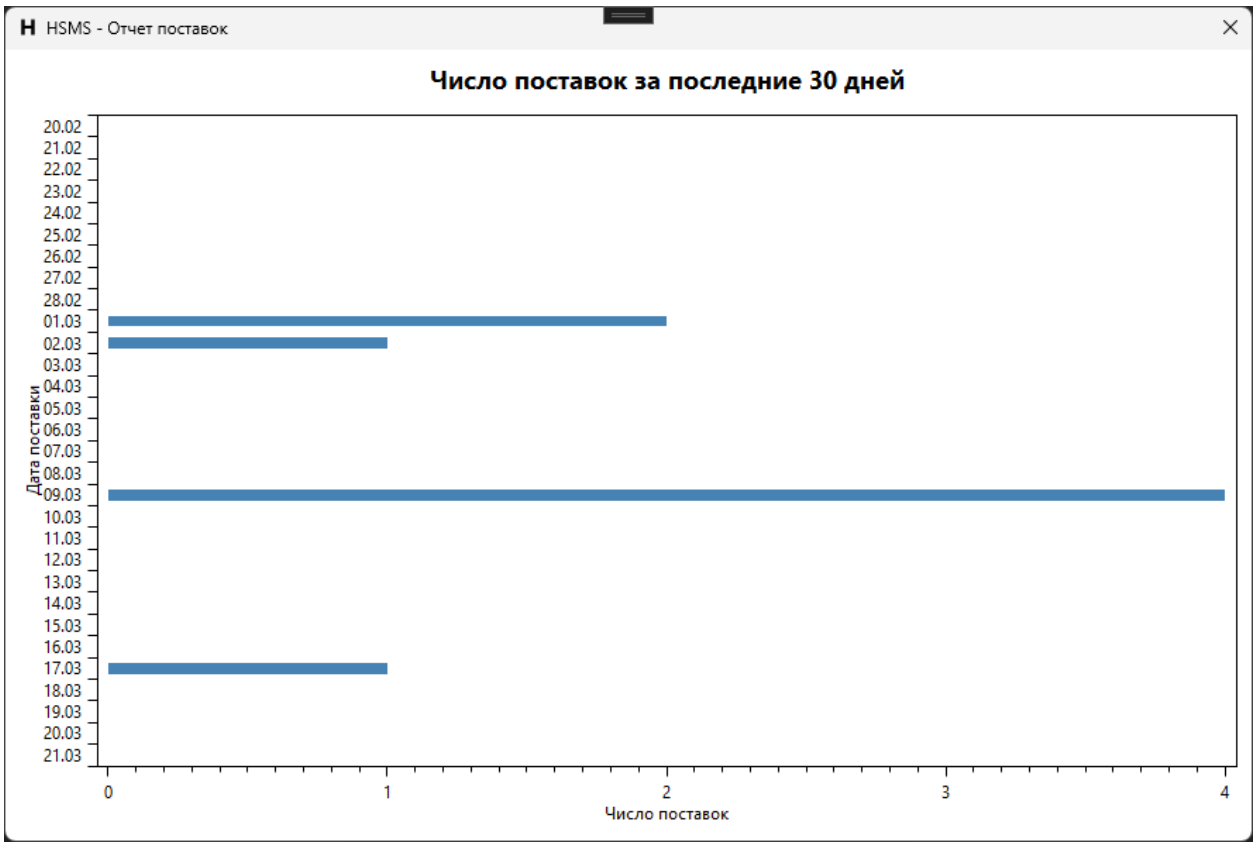


Рисунок 3.33 – График числа поставок за последние 30 дней

3.4.2.10 Вкладка «Заказы»

На рисунке 3.34 изображена таблица заказов. Для взаимодействия с ней доступны следующие инструменты:

1. Обновить (загрузить данные из базы данных заново).
2. Сохранить изменения, сделанные в таблице.
3. Отфильтровать данные по разным параметрам.
4. Построить график.
5. Экспортировать данные обо всех заказах в файл формата .xlsx.

Выполнение CRUD-операций происходит только через таблицу.

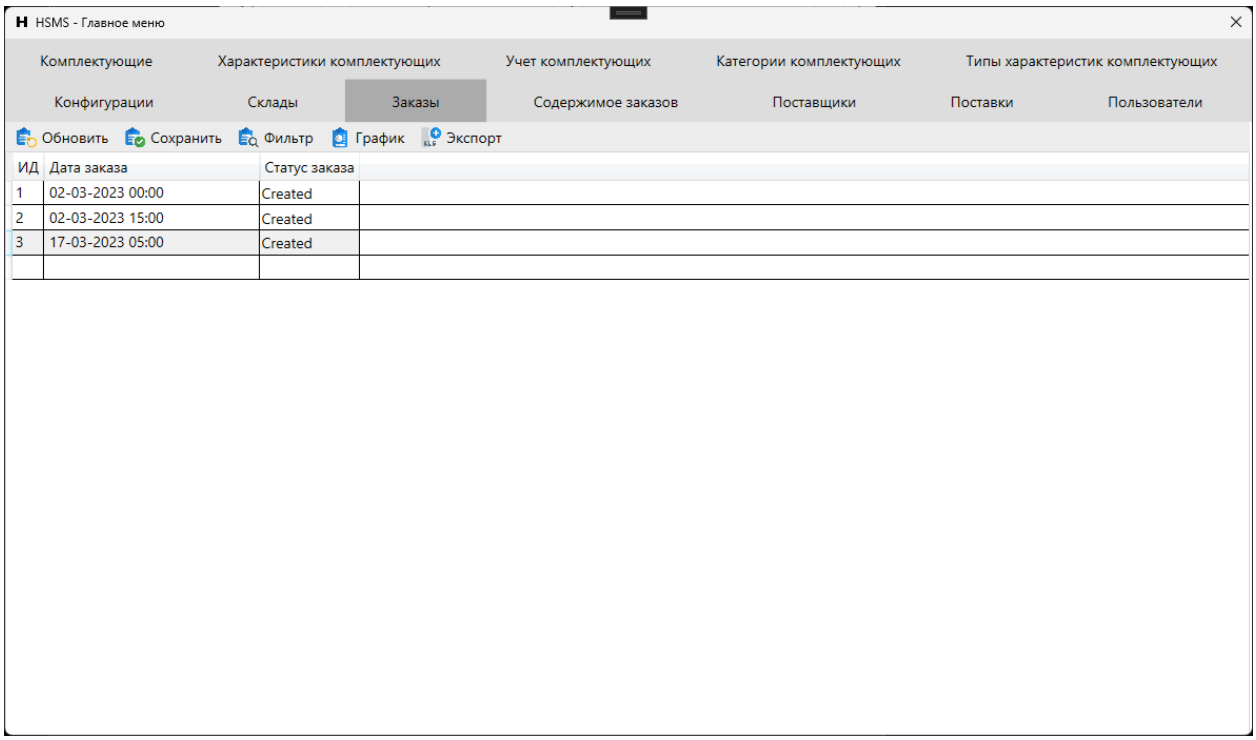


Рисунок 3.34 – Вкладка «Заказы»

При нажатии на кнопку «Фильтр» открывается окно «HSMS – Фильтр заказов» (рисунок 3.35). Необходимо заполнить те поля, по которым должен осуществляться фильтр данных. Остальные нужно оставить пустыми.

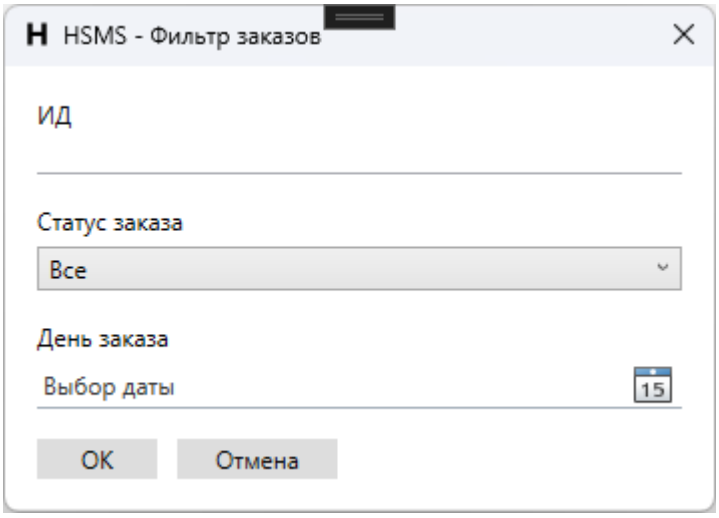


Рисунок 3.35 – Окно фильтра заказов

При нажатии на кнопку «График» открывается окно со столбчатой диаграммой, на которой изображено число заказов за последние 30 дней (рисунок 3.36).

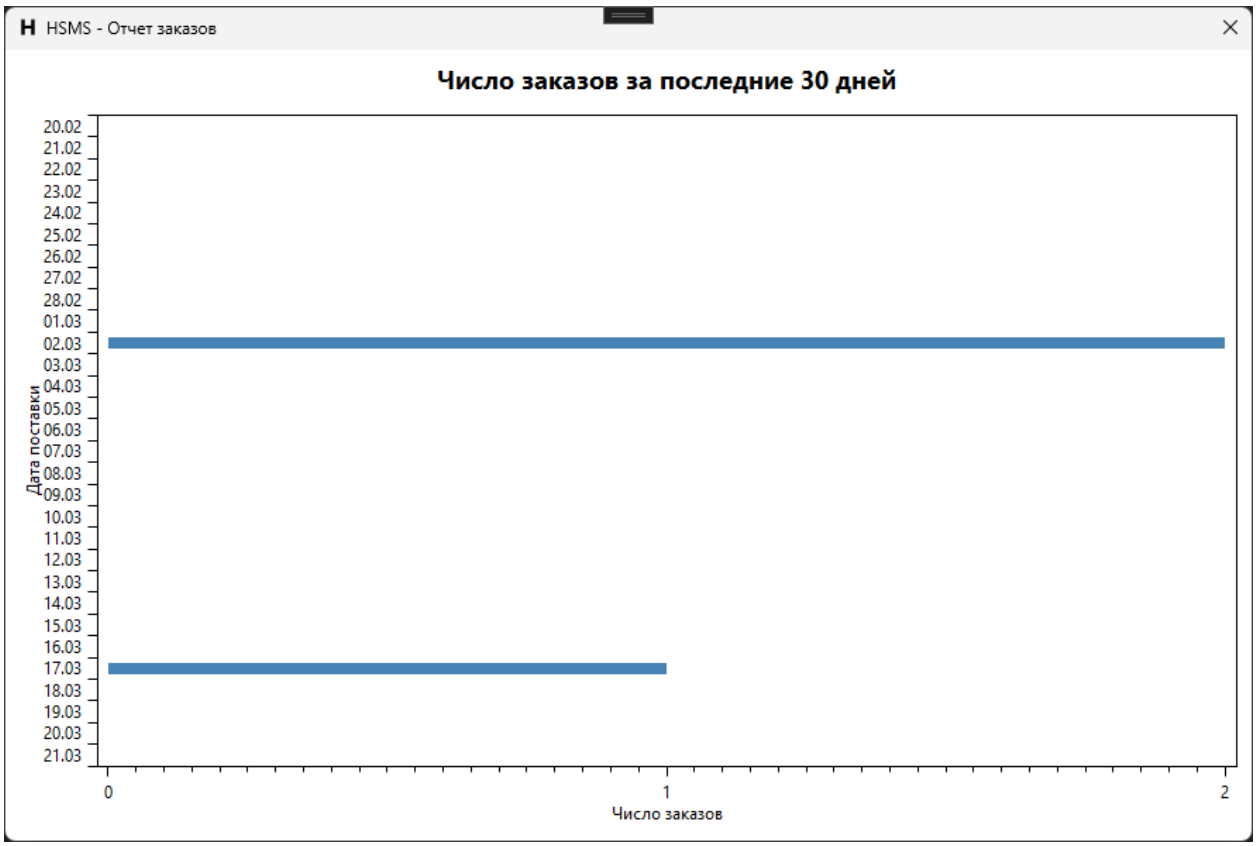


Рисунок 3.36 – График числа заказов за последние 30 дней

3.4.2.11 Вкладка «Содержимое заказов»

На рисунке 3.37 изображена таблица содержимого заказов. Для взаимодействия с ней доступны следующие инструменты:

1. Обновить (загрузить данные из базы данных заново).
2. Сохранить изменения, сделанные в таблице.
3. Отфильтровать данные по разным параметрам.
4. Экспортировать данные о содержимом всех заказов в файл формата .xlsx.

Выполнение CRUD-операций происходит только через таблицу.

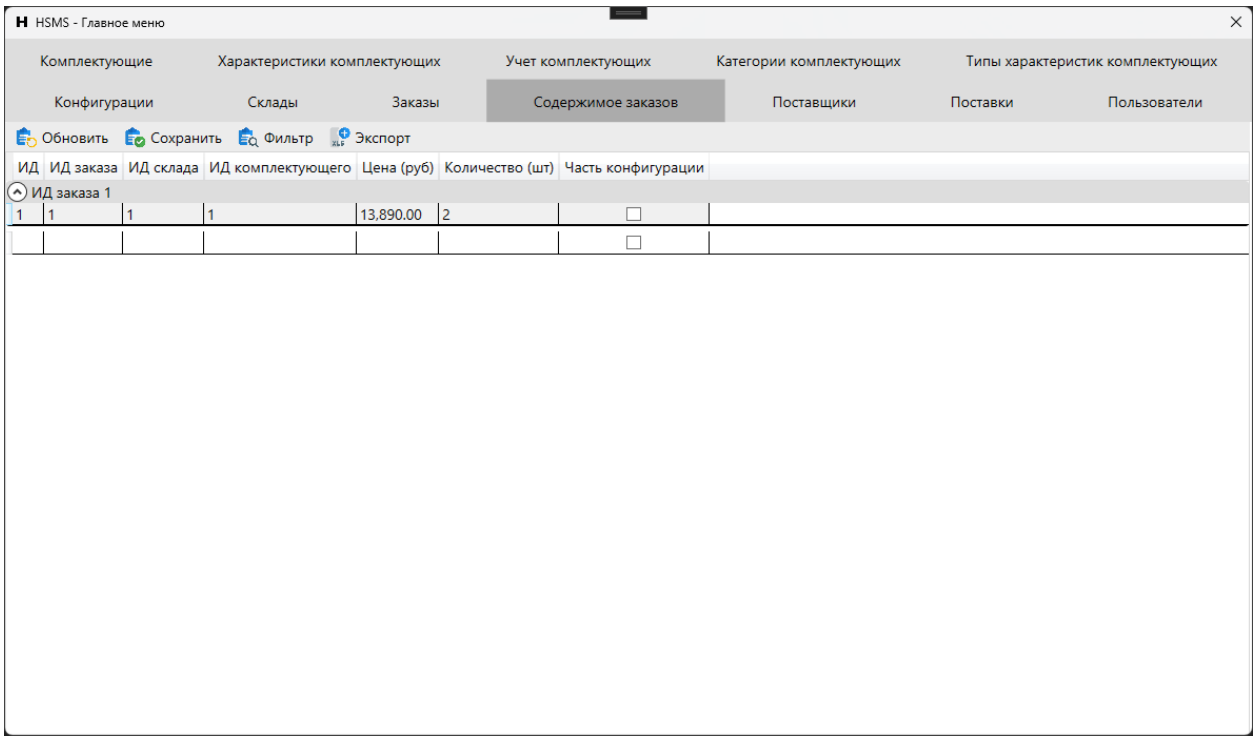


Рисунок 3.37 – Вкладка «Содержимое заказов»

При нажатии на кнопку «Фильтр» открывается окно «HSMS – Фильтр содержимого заказов» (рисунок 3.38). Необходимо заполнить те поля, по которым должен осуществляться фильтр данных. Остальные нужно оставить пустыми.

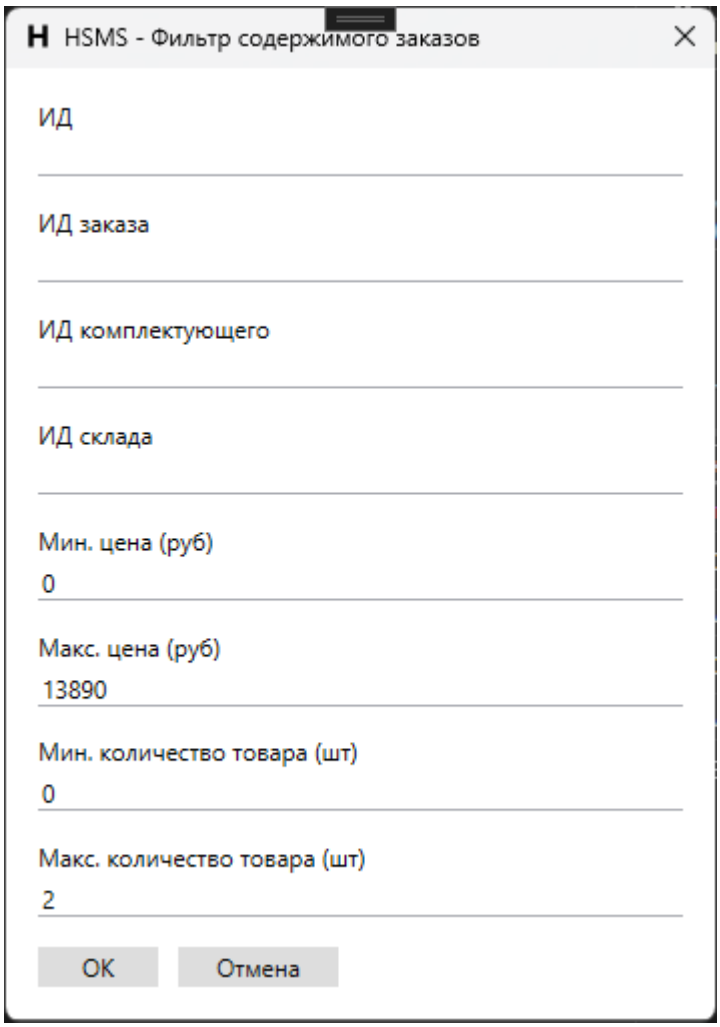


Рисунок 3.38 – Окно фильтра содержимого заказов

3.4.2.12 Вкладка «Конфигурации»

На рисунке 3.39 изображена таблица конфигураций. Для взаимодействия с ней доступны следующие инструменты:

1. Обновить (загрузить данные из базы данных заново).
2. Сохранить изменения, сделанные в таблице.
3. Отфильтровать данные по разным параметрам.
4. Экспортировать данные обо всех конфигурациях в файл формата .xlsx.

Выполнение CRUD-операций происходит только через таблицу.

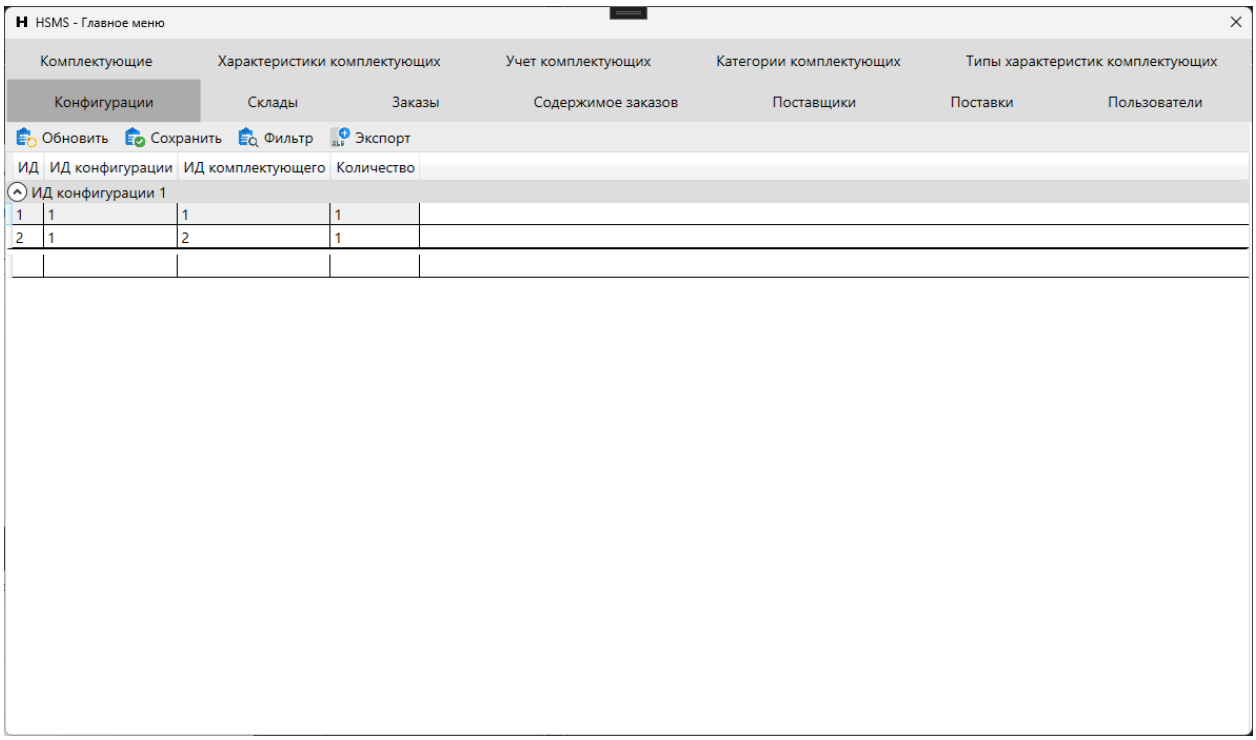


Рисунок 3.39 – Вкладка «Конфигурации»

При нажатии на кнопку «Фильтр» открывается окно «HSMS – Фильтр конфигураций» (рисунок 3.40). Необходимо заполнить те поля, по которым должен осуществляться фильтр данных. Остальные нужно оставить пустыми.

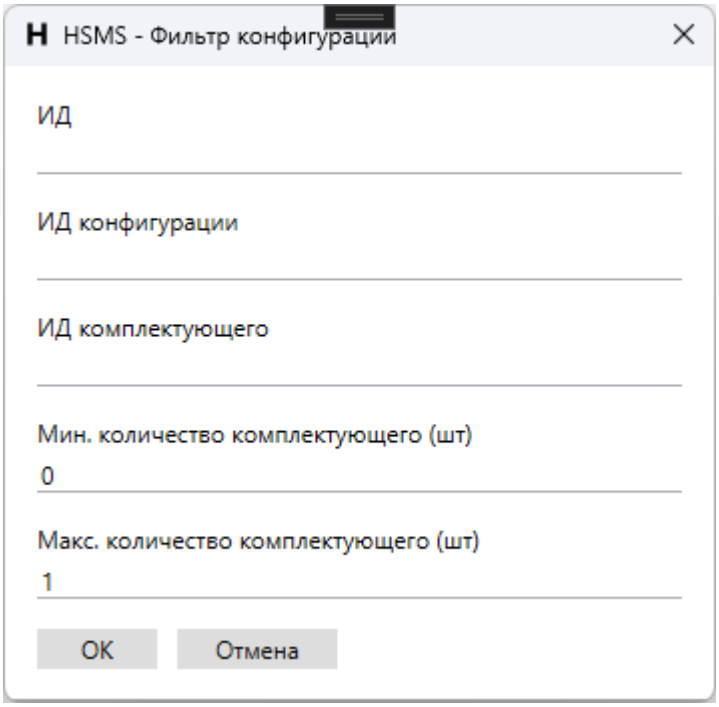


Рисунок 3.40 – Окно фильтра конфигураций

3.6 Выводы по третьей главе

По итогам выполнения этапа реализации были получены следующие результаты:

1. Выбраны инструментальные средства: приложение на языке программирования C# с использованием фреймворков WPF и Entity Framework, СУБД – PostgreSQL.
2. Созданы модели для взаимодействия с таблицами из базы данных.
3. Выполнена настройка подключения к СУБД.
4. Разработаны запросы для работы с базой данных.
5. Реализован интерфейс.

Заключение

В ходе выполнения курсовой работы на тему «Разработка информационной системы для персонала магазина компьютерных комплектующих» были достигнуты следующие результаты:

1. Проведен подробный анализ существующего решения – приложения на платформе «1С:Предприятие» - «1С:Розница 8. Магазин бытовой техники и средств связи».
2. Сформированы функциональные и нефункциональные требования, построена диаграмма вариантов с описанием всех прецедентов.
3. Выполнено проектирование базы данных с установлением функциональных зависимостей и нормализацией до третьей нормальной формы.
4. Спроектированы интерфейс приложения и отчеты.
5. Созданы модели данных, общее число которых равно 12.
6. Реализовано настольное приложение на языке программирования C# с использованием графического фреймворка WPF и библиотеки ORM – Entity Framework. В качестве системы управления базами данных выбран PostgreSQL, идеальное решение для малых и средних проектов.

Библиографический список

1. Джуба С. Изучаем PostgreSQL 10 / Джуба С., Волков А. : Packt Publishing, 2018. - 400 с. - ISBN 978-5-97060-643-8.
2. Гвоздева, Т.В. Проектирование информационных систем. Стандартизация: Учебное пособие / Т.В. Гвоздева, Б.А. Баллод. - СПб.: Лань, 2019. - 252 с.
3. Перлова, О.Н. Проектирование и разработка информационных систем: Учебник / О.Н. Перлова, О.П. Ляпина, А.В. Гусева. - М.: Academia, 2017. - 416 с.

ПРИЛОЖЕНИЕ А.
Код программы

```
using HardwareStore.Models;
using System;
using System.Collections.Generic;
using System.Collections.ObjectModel;
using System.ComponentModel.DataAnnotations;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace HardwareStore.Models
{
   public class Component
   {
      public int Id { get; set; }

      public int ComponentTypeId { get; set; }
      public virtual ComponentType ComponentType { get; set; } = null!;

      [StringLength(256)]
      public required string Name { get; set; }
      public int Warranty { get; set; }

      public virtual ICollection<ComponentDetail> ComponentDetails { get; private set; }
= new ObservableCollection<ComponentDetail>();
      public virtual ICollection<ComponentStorage> ComponentStorages { get; private
set; } = new ObservableCollection<ComponentStorage>();
      public virtual ICollection<OrderComponent> OrderComponents { get; private set; }
= new ObservableCollection<OrderComponent>();
      public virtual ICollection<Configuration> Configurations { get; private set; } = new
ObservableCollection<Configuration>();
      public virtual ICollection<Supply> Supplies { get; private set; } = new
ObservableCollection<Supply>();
   }
}
```

Листинг А.1 – Код модели «Component»

```
using Microsoft.EntityFrameworkCore;
using System;
using System.Collections.Generic;
using System.ComponentModel.DataAnnotations;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace HardwareStore.Models
{
    [Index(nameof(DetailTypeId), nameof(ComponentId), IsUnique = true)]
    public class ComponentDetail
    {
        public int Id { get; set; }

        public int DetailTypeId { get; set; }
        public virtual DetailType DetailType { get; set; } = null!;

        public int ComponentId { get; set; }
        public virtual Component Component { get; set; } = null!;

        [StringLength(128)]
        public required string Value { get; set; }
    }
}
```

Листинг А.2 – Код модели «ComponentDetail»

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace HardwareStore.Models
{
   public class ComponentStorage
   {
      public int Id { get; set; }
      public float Price { get; set; }
      public int Amount { get; set; }

      public int WarehouseId { get; set; }
      public virtual Warehouse Warehouse { get; set; } = null!;

      public int ComponentId { get; set; }
      public virtual Component Component { get; set; } = null!;
   }
}
```

Листинг А.3 – Код модели «ComponentStorage»

```
using Microsoft.EntityFrameworkCore;
using System;
using System.Collections.Generic;
using System.Collections.ObjectModel;
using System.ComponentModel.DataAnnotations;
using System.Linq;
using System.Net.Mail;
using System.Text;
using System.Threading.Tasks;

namespace HardwareStore.Models
{
   [Index(nameof(Name), IsUnique = true)]
   public class ComponentType
   {
      public int Id { get; set; }

      [StringLength(64)]
      public required string Name { get; set; }
      public virtual ICollection<Component> Components { get; private set; } = new
ObservableCollection<Component>();
   }
}
```

Листинг А.4 – Код модели «ComponentType»

```
using Microsoft.EntityFrameworkCore;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace HardwareStore.Models
{
    [Index(nameof(ConfigurationId), nameof(ComponentId), IsUnique = true)]
    public class Configuration
    {
        public int Id { get; set; }
        public int ConfigurationId { get; set; }
        public int Amount { get; set; }

        public int ComponentId { get; set; }
        public virtual Component Component { get; set; } = null!;
    }
}
```

Листинг А.5 – Код модели «Configuration»

```
using Microsoft.EntityFrameworkCore;
using System;
using System.Collections.Generic;
using System.Collections.ObjectModel;
using System.ComponentModel;
using System.ComponentModel.DataAnnotations;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace HardwareStore.Models
{
   [Index(nameof(Name), IsUnique = true)]
   public class DetailType
   {
      public int Id { get; set; }

      [StringLength(64)]
      public required string Name { get; set; }
      public virtual ICollection<ComponentDetail> ComponentDetails { get; private set; }
= new ObservableCollection<ComponentDetail>();
   }
}
```

Листинг А.6 – Код модели «DetailType»

```
using HardwareStore.Models;
using System;
using System.Collections.Generic;
using System.Collections.ObjectModel;
using System.ComponentModel;
using System.ComponentModel.DataAnnotations.Schema;
using System.Linq;
using System.Reflection.Metadata;
using System.Text;
using System.Text.RegularExpressions;
using System.Threading.Tasks;
using System.Windows.Documents;

namespace HardwareStore.Models
{
    public enum OrderStatus : int
    {
        [Description("Создан")]
        Created = 0,
        [Description("В обработке")]
        Processing = 1,
        [Description("Готов")]
        Ready = 2,
        [Description("Получен")]
        Received = 3,
        [Description("Отменен")]
        Canceled = 4
    }

    public class Order
    {
        public int Id { get; set; }
        public long Date { get; set; }

        [NotMapped]
        public DateTime DateHandler
        {
            get
            {
                return Utilities.UnixTimeStampToDateTime(Date);
            }
            set
            {
                Date = Utilities.DateTimeToUnixTimeStamp(value);
            }
        }

        public OrderStatus Status { get; set; }
```

```
        public virtual ICollection<OrderComponent> OrderComponents { get; private set; }
= new ObservableCollection<OrderComponent>();
    }
}
```

Листинг А.7 – Код модели «Order»

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace HardwareStore.Models
{
    public class OrderComponent
    {
        public int Id { get; set; }
        public float Price { get; set; }
        public int Amount { get; set; }
        public bool IsPartOfConfiguration { get; set; }

        public int OrderId { get; set; }
        public virtual Order Order { get; set; } = null!;

        public int WarehouseId { get; set; }
        public virtual Warehouse Warehouse { get; set; } = null!;

        public int ComponentId { get; set; }
        public virtual Component Component { get; set; } = null!;
    }
}
```

Листинг А.8 – Код модели «OrderComponent»

```
using HardwareStore.Models;
using Microsoft.EntityFrameworkCore;
using System;
using System.Collections.Generic;
using System.Collections.ObjectModel;
using System.ComponentModel;
using System.ComponentModel.DataAnnotations;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace HardwareStore.Models
{
   [Index(nameof(Name), IsUnique = true)]
   public class Supplier
   {
      public int Id { get; set; }

      [StringLength(128)]
      public required string Name { get; set; }
      public virtual ICollection<Supply> Supplies { get; private set; } = new
ObservableCollection<Supply>();
   }
}
```

Листинг А.9 – Код модели «Supplier»

```
using System;
using System.Collections.Generic;
using System.Collections.ObjectModel;
using System.ComponentModel.DataAnnotations.Schema;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace HardwareStore.Models
{
    public class Supply
    {
        public int Id { get; set; }
        public float SupplyPrice { get; set; }
        public int Amount { get; set; }
        public long Date { get; set; }

        [NotMapped]
        public DateTime DateHandler
        {
            get
            {
                return Utilities.UnixTimeStampToDateTime(Date);
            }
            set
            {
                Date = Utilities.DateTimeToUnixTimeStamp(value);
            }
        }

        public int SupplierId { get; set; }
        public virtual Supplier Supplier { get; set; } = null!;

        public int ComponentId { get; set; }
        public virtual Component Component { get; set; } = null!;

        public int WarehouseId { get; set; }
        public virtual Warehouse Warehouse { get; set; } = null!;
    }
}
```

Листинг А.10 – Код модели «Supply»

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.ComponentModel.DataAnnotations.Schema;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using Microsoft.EntityFrameworkCore;

namespace HardwareStore.Models
{
    public enum Group : int
    {
        [Description("Продавец-консультант")]
        Consultant = 0,

        [Description("Менеджер по закупкам")]
        PurchasingManager = 1,

        [Description("Складской работник")]
        WarehouseWorker = 2,

        [Description("Аналитик")]
        Analyst = 3,

        [Description("Администратор")]
        Admin = 4,
    }

    public class User
    {
        public int Id { get; set; }
        public required string Password { get; set; }
        public long RegistrationDate { get; set; }
        public long? LastVisitDate { get; set; }
        public Group Group { get; set; }

        [NotMapped]
        public string RegistrationDateText
        {
            get
            {
                DateTime dateTime =
Utilities.UnixTimeStampToDateTime(RegistrationDate);
                return dateTime.ToString("dd-MM-yyyy HH:mm");
            }
        }
```

```
        [NotMapped]
        public string LastVisitDateText
        {
            get
            {
                if (LastVisitDate == null)
                    return "";

                DateTime dateTime =
Utilities.UnixTimeStampToDateTime((long)LastVisitDate);
                return dateTime.ToString("dd-MM-yyyy HH:mm");
            }
        }
    }
}
```

Листинг А.11 – Код модели «User»

```
using HardwareStore.Models;
using Microsoft.EntityFrameworkCore;
using System;
using System.Collections.Generic;
using System.Collections.ObjectModel;
using System.ComponentModel;
using System.ComponentModel.DataAnnotations;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace HardwareStore.Models
{
   [Index(nameof(City), nameof(Street), nameof(House), IsUnique = true)]
   public class Warehouse
   {
      public int Id { get; set; }

      [StringLength(64)]
      public required string City { get; set; }

      [StringLength(64)]
      public required string Street { get; set; }

      [StringLength(8)]
      public required string House { get; set; }

      public int Zip { get; set; }
      public virtual ICollection<ComponentStorage> ComponentStorages { get; private
set; } = new ObservableCollection<ComponentStorage>();
      public virtual ICollection<OrderComponent> OrderComponents { get; private set; }
= new ObservableCollection<OrderComponent>();
      public virtual ICollection<Supply> Supplies { get; private set; } = new
ObservableCollection<Supply>();
   }
}
```

Листинг А.12 – Код модели «Warehouse»

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using HardwareStore.Models;

namespace HardwareStore
{
   public struct PermissionOptions
   {
      public PermissionOptions() { }

      public PermissionOptions(bool CanAdd, bool CanEdit, bool CanDelete, bool
CanView)
      {
         this.CanAdd = CanAdd;
         this.CanEdit = CanEdit;
         this.CanDelete = CanDelete;
         this.CanView = CanView;
      }

      public PermissionOptions(string permission)
      {
         if (permission.Length != 4 || (permission.Count(c => c == '0') +
permission.Count(c => c == '1')) != 4)
            throw new ArgumentException("Permission string does not fit the format!");

         CanAdd = permission[0] == '1';
         CanEdit = permission[1] == '1';
         CanDelete = permission[2] == '1';
         CanView = permission[3] == '1';
      }

      public bool CanAdd { get; set; }
      public bool CanEdit { get; set; }
      public bool CanDelete { get; set; }
      public bool CanView { get; set; }
   }

   public class GroupPermissions
   {
      public static int UserId { get; set; }

      public static Group Group
      {
         get
         {
```

```
        using StoreContext context = new();

        User? user = context.Users.Find(UserId);

        return (user != null) ? user.Group : Group.Consultant;
    }
}

public static Dictionary<Group, GroupPermissions> permissions = new()
{
    {
        Group.Consultant, new GroupPermissions
        {
            Components = new("1111"),
            ComponentDetails = new("1111"),
            ComponentStorages = new("0101"),
            ComponentTypes = new("0001"),
            Configurations = new("1111"),
            DetailTypes = new("0001"),
            Orders = new("1111"),
            OrderComponents = new("1111"),
            Suppliers = new("0001"),
            Supplies = new("0001"),
            Warehouses = new("0001"),
            Users = new("0000")
        }
    },
    {
        Group.PurchasingManager, new GroupPermissions
        {
            Components = new("0001"),
            ComponentDetails = new("0001"),
            ComponentStorages = new("0001"),
            ComponentTypes = new("0001"),
            Configurations = new("0001"),
            DetailTypes = new("0001"),
            Orders = new("0001"),
            OrderComponents = new("0001"),
            Suppliers = new("1111"),
            Supplies = new("1111"),
            Warehouses = new("0001"),
            Users = new("0000")
        }
    },
    {
        Group.WarehouseWorker, new GroupPermissions
        {
            Components = new("0001"),
```

```
            ComponentDetails = new("0001"),
            ComponentStorages = new("1111"),
            ComponentTypes = new("0001"),
            Configurations = new("0001"),
            DetailTypes = new("0001"),
            Orders = new("0001"),
            OrderComponents = new("0001"),
            Suppliers = new("0001"),
            Supplies = new("0001"),
            Warehouses = new("1111"),
            Users = new("0000")
        }
    },
    {
        Group.Analyst, new GroupPermissions
        {
            Components = new("0001"),
            ComponentDetails = new("0001"),
            ComponentStorages = new("0001"),
            ComponentTypes = new("0001"),
            Configurations = new("0001"),
            DetailTypes = new("0001"),
            Orders = new("0001"),
            OrderComponents = new("0001"),
            Suppliers = new("0000"),
            Supplies = new("0000"),
            Warehouses = new("0001"),
            Users = new("0000")
        }
    },
    {
        Group.Admin, new GroupPermissions
        {
            Components = new("1111"),
            ComponentDetails = new("1111"),
            ComponentStorages = new("1111"),
            ComponentTypes = new("1111"),
            Configurations = new("1111"),
            DetailTypes = new("1111"),
            Orders = new("1111"),
            OrderComponents = new("1111"),
            Suppliers = new("1111"),
            Supplies = new("1111"),
            Warehouses = new("1111"),
            Users = new("1111")
        }
    }
};
```

```
        public PermissionOptions Components { get; set; }
        public PermissionOptions ComponentDetails { get; set; }
        public PermissionOptions ComponentStorages { get; set; }
        public PermissionOptions ComponentTypes { get; set; }
        public PermissionOptions Configurations { get; set; }
        public PermissionOptions DetailTypes { get; set; }
        public PermissionOptions Orders { get; set; }
        public PermissionOptions OrderComponents { get; set; }
        public PermissionOptions Suppliers { get; set; }
        public PermissionOptions Supplies { get; set; }
        public PermissionOptions Warehouses { get; set; }
        public PermissionOptions Users { get; set; }
    }
}
```

Листинг А.13 – Код класса «GroupPermission»

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Text.RegularExpressions;
using System.Threading.Tasks;

namespace HardwareStore
{
   public static partial class RegularExpressions
   {
      [GeneratedRegex(@"[^\d]", RegexOptions.IgnoreCase, "en-US")]
      public static partial Regex GetUnsignedIntRegex();

      [GeneratedRegex(@"[^\d,]", RegexOptions.IgnoreCase, "en-US")]
      public static partial Regex GetUnsignedFloatRegex();

      [GeneratedRegex(@"[^a-zA-Za-яA-Я- ]", RegexOptions.IgnoreCase, "en-US")]
      public static partial Regex GetCityRegex();

      [GeneratedRegex(@"[^a-zA-Za-яA-Я- ]", RegexOptions.IgnoreCase, "en-US")]
      public static partial Regex GetStreetRegex();

      [GeneratedRegex(@"[^a-zA-Za-яA-Я-'\"" ]", RegexOptions.IgnoreCase, "en-US")]
      public static partial Regex GetSupplierNameRegex();

      [GeneratedRegex(@"[^a-zA-Za-яA-Я-()\d.,/'\"" ]", RegexOptions.IgnoreCase, "en-
US")]
      public static partial Regex GetComponentTypeRegex();

      [GeneratedRegex(@"[^a-zA-Za-яA-Я-()\d.,/ ]", RegexOptions.IgnoreCase, "en-
US")]
      public static partial Regex GetDetailTypeRegex();

      [GeneratedRegex(@"[^a-zA-Za-яA-Я-\d\[\]().,/ ]", RegexOptions.IgnoreCase, "en-
US")]
      public static partial Regex GetComponentNameRegex();
   }
}
```

Листинг А.14 – Код класса «RegularExpressions»

```
using HardwareStore.Models;
using Microsoft.EntityFrameworkCore;
using Npgsql;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows;
using System.Xml;

namespace HardwareStore
{
    public readonly struct DbData
    {
        public static readonly string Database = "hardware_store_ms";
        public static readonly string Username = "postgres";
        public static readonly string Password = "root";
        public static readonly string Host = "localhost";
        public static readonly string Port = "5433";
    }

    public class StoreContext : DbContext
    {
        public DbSet<User> Users { get; set; }
        public DbSet<Component> Components { get; set; }
        public DbSet<ComponentDetail> ComponentDetails { get; set; }
        public DbSet<ComponentStorage> ComponentStorages { get; set; }
        public DbSet<ComponentType> ComponentTypes { get; set; }
        public DbSet<Order> Orders { get; set; }
        public DbSet<OrderComponent> OrderComponents { get; set; }
        public DbSet<Supplier> Suppliers { get; set; }
        public DbSet<Supply> Supplies { get; set; }
        public DbSet<Warehouse> Warehouses { get; set; }
        public DbSet<Configuration> Configurations { get; set; }
        public DbSet<DetailType> DetailTypes { get; set; }

        public StoreContext()
        {
            try
            {
                Database.EnsureCreated();
            }
            catch (NpgsqlException)
            {
                MessageBox.Show("Нет доступа к базе данных, обратитесь к системному
администратору!", "Ошибка");
                Environment.Exit(0);
```

```
            }
        }

        protected override void OnConfiguring(DbContextOptionsBuilder optionsBuilder)
        {

optionsBuilder.UseNpgsql($"Host={DbData.Host};Port={DbData.Port};Database={DbD
ata.Database};Username={DbData.Username};Password={DbData.Password}");
            optionsBuilder.UseLazyLoadingProxies();
        }
    }
}
```

Листинг А.15 – Код класса «StoreContext»

```
using HardwareStore.Models;
using Microsoft.EntityFrameworkCore;
using Microsoft.Win32;
using OfficeOpenXml;
using System;
using System.Collections.Generic;
using System.IO;
using System.Linq;
using System.Security.Cryptography;
using System.Text;
using System.Text.RegularExpressions;
using System.Threading.Tasks;
using System.Windows.Controls;
using System.Windows;

namespace HardwareStore
{
    public static class Utilities
    {
        public static string GetHash(HashAlgorithm hashAlgorithm, string input)
        {
            byte[] data = hashAlgorithm.ComputeHash(Encoding.UTF8.GetBytes(input));
            var sBuilder = new StringBuilder();

            for (int i = 0; i < data.Length; i++)
                sBuilder.Append(data[i].ToString("x2"));

            return sBuilder.ToString();
        }

        public static DateTime UnixTimeStampToDateTime(long unixTimeStamp)
        {
            DateTime dateTime = new(1970, 1, 1, 0, 0, 0, 0, DateTimeKind.Utc);
            dateTime = dateTime.AddSeconds(unixTimeStamp).ToLocalTime();
            return dateTime;
        }

        public static long DateTimeToUnixTimeStamp(DateTime time)
        {
            return ((DateTimeOffset)time).ToUnixTimeSeconds();
        }

        public static long GetCurrentUnixTimeStamp()
        {
            return new DateTimeOffset(DateTime.UtcNow).ToUnixTimeSeconds();
        }
```

```
        public static void ValidateTextBox(Regex regex, object sender,
TextChangedEventArgs e)
        {
            string correct = regex.Replace(((TextBox)sender).Text, "");

            ((TextBox)sender).Text = correct;
            ((TextBox)sender).CaretIndex = ((TextBox)sender).Text.Length;
        }

        public static int TryParse(string source, int defaultValue)
        {
            if (int.TryParse(source, out int result))
                return result;
            else

                return defaultValue;
        }

        public static float TryParse(string source, float defaultValue)
        {
            if (float.TryParse(source, out float result))
                return result;
            else
                return defaultValue;
        }

        public static void ExportToExcel(string worksheetName, List<string[]> headerRow,
List<object[]> cellData)
        {
            SaveFileDialog saveFileDialog = new()
            {
                Filter = "Excel files (*.xlsx)|*.xlsx"
            };

            if (saveFileDialog.ShowDialog() == true)
            {
                ExcelPackage.LicenseContext = LicenseContext.NonCommercial;

                using ExcelPackage excel = new();

                excel.Workbook.Worksheets.Add(worksheetName);

                string headerRange = "A1:" + Char.ConvertFromUtf32(headerRow[0].Length
+ 64) + "1";

                ExcelWorksheet worksheet = excel.Workbook.Worksheets[worksheetName];
                worksheet.Cells[headerRange].LoadFromArrays(headerRow);
```

```
            worksheet.Cells[2, 1].LoadFromArrays(cellData);

            FileInfo excelFile = new(saveFileDialog.FileName);
            excel.SaveAs(excelFile);

            MessageBox.Show("Данные успешно экспортированы!", "Успех");
        }
    }

    public static void HandleSaveDataGrid(in StoreContext context, in DataGrid
dataGrid)
    {
        string messageError = "Убедитесь в правительности заполнения всех данных!
После добавления или изменения записи необходимо нажать Enter.";

        try
        {
            context.SaveChanges();
            dataGrid.Items.Refresh();
        }
        catch (DbUpdateException)
        {
            MessageBox.Show(messageError, "Ошибка");
        }
        catch (InvalidOperationException)
        {
            MessageBox.Show(messageError, "Ошибка");
        }
    }
  }
}
```

Листинг А.16 – Код класса «Utilities»

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Linq;
using System.Security.Cryptography;
using System.Text;
using System.Text.RegularExpressions;
using System.Threading.Tasks;
using System.Windows;
using System.Windows.Controls;
using System.Windows.Data;
using System.Windows.Documents;
using System.Windows.Input;
using System.Windows.Media;
using System.Windows.Media.Imaging;
using System.Windows.Shapes;
using HardwareStore.Models;
using Npgsql;

namespace HardwareStore.Windows
{
    /// <summary>
    /// Логика взаимодействия для LoginWindow.xaml
    /// </summary>
    public partial class LoginWindow : Window
    {
        public LoginWindow()
        {
            InitializeComponent();
        }

        private void loginButton_Click(object sender, RoutedEventArgs e)
        {
            if (loginTextBox.Text.Length == 0 || passwordBox.Password.Length == 0)
            {
                MessageBox.Show("Пожалуйста, заполните все поля!", "Ошибка");
                return;
            }

            int id = int.Parse(loginTextBox.Text);
            string encryptedPassword = Utilities.GetHash(SHA256.Create(),
passwordBox.Password);

            using StoreContext context = new();

            User? user = context.Users.Where(u => u.Id == id && u.Password ==
encryptedPassword).FirstOrDefault();
```

```
            if (user != null)
            {
                user.LastVisitDate = Utilities.GetCurrentUnixTimeStamp();
                context.SaveChanges();

                GroupPermissions.UserId = user.Id;

                MainWindow mainWindow = new();
                Close();
                mainWindow.Show();
            }
            else
            {
                MessageBox.Show("Данного пользователя не существует!", "Ошибка");
            }
        }

        private void loginTextBox_TextChanged(object sender, TextChangedEventArgs e)
        {
            string correctLogin =
RegularExpressions.GetUnsignedIntRegex().Replace(((TextBox)sender).Text, "");
            ((TextBox)sender).Text = correctLogin;

            ((TextBox)sender).CaretIndex = ((TextBox)sender).Text.Length;
        }
    }
}
```

Листинг А.17 – Код класса «LoginWindow»

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows;
using System.Windows.Controls;
using System.Windows.Data;
using System.Windows.Documents;
using System.Windows.Input;
using System.Windows.Media;
using System.Windows.Media.Imaging;
using System.Windows.Navigation;
using System.Windows.Shapes;

namespace HardwareStore.Windows
{
    /// <summary>
    /// Interaction logic for MainWindow.xaml
    /// </summary>
    public partial class MainWindow : Window
    {
        public MainWindow()
        {
            InitializeComponent();
        }

        private void Window_Loaded(object sender, RoutedEventArgs e)
        {
            var visibility = (bool permissionOption) => (permissionOption) ?
Visibility.Visible : Visibility.Collapsed;

            componentsTab.Visibility =
visibility(GroupPermissions.permissions[GroupPermissions.Group].Components.CanVie
w);
            componentDetailsTab.Visibility =
visibility(GroupPermissions.permissions[GroupPermissions.Group].ComponentDetails.C
anView);
            componentStoragesTab.Visibility =
visibility(GroupPermissions.permissions[GroupPermissions.Group].ComponentStorages.
CanView);
            componentTypesTab.Visibility =
visibility(GroupPermissions.permissions[GroupPermissions.Group].ComponentTypes.Ca
nView);
            configurationsTab.Visibility =
visibility(GroupPermissions.permissions[GroupPermissions.Group].Configurations.Can
View);
```

```
            detailTypesTab.Visibility =
visibility(GroupPermissions.permissions[GroupPermissions.Group].DetailTypes.CanVie
w);
            ordersTab.Visibility =
visibility(GroupPermissions.permissions[GroupPermissions.Group].Orders.CanView);
            orderComponentsTab.Visibility =
visibility(GroupPermissions.permissions[GroupPermissions.Group].OrderComponents.C
anView);
            suppliersTab.Visibility =
visibility(GroupPermissions.permissions[GroupPermissions.Group].Suppliers.CanView);
            suppliesTab.Visibility =
visibility(GroupPermissions.permissions[GroupPermissions.Group].Supplies.CanView);
            usersTab.Visibility =
visibility(GroupPermissions.permissions[GroupPermissions.Group].Users.CanView);
            warehousesTab.Visibility =
visibility(GroupPermissions.permissions[GroupPermissions.Group].Warehouses.CanVie
w);

            foreach (TabItem tabItem in tabControl.Items)
            {
                if (tabItem.Visibility == Visibility.Visible)
                {
                    tabControl.SelectedIndex = tabItem.TabIndex;
                    break;
                }
            }
        }
    }
}
```

Листинг А.18 – Код класса «MainWindow»

```
using HardwareStore.Windows.Components;
using Microsoft.EntityFrameworkCore;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows;
using System.Windows.Controls;
using System.Windows.Data;
using System.Windows.Documents;
using System.Windows.Input;
using System.Windows.Media;
using System.Windows.Media.Imaging;
using System.Windows.Navigation;
using System.Windows.Shapes;
using HardwareStore.Models;
using System.IO;
using OfficeOpenXml;
using Microsoft.Win32;
using System.ComponentModel;

namespace HardwareStore.Windows.ComponentDetails
{
    /// <summary>
    /// Логика взаимодействия для ComponentDetailsUserControl.xaml
    /// </summary>
    public partial class ComponentDetailsUserControl : UserControl
    {
        private StoreContext _context = new();

        private readonly CollectionViewSource componentDetailsViewSource;
        private readonly CollectionViewSource detailTypesViewSource;

        private ComponentDetailsFilter Filter { get; set; }

        public ComponentDetailsUserControl()
        {
            InitializeComponent();
            componentDetailsViewSource =
(CollectionViewSource)FindResource(nameof(componentDetailsViewSource));
            detailTypesViewSource =
(CollectionViewSource)FindResource(nameof(detailTypesViewSource));
            Filter = new();
        }

        private void UserControl_Loaded(object sender, RoutedEventArgs e)
        {
```

```
            _context.ComponentDetails.Load();
            _context.DetailTypes.Load();

            componentDetailsViewSource.GroupDescriptions.Clear();
            componentDetailsViewSource.GroupDescriptions.Add(new
PropertyGroupDescription("ComponentId"));
            componentDetailsViewSource.Source =
_context.ComponentDetails.Local.ToObservableCollection();

            detailTypesViewSource.Source =
_context.DetailTypes.Local.ToObservableCollection();

            componentDetailsDataGrid.CanUserAddRows =
GroupPermissions.permissions[GroupPermissions.Group].ComponentDetails.CanAdd;
            componentDetailsDataGrid.CanUserDeleteRows =
GroupPermissions.permissions[GroupPermissions.Group].ComponentDetails.CanDelete;
            componentDetailsDataGrid.IsReadOnly =
!GroupPermissions.permissions[GroupPermissions.Group].ComponentDetails.CanEdit;
        }

        private void updateComponentDetails_Click(object sender, RoutedEventArgs e)
        {
            Filter = new();

            _context = new StoreContext();
            _context.ComponentDetails.Load();
            _context.DetailTypes.Load();

            componentDetailsViewSource.GroupDescriptions.Clear();
            componentDetailsViewSource.GroupDescriptions.Add(new
PropertyGroupDescription("ComponentId"));
            componentDetailsViewSource.Source =
_context.ComponentDetails.Local.ToObservableCollection();

            detailTypesViewSource.Source =
_context.DetailTypes.Local.ToObservableCollection();
        }

        private void saveComponentDetails_Click(object sender, RoutedEventArgs e)
        {
            Utilities.HandleSaveDataGrid(_context, componentDetailsDataGrid);
        }

        private void filterComponentDetails_Click(object sender, RoutedEventArgs e)
        {
            FilterComponentDetailsWindow filterComponentDetailsWindow = new()
            {
                Filter = Filter
```

```
            };

            if (filterComponentDetailsWindow.ShowDialog() == true)
            {
                Filter = filterComponentDetailsWindow.Filter;

CollectionViewSource.GetDefaultView(componentDetailsDataGrid.ItemsSource).Refres
h();
            }
        }

        private void exportComponentDetails_Click(object sender, RoutedEventArgs e)
        {
            List<string[]> headerRow = new() { new string[] { "ИД", "ИД
комплектующего", "ИД характеристики", "Название характеристики", "Значение" }
};
            List<object[]> cellData = new() { };

            foreach (ComponentDetail? componentDetail in
_context.ComponentDetails.ToList())
            {
                if (componentDetail == null)
                    continue;

                cellData.Add(new object[] { componentDetail.Id,
componentDetail.ComponentId, componentDetail.DetailTypeId,
componentDetail.DetailType.Name, componentDetail.Value });
            }

            Utilities.ExportToExcel("Характеристики комплектующих", headerRow,
cellData);
        }

        private void CollectionViewSource_Filter(object sender, FilterEventArgs e)
        {
            if (e.Item is ComponentDetail componentDetail)
            {
                if (!Filter.IsFiltered)
                {
                    e.Accepted = true;
                    return;
                }

                e.Accepted = (Filter.Id == -1 || Filter.Id == componentDetail.Id)
                    && (Filter.ComponentId == -1 || Filter.Id ==
componentDetail.ComponentId)
                    && (Filter.DetailTypeId == -1 || Filter.Id == componentDetail.DetailTypeId)
                    && (Filter.Value.Length == 0 || Filter.Value == componentDetail.Value);
```

```
            }
        }
    }
}
```

Листинг А.19 – Код класса «ComponentDetailsUserControl»

```
using HardwareStore.Models;
using HardwareStore.Windows.Components;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows;
using System.Windows.Controls;
using System.Windows.Data;
using System.Windows.Documents;
using System.Windows.Input;
using System.Windows.Media;
using System.Windows.Media.Imaging;
using System.Windows.Shapes;

namespace HardwareStore.Windows.ComponentDetails
{
    public struct ComponentDetailsFilter
    {
        public ComponentDetailsFilter()
        {
            Id = -1;
            DetailTypeId = -1;
            ComponentId = -1;

            Value = "";

            IsFiltered = false;
        }

        public int Id { get; set; }
        public int DetailTypeId { get; set; }
        public int ComponentId { get; set; }
        public string Value { get; set; }

        public bool IsFiltered { get; set; }
    }

    /// <summary>
    /// Логика взаимодействия для FilterComponentDetailsWindow.xaml
    /// </summary>
    public partial class FilterComponentDetailsWindow : Window
    {
        public ComponentDetailsFilter Filter;

        public FilterComponentDetailsWindow()
        {
```

```
            InitializeComponent();
        }

        private void Window_Loaded(object sender, RoutedEventArgs e)
        {
            if (Filter.Id >= 1)
                idTextBox.Text = Filter.Id.ToString();

            if (Filter.DetailTypeId >= 1)
                detailTypeIdTextBox.Text = Filter.DetailTypeId.ToString();

            if (Filter.ComponentId >= 1)
                componentIdTextBox.Text = Filter.ComponentId.ToString();

            valueTextBox.Text = Filter.Value;
        }

        private void idTextBox_TextChanged(object sender, TextChangedEventArgs e)
        {
            Utilities.ValidateTextBox(RegularExpressions.GetUnsignedIntRegex(), sender,
e);
        }

        private void detailTypeIdTextBox_TextChanged(object sender,
TextChangedEventArgs e)
        {
            Utilities.ValidateTextBox(RegularExpressions.GetUnsignedIntRegex(), sender,
e);
        }

        private void componentIdTextBox_TextChanged(object sender,
TextChangedEventArgs e)
        {
            Utilities.ValidateTextBox(RegularExpressions.GetUnsignedIntRegex(), sender,
e);
        }

        private void filterButton_Click(object sender, RoutedEventArgs e)
        {
            ComponentDetailsFilter NewFilter = new()
            {
                Id = Utilities.TryParse(idTextBox.Text, -1),
                DetailTypeId = Utilities.TryParse(detailTypeIdTextBox.Text, -1),
                ComponentId = Utilities.TryParse(componentIdTextBox.Text, -1),
                Value = valueTextBox.Text,
                IsFiltered = true
            };
```

```
            Filter = NewFilter;

            DialogResult = true;
        }
    }
}
```

Листинг А.20 – Код класса «FilterComponentDetailsWindow»

```
using HardwareStore.Models;
using HardwareStore.Windows.OrderComponents;
using HardwareStore.Windows.Orders;
using HardwareStore.Windows.Suppliers;
using Microsoft.EntityFrameworkCore;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows;
using System.Windows.Controls;
using System.Windows.Data;
using System.Windows.Documents;
using System.Windows.Input;
using System.Windows.Media;
using System.Windows.Media.Imaging;
using System.Windows.Navigation;
using System.Windows.Shapes;

namespace HardwareStore.Windows.Components
{
    /// <summary>
    /// Логика взаимодействия для ComponentsUserControl.xaml
    /// </summary>
    public partial class ComponentsUserControl : UserControl
    {
        private StoreContext _context = new();

        private readonly CollectionViewSource componentsViewSource;
        private readonly CollectionViewSource componentTypesViewSource;

        private ComponentsFilter Filter { get; set; }

        public ComponentsUserControl()
        {
            InitializeComponent();
            componentsViewSource =
(CollectionViewSource)FindResource(nameof(componentsViewSource));
            componentTypesViewSource =
(CollectionViewSource)FindResource(nameof(componentTypesViewSource));
            Filter = new(_context);
        }

        private void UserControl_Loaded(object sender, RoutedEventArgs e)
        {
            _context.Components.Load();
            _context.ComponentTypes.Load();
```

```
            componentsViewSource.Source =
_context.Components.Local.ToObservableCollection();
            componentTypesViewSource.Source =
_context.ComponentTypes.Local.ToObservableCollection();

            componentsDataGrid.CanUserAddRows =
GroupPermissions.permissions[GroupPermissions.Group].Components.CanAdd;
            componentsDataGrid.CanUserDeleteRows =
GroupPermissions.permissions[GroupPermissions.Group].Components.CanDelete;
            componentsDataGrid.IsReadOnly =
!GroupPermissions.permissions[GroupPermissions.Group].Components.CanEdit;
        }

        private void updateComponents_Click(object sender, RoutedEventArgs e)
        {
            Filter = new(_context);

            _context = new StoreContext();
            _context.Components.Load();
            _context.ComponentTypes.Load();
            componentsViewSource.Source =
_context.Components.Local.ToObservableCollection();
            componentTypesViewSource.Source =
_context.ComponentTypes.Local.ToObservableCollection();
        }

        private void saveComponents_Click(object sender, RoutedEventArgs e)
        {
            Utilities.HandleSaveDataGrid(_context, componentsDataGrid);
        }

        private void filterComponents_Click(object sender, RoutedEventArgs e)
        {
            FilterComponentsWindow filterComponentsWindow = new()
            {
                Filter = Filter
            };

            if (filterComponentsWindow.ShowDialog() == true)
            {
                Filter = filterComponentsWindow.Filter;

CollectionViewSource.GetDefaultView(componentsDataGrid.ItemsSource).Refresh();
            }
        }

        private void CollectionViewSource_Filter(object sender, FilterEventArgs e)
```

```
        {
            if (e.Item is Component component)
            {
                bool IsWarrantyInRange = Filter.Warranty.Min <= component.Warranty &&
component.Warranty <= Filter.Warranty.Max;

                if (!Filter.IsFiltered)
                {
                    e.Accepted = true;
                    return;
                }

                e.Accepted = IsWarrantyInRange
                    && (Filter.Id == -1 || Filter.Id == component.Id)
                    && (Filter.Name.Length == 0 || component.Name == Filter.Name)
                    && (Filter.ComponentTypeId == -1 || Filter.ComponentTypeId ==
component.ComponentTypeId);
            }
        }

        private void reportComponents_Click(object sender, RoutedEventArgs e)
        {
            if (!_context.Components.Any())
            {
                MessageBox.Show("Для создания отчета необходима, как минимум, одна
запись!", "Ошибка");
                return;
            }

            ReportComponentsWindow reportComponentsWindow = new();
            reportComponentsWindow.ShowDialog();
        }

        private void exportComponents_Click(object sender, RoutedEventArgs e)
        {
            List<string[]> headerRow = new() { new string[] { "ИД", "Модель", "ИД
категории", "Название категории", "Гарантия (мес)" } };
            List<object[]> cellData = new() { };

            foreach (Component? component in _context.Components.ToList())
            {
                if (component == null)
                    continue;

                cellData.Add(new object[] { component.Id, component.Name,
component.ComponentTypeId, component.ComponentType.Name, component.Warranty
});
            }
```

```
            Utilities.ExportToExcel("Комплектующие", headerRow, cellData);
        }
    }
}
```

Листинг А.21 – Код класса «ComponentsUserControl»

```
using HardwareStore.Windows.OrderComponents;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows;
using System.Windows.Controls;
using System.Windows.Data;
using System.Windows.Documents;
using System.Windows.Input;
using System.Windows.Media;
using System.Windows.Media.Imaging;
using System.Windows.Shapes;
using HardwareStore.Models;

namespace HardwareStore.Windows.Components
{
    public struct ComponentsFilter
    {
        public ComponentsFilter(StoreContext context)
        {
            Id = -1;
            ComponentTypeId = -1;

            Name = "";

            Component? maxWarranty = context.Components.OrderByDescending(c =>
c.Warranty).FirstOrDefault();

            Warranty = (0, (maxWarranty != null) ? maxWarranty.Warranty : 0);

            IsFiltered = false;
        }

        public int Id { get; set; }
        public string Name { get; set; }
        public int ComponentTypeId { get; set; }

        public (int Min, int Max) Warranty { get; set; }

        public bool IsFiltered { get; set; }
    }

    /// <summary>
    /// Логика взаимодействия для FilterComponentsWindow.xaml
    /// </summary>
    public partial class FilterComponentsWindow : Window
```

```
    {
        public ComponentsFilter Filter;

        public FilterComponentsWindow()
        {
            InitializeComponent();
        }

        private void Window_Loaded(object sender, RoutedEventArgs e)
        {
            if (Filter.Id >= 1)
                idTextBox.Text = Filter.Id.ToString();

            minWarrantyTextBox.Text = Filter.Warranty.Min.ToString();
            maxWarrantyTextBox.Text = Filter.Warranty.Max.ToString();

            nameTextBox.Text = Filter.Name;

            StoreContext context = new();

            foreach (ComponentType componentType in context.ComponentTypes.ToList())
            {
                ComboBoxItem componentTypeComboBoxItem = new()
                {
                    Tag = componentType.Id.ToString(),
                    Content = componentType.Name
                };

                componentTypeComboBox.Items.Add(componentTypeComboBoxItem);
            }

            for (int i = 0; i < componentTypeComboBox.Items.Count; i++)
            {
                if (componentTypeComboBox.Items[i] is ComboBoxItem comboBoxItem &&
comboBoxItem.Tag.ToString() == Filter.ComponentTypeId.ToString())
                {
                    componentTypeComboBox.SelectedIndex = i;
                    break;
                }
            }
        }

        private void idTextBox_TextChanged(object sender, TextChangedEventArgs e)
        {
            Utilities.ValidateTextBox(RegularExpressions.GetUnsignedIntRegex(), sender,
e);
        }
```

```
        private void nameTextBox_TextChanged(object sender, TextChangedEventArgs e)
        {
            //Utilities.ValidateTextBox(RegularExpressions.GetUnsignedIntRegex(), sender,
e);
        }

        private void minWarrantyTextBox_TextChanged(object sender,
TextChangedEventArgs e)
        {
            Utilities.ValidateTextBox(RegularExpressions.GetUnsignedIntRegex(), sender,
e);
        }

        private void maxWarrantyTextBox_TextChanged(object sender,
TextChangedEventArgs e)
        {
            Utilities.ValidateTextBox(RegularExpressions.GetUnsignedIntRegex(), sender,
e);
        }

        private void filterButton_Click(object sender, RoutedEventArgs e)
        {
            ComponentsFilter NewFilter = new()
            {
                Warranty = (Utilities.TryParse(minWarrantyTextBox.Text,
Filter.Warranty.Min), Utilities.TryParse(maxWarrantyTextBox.Text,
Filter.Warranty.Max)),
                Id = Utilities.TryParse(idTextBox.Text, -1),
                ComponentTypeId =
int.Parse((string)((ComboBoxItem)componentTypeComboBox.SelectedItem).Tag),
                Name = nameTextBox.Text,
                IsFiltered = true
            };

            Filter = NewFilter;

            DialogResult = true;
        }
    }
}
```

Листинг А.22 – Код класса «FilterComponentsWindow»

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows;
using System.Windows.Controls;
using System.Windows.Data;
using System.Windows.Documents;
using System.Windows.Input;
using System.Windows.Media;
using System.Windows.Media.Imaging;
using System.Windows.Shapes;
using OxyPlot;
using OxyPlot.Axes;
using OxyPlot.Series;
using OxyPlot.Wpf;
using HardwareStore.Models;

namespace HardwareStore.Windows.Components
{
   public class ComponentsPlotModel
   {
      private StoreContext _context = new();

      public PlotModel Model { get; private set; }

      public ComponentsPlotModel()
      {
         Model = new() { Title = "Доля категорий комплектующих" };

         dynamic series = new PieSeries() { StrokeThickness = 2.0, InsideLabelPosition =
0.8, AngleSpan = 360, StartAngle = 0 };

         int totalEntries = _context.Components.Count();

         var categories = _context.Components.GroupBy(c =>
c.ComponentTypeId).Select(c => new { categoryTypeId = c.Key, count = c.Count()
}).ToList();

         foreach (var category in categories)
         {
            ComponentType? componentType =
_context.ComponentTypes.Find(category.categoryTypeId);

            if (componentType != null)
            {
               series.Slices.Add(new PieSlice(componentType.Name, category.count)
```

```
                {
                    IsExploded = false
                });
            }
        }

        Model.Series.Add(series);
    }
}

/// <summary>
/// Логика взаимодействия для ReportComponentsWindow.xaml
/// </summary>
public partial class ReportComponentsWindow : Window
{
    public ReportComponentsWindow()
    {
        InitializeComponent();
    }
}
}
```

Листинг А.23 – Код класса «ReportComponentsWindow»

```
using Microsoft.EntityFrameworkCore;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows;
using System.Windows.Controls;
using System.Windows.Data;
using System.Windows.Documents;
using System.Windows.Input;
using System.Windows.Media;
using System.Windows.Media.Imaging;
using System.Windows.Navigation;
using System.Windows.Shapes;
using HardwareStore.Models;
using HardwareStore.Windows.Components;
using System.ComponentModel;

namespace HardwareStore.Windows.ComponentStorages
{
    /// <summary>
    /// Логика взаимодействия для ComponentStoragesUserControl.xaml
    /// </summary>
    public partial class ComponentStoragesUserControl : UserControl
    {
        private StoreContext _context = new();

        private readonly CollectionViewSource componentStoragesViewSource;

        private ComponentStoragesFilter Filter { get; set; }

        public ComponentStoragesUserControl()
        {
            InitializeComponent();
            componentStoragesViewSource =
(CollectionViewSource)FindResource(nameof(componentStoragesViewSource));
            Filter = new(_context);
        }

        private void UserControl_Loaded(object sender, RoutedEventArgs e)
        {
            _context.ComponentStorages.Load();
            componentStoragesViewSource.Source =
_context.ComponentStorages.Local.ToObservableCollection();

            componentStoragesDataGrid.CanUserAddRows =
GroupPermissions.permissions[GroupPermissions.Group].ComponentStorages.CanAdd;
```

```
            componentStoragesDataGrid.CanUserDeleteRows =
GroupPermissions.permissions[GroupPermissions.Group].ComponentStorages.CanDelet
e;
            componentStoragesDataGrid.IsReadOnly =
!GroupPermissions.permissions[GroupPermissions.Group].ComponentStorages.CanEdit;
        }

        private void updateComponentStorages_Click(object sender, RoutedEventArgs e)
        {
            Filter = new(_context);

            _context = new StoreContext();
            _context.ComponentStorages.Load();
            componentStoragesViewSource.Source =
_context.ComponentStorages.Local.ToObservableCollection();
        }

        private void saveComponentStorages_Click(object sender, RoutedEventArgs e)
        {
            Utilities.HandleSaveDataGrid(_context, componentStoragesDataGrid);
        }

        private void filterComponentStorages_Click(object sender, RoutedEventArgs e)
        {
            FilterComponentStoragesWindow filterComponentStoragesWindow = new()
            {
                Filter = Filter
            };

            if (filterComponentStoragesWindow.ShowDialog() == true)
            {
                Filter = filterComponentStoragesWindow.Filter;

CollectionViewSource.GetDefaultView(componentStoragesDataGrid.ItemsSource).Refre
sh();
            }
        }

        private void CollectionViewSource_Filter(object sender, FilterEventArgs e)
        {
            if (e.Item is ComponentStorage componentStorage)
            {
                bool IsAmountInRange = Filter.Amount.Min <= componentStorage.Amount
&& componentStorage.Amount <= Filter.Amount.Max,
                    IsPriceInRange = Filter.Price.Min <= componentStorage.Price &&
componentStorage.Price <= Filter.Price.Max;

                if (!Filter.IsFiltered)
```

```
            {
                e.Accepted = true;
                return;
            }

            e.Accepted = IsAmountInRange && IsPriceInRange
                && (Filter.Id == -1 || Filter.Id == componentStorage.Id)
                && (Filter.ComponentId == -1 || Filter.ComponentId ==
componentStorage.ComponentId)
                && (Filter.WarehouseId == -1 || Filter.WarehouseId ==
componentStorage.WarehouseId);
        }
    }

    private void exportComponentStorages_Click(object sender, RoutedEventArgs e)
    {
        List<string[]> headerRow = new() { new string[] { "ИД", "ИД склада", "ИД
комплектующего", "Цена (руб)", "Количество (шт)" } };
        List<object[]> cellData = new() { };

        foreach (ComponentStorage? componentStorage in
_context.ComponentStorages.ToList())
        {
            if (componentStorage == null)
                continue;

            cellData.Add(new object[] { componentStorage.Id,
componentStorage.WarehouseId, componentStorage.ComponentId,
componentStorage.Price, componentStorage.Amount });
        }

        Utilities.ExportToExcel("Учет комплектующих", headerRow, cellData);
    }
  }
}
```

Листинг А.24 – Код класса «ComponentStoragesUserControl»

```
using HardwareStore.Models;
using HardwareStore.Windows.Components;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows;
using System.Windows.Controls;
using System.Windows.Data;
using System.Windows.Documents;
using System.Windows.Input;
using System.Windows.Media;
using System.Windows.Media.Imaging;
using System.Windows.Shapes;

namespace HardwareStore.Windows.ComponentStorages
{
    public struct ComponentStoragesFilter
    {
        public ComponentStoragesFilter(StoreContext context)
        {
            Id = -1;
            WarehouseId = -1;
            ComponentId = -1;

            ComponentStorage? maxPrice =
context.ComponentStorages.OrderByDescending(cs => cs.Price).FirstOrDefault();
            ComponentStorage? maxAmount =
context.ComponentStorages.OrderByDescending(cs => cs.Amount).FirstOrDefault();

            Amount = (0, (maxAmount != null) ? maxAmount.Amount : 0);
            Price = (0, (maxPrice != null) ? maxPrice.Price : 0);

            IsFiltered = false;
        }

        public int Id { get; set; }
        public int WarehouseId { get; set; }
        public int ComponentId { get; set; }

        public (float Min, float Max) Price { get; set; }
        public (int Min, int Max) Amount { get; set; }

        public bool IsFiltered { get; set; }
    }
```

```
    /// <summary>
    /// Логика взаимодействия для FilterComponentStoragesWindow.xaml
    /// </summary>
    public partial class FilterComponentStoragesWindow : Window
    {
        public ComponentStoragesFilter Filter;

        public FilterComponentStoragesWindow()
        {
            InitializeComponent();
        }

        private void Window_Loaded(object sender, RoutedEventArgs e)
        {
            if (Filter.Id >= 1)
                idTextBox.Text = Filter.Id.ToString();

            if (Filter.WarehouseId >= 1)
                warehouseIdTextBox.Text = Filter.WarehouseId.ToString();

            if (Filter.ComponentId >= 1)
                componentIdTextBox.Text = Filter.ComponentId.ToString();

            minPriceTextBox.Text = Filter.Price.Min.ToString();
            maxPriceTextBox.Text = Filter.Price.Max.ToString();

            minAmountTextBox.Text = Filter.Amount.Min.ToString();
            maxAmountTextBox.Text = Filter.Amount.Max.ToString();
        }

        private void idTextBox_TextChanged(object sender, TextChangedEventArgs e)
        {
            Utilities.ValidateTextBox(RegularExpressions.GetUnsignedIntRegex(), sender,
e);
        }

        private void warehouseIdTextBox_TextChanged(object sender,
TextChangedEventArgs e)
        {
            Utilities.ValidateTextBox(RegularExpressions.GetUnsignedIntRegex(), sender,
e);
        }

        private void componentIdTextBox_TextChanged(object sender,
TextChangedEventArgs e)
        {
            Utilities.ValidateTextBox(RegularExpressions.GetUnsignedIntRegex(), sender,
e);
```

```
        }

        private void minPriceTextBox_TextChanged(object sender, TextChangedEventArgs
e)
        {
            Utilities.ValidateTextBox(RegularExpressions.GetUnsignedFloatRegex(), sender,
e);
        }

        private void maxPriceTextBox_TextChanged(object sender,
TextChangedEventArgs e)
        {
            Utilities.ValidateTextBox(RegularExpressions.GetUnsignedFloatRegex(), sender,
e);
        }

        private void minAmountTextBox_TextChanged(object sender,
TextChangedEventArgs e)
        {
            Utilities.ValidateTextBox(RegularExpressions.GetUnsignedIntRegex(), sender,
e);
        }

        private void maxAmountTextBox_TextChanged(object sender,
TextChangedEventArgs e)
        {
            Utilities.ValidateTextBox(RegularExpressions.GetUnsignedIntRegex(), sender,
e);
        }

        private void filterButton_Click(object sender, RoutedEventArgs e)
        {
            ComponentStoragesFilter NewFilter = new()
            {
                Id = Utilities.TryParse(idTextBox.Text, -1),
                WarehouseId = Utilities.TryParse(warehouseIdTextBox.Text, -1),
                ComponentId = Utilities.TryParse(componentIdTextBox.Text, -1),
                Amount = (Utilities.TryParse(minAmountTextBox.Text, Filter.Amount.Min),
Utilities.TryParse(maxAmountTextBox.Text, Filter.Amount.Max)),
                Price = (Utilities.TryParse(minPriceTextBox.Text, Filter.Price.Min),
Utilities.TryParse(maxPriceTextBox.Text, Filter.Price.Max)),
                IsFiltered = true
            };

            Filter = NewFilter;

            DialogResult = true;
        }
```

```
    }
}
```

Листинг А.25 – Код класса «FilterComponentStoragesWindow»

```
using HardwareStore.Models;
using HardwareStore.Windows.Suppliers;
using HardwareStore.Windows.Warehouses;
using Microsoft.EntityFrameworkCore;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows;
using System.Windows.Controls;
using System.Windows.Data;
using System.Windows.Documents;
using System.Windows.Input;
using System.Windows.Media;
using System.Windows.Media.Imaging;
using System.Windows.Navigation;
using System.Windows.Shapes;

namespace HardwareStore.Windows.ComponentTypes
{
   /// <summary>
   /// Логика взаимодействия для ComponentTypesUserControl.xaml
   /// </summary>
   public partial class ComponentTypesUserControl : UserControl
   {
      private StoreContext _context = new();

      private readonly CollectionViewSource componentTypesViewSource;

      public ComponentTypesUserControl()
      {
         InitializeComponent();
         componentTypesViewSource =
(CollectionViewSource)FindResource(nameof(componentTypesViewSource));
      }

      private void UserControl_Loaded(object sender, RoutedEventArgs e)
      {
         _context.ComponentTypes.Load();
         componentTypesViewSource.Source =
_context.ComponentTypes.Local.ToObservableCollection();

         componentTypesDataGrid.CanUserAddRows =
GroupPermissions.permissions[GroupPermissions.Group].ComponentTypes.CanAdd;
         componentTypesDataGrid.CanUserDeleteRows =
GroupPermissions.permissions[GroupPermissions.Group].ComponentTypes.CanDelete;
```

```
            componentTypesDataGrid.IsReadOnly =
!GroupPermissions.permissions[GroupPermissions.Group].ComponentTypes.CanEdit;
        }

        private void updateComponentTypes_Click(object sender, RoutedEventArgs e)
        {
            _context = new StoreContext();
            _context.ComponentTypes.Load();
            componentTypesViewSource.Source =
_context.ComponentTypes.Local.ToObservableCollection();
        }

        private void saveComponentTypes_Click(object sender, RoutedEventArgs e)
        {
            Utilities.HandleSaveDataGrid(_context, componentTypesDataGrid);
        }

        private void exportComponentTypes_Click(object sender, RoutedEventArgs e)
        {
            List<string[]> headerRow = new() { new string[] { "ИД", "Название" } };
            List<object[]> cellData = new() { };

            foreach (ComponentType? componentType in
_context.ComponentTypes.ToList())
            {
                if (componentType == null)
                    continue;

                cellData.Add(new object[] { componentType.Id, componentType.Name });
            }

            Utilities.ExportToExcel("Категории комплектующих", headerRow, cellData);
        }
    }
}
```

Листинг А.26 – Код класса «ComponentTypesUserControl»

```
using HardwareStore.Models;
using HardwareStore.Windows.Supplies;
using Microsoft.EntityFrameworkCore;
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows;
using System.Windows.Controls;
using System.Windows.Data;
using System.Windows.Documents;
using System.Windows.Input;
using System.Windows.Media;
using System.Windows.Media.Imaging;
using System.Windows.Navigation;
using System.Windows.Shapes;

namespace HardwareStore.Windows.Configurations
{
    /// <summary>
    /// Логика взаимодействия для ConfigurationsUserControl.xaml
    /// </summary>
    public partial class ConfigurationsUserControl : UserControl
    {
        private StoreContext _context = new();

        private readonly CollectionViewSource configurationsViewSource;

        private ConfigurationsFilter Filter { get; set; }

        public ConfigurationsUserControl()
        {
            InitializeComponent();
            configurationsViewSource =
(CollectionViewSource)FindResource(nameof(configurationsViewSource));
            Filter = new(_context);
        }

        private void UserControl_Loaded(object sender, RoutedEventArgs e)
        {
            _context.Configurations.Load();

            configurationsViewSource.GroupDescriptions.Clear();
            configurationsViewSource.GroupDescriptions.Add(new
PropertyGroupDescription("ConfigurationId"));
```

```
            configurationsViewSource.Source =
_context.Configurations.Local.ToObservableCollection();

            configurationsDataGrid.CanUserAddRows =
GroupPermissions.permissions[GroupPermissions.Group].Configurations.CanAdd;
            configurationsDataGrid.CanUserDeleteRows =
GroupPermissions.permissions[GroupPermissions.Group].Configurations.CanDelete;
            configurationsDataGrid.IsReadOnly =
!GroupPermissions.permissions[GroupPermissions.Group].Configurations.CanEdit;
        }

        private void updateConfigurations_Click(object sender, RoutedEventArgs e)
        {
            Filter = new(_context);

            _context = new StoreContext();
            _context.Configurations.Load();

            configurationsViewSource.GroupDescriptions.Clear();
            configurationsViewSource.GroupDescriptions.Add(new
PropertyGroupDescription("ConfigurationId"));
            configurationsViewSource.Source =
_context.Configurations.Local.ToObservableCollection();
        }

        private void saveConfigurations_Click(object sender, RoutedEventArgs e)
        {
            Utilities.HandleSaveDataGrid(_context, configurationsDataGrid);
        }

        private void filterConfigurations_Click(object sender, RoutedEventArgs e)
        {
            FilterConfigurationsWindow filterConfigurationsWindow = new()
            {
                Filter = Filter
            };

            if (filterConfigurationsWindow.ShowDialog() == true)
            {
                Filter = filterConfigurationsWindow.Filter;

CollectionViewSource.GetDefaultView(configurationsDataGrid.ItemsSource).Refresh();
            }
        }

        private void CollectionViewSource_Filter(object sender, FilterEventArgs e)
        {
            if (e.Item is Configuration configuration)
```

```
            {
                bool IsAmountInRange = Filter.MinAmount <= configuration.Amount &&
configuration.Amount <= Filter.MaxAmount;

                if (!Filter.IsFiltered)
                {
                    e.Accepted = true;
                    return;
                }

                e.Accepted = IsAmountInRange
                    && (Filter.Id == -1 || Filter.Id == configuration.Id)
                    && (Filter.ComponentId == -1 || Filter.ComponentId ==
configuration.ComponentId)
                    && (Filter.ConfigurationId == -1 || Filter.ConfigurationId ==
configuration.ConfigurationId);
            }
        }

        private void exportConfigurations_Click(object sender, RoutedEventArgs e)
        {
            List<string[]> headerRow = new() { new string[] { "ИД", "ИД конфигурации",
"ИД комплектующего", "Количество" } };
            List<object[]> cellData = new() { };

            foreach (Configuration? configuration in _context.Configurations.ToList())
            {
                if (configuration == null)
                    continue;

                cellData.Add(new object[] { configuration.Id, configuration.ConfigurationId,
configuration.ComponentId, configuration.Amount });
            }

            Utilities.ExportToExcel("Конфигурации", headerRow, cellData);
        }
    }
}
```

Листинг А.27 – Код класса «ConfigurationsUserControl»

```
using HardwareStore.Windows.Supplies;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows;
using System.Windows.Controls;
using System.Windows.Data;
using System.Windows.Documents;
using System.Windows.Input;
using System.Windows.Media;
using System.Windows.Media.Imaging;
using System.Windows.Shapes;
using HardwareStore.Models;

namespace HardwareStore.Windows.Configurations
{
    public struct ConfigurationsFilter
    {
        public ConfigurationsFilter(StoreContext context)
        {
            Id = -1;
            ConfigurationId = -1;
            ComponentId = -1;

            Configuration? maxAmount = context.Configurations.OrderByDescending(c =>
c.Amount).FirstOrDefault();

            MinAmount = 0;
            MaxAmount = (maxAmount != null) ? maxAmount.Amount : 0;

            IsFiltered = false;
        }

        public int Id { get; set; }
        public int ConfigurationId { get; set; }
        public int ComponentId { get; set; }
        public int MinAmount { get; set; }
        public int MaxAmount { get; set; }

        public bool IsFiltered { get; set; }
    }

    /// <summary>
    /// Логика взаимодействия для FilterConfigurationsWindow.xaml
    /// </summary>
    public partial class FilterConfigurationsWindow : Window
```

```
    {
        public ConfigurationsFilter Filter;

        public FilterConfigurationsWindow()
        {
            InitializeComponent();
        }

        private void Window_Loaded(object sender, RoutedEventArgs e)
        {
            if (Filter.Id >= 1)
                idTextBox.Text = Filter.Id.ToString();

            if (Filter.ComponentId >= 1)
                componentIdTextBox.Text = Filter.ComponentId.ToString();

            if (Filter.ConfigurationId >= 1)
                configurationIdTextBox.Text = Filter.ConfigurationId.ToString();

            minAmountTextBox.Text = Filter.MinAmount.ToString();
            maxAmountTextBox.Text = Filter.MaxAmount.ToString();
        }

        private void idTextBox_TextChanged(object sender, TextChangedEventArgs e)
        {
            Utilities.ValidateTextBox(RegularExpressions.GetUnsignedIntRegex(), sender,
e);
        }

        private void configurationIdTextBox_TextChanged(object sender,
TextChangedEventArgs e)
        {
            Utilities.ValidateTextBox(RegularExpressions.GetUnsignedIntRegex(), sender,
e);
        }

        private void componentIdTextBox_TextChanged(object sender,
TextChangedEventArgs e)
        {
            Utilities.ValidateTextBox(RegularExpressions.GetUnsignedIntRegex(), sender,
e);
        }

        private void minAmountTextBox_TextChanged(object sender,
TextChangedEventArgs e)
        {
            Utilities.ValidateTextBox(RegularExpressions.GetUnsignedIntRegex(), sender,
e);
```

```
        }

        private void maxAmountTextBox_TextChanged(object sender,
TextChangedEventArgs e)
        {
            Utilities.ValidateTextBox(RegularExpressions.GetUnsignedIntRegex(), sender,
e);
        }

        private void filterButton_Click(object sender, RoutedEventArgs e)
        {
            ConfigurationsFilter NewFilter = new()
            {
                MinAmount = Utilities.TryParse(minAmountTextBox.Text,
Filter.MinAmount),
                MaxAmount = Utilities.TryParse(maxAmountTextBox.Text,
Filter.MaxAmount),
                Id = Utilities.TryParse(idTextBox.Text, -1),
                ConfigurationId = Utilities.TryParse(configurationIdTextBox.Text, -1),
                ComponentId = Utilities.TryParse(componentIdTextBox.Text, -1),
                IsFiltered = true
            };

            Filter = NewFilter;

            DialogResult = true;
        }
    }
}
```

Листинг А.28 – Код класса «FilterConfigurationsWindow»

```
using HardwareStore.Models;
using Microsoft.EntityFrameworkCore;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows;
using System.Windows.Controls;
using System.Windows.Data;
using System.Windows.Documents;
using System.Windows.Input;
using System.Windows.Media;
using System.Windows.Media.Imaging;
using System.Windows.Navigation;
using System.Windows.Shapes;

namespace HardwareStore.Windows.DetailTypes
{
    /// <summary>
    /// Логика взаимодействия для DetailTypesUserControl.xaml
    /// </summary>
    public partial class DetailTypesUserControl : UserControl
    {
        private StoreContext _context = new();

        private readonly CollectionViewSource detailTypesViewSource;

        public DetailTypesUserControl()
        {
            InitializeComponent();
            detailTypesViewSource =
(CollectionViewSource)FindResource(nameof(detailTypesViewSource));
        }

        private void UserControl_Loaded(object sender, RoutedEventArgs e)
        {
            _context.DetailTypes.Load();
            detailTypesViewSource.Source =
_context.DetailTypes.Local.ToObservableCollection();

            detailTypesDataGrid.CanUserAddRows =
GroupPermissions.permissions[GroupPermissions.Group].DetailTypes.CanAdd;
            detailTypesDataGrid.CanUserDeleteRows =
GroupPermissions.permissions[GroupPermissions.Group].DetailTypes.CanDelete;
            detailTypesDataGrid.IsReadOnly =
!GroupPermissions.permissions[GroupPermissions.Group].DetailTypes.CanEdit;
        }
```

```
        private void updateDetailTypes_Click(object sender, RoutedEventArgs e)
        {
            _context = new StoreContext();
            _context.DetailTypes.Load();
            detailTypesViewSource.Source =
_context.DetailTypes.Local.ToObservableCollection();
        }

        private void saveDetailTypes_Click(object sender, RoutedEventArgs e)
        {
            Utilities.HandleSaveDataGrid(_context, detailTypesDataGrid);
        }

        private void exportDetailTypes_Click(object sender, RoutedEventArgs e)
        {
            List<string[]> headerRow = new() { new string[] { "ИД", "Название" } };
            List<object[]> cellData = new() { };

            foreach (DetailType? detailType in _context.DetailTypes.ToList())
            {
                if (detailType == null)
                    continue;

                cellData.Add(new object[] { detailType.Id, detailType.Name });
            }

            Utilities.ExportToExcel("Типы характеристик комплектующих", headerRow,
cellData);
        }
    }
}
```

Листинг А.29 – Код класса «DetailTypesUserControl»

```
using HardwareStore.Models;
using HardwareStore.Windows.Supplies;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows;
using System.Windows.Controls;
using System.Windows.Data;
using System.Windows.Documents;
using System.Windows.Input;
using System.Windows.Media;
using System.Windows.Media.Imaging;
using System.Windows.Shapes;

namespace HardwareStore.Windows.OrderComponents
{
    public struct OrderComponentsFilter
    {
        public OrderComponentsFilter(StoreContext context)
        {
            Id = -1;
            OrderId = -1;
            ComponentId = -1;
            WarehouseId = -1;

            OrderComponent? maxPrice = context.OrderComponents.OrderByDescending(s
=> s.Price).FirstOrDefault();
            OrderComponent? maxAmount =
context.OrderComponents.OrderByDescending(s => s.Amount).FirstOrDefault();

            Price = (0, (maxPrice != null) ? maxPrice.Price : 0);
            Amount = (0, (maxAmount != null) ? maxAmount.Amount : 0);

            IsPartOfConfiguration = false;

            IsFiltered = false;
        }

        public int Id { get; set; }
        public int OrderId { get; set; }
        public int ComponentId { get; set; }
        public int WarehouseId { get; set; }

        public (float Min, float Max) Price { get; set; }
        public (int Min, int Max) Amount { get; set; }
```

```
        public bool IsPartOfConfiguration { get; set; }

        public bool IsFiltered { get; set; }
    }

    /// <summary>
    /// Логика взаимодействия для FilterOrderComponentsWindow.xaml
    /// </summary>
    public partial class FilterOrderComponentsWindow : Window
    {
        public OrderComponentsFilter Filter;

        public FilterOrderComponentsWindow()
        {
            InitializeComponent();
        }

        private void Window_Loaded(object sender, RoutedEventArgs e)
        {
            if (Filter.Id >= 1)
                idTextBox.Text = Filter.Id.ToString();

            if (Filter.ComponentId >= 1)
                componentIdTextBox.Text = Filter.ComponentId.ToString();

            if (Filter.WarehouseId >= 1)
                warehouseIdTextBox.Text = Filter.WarehouseId.ToString();

            if (Filter.OrderId >= 1)
                orderIdTextBox.Text = Filter.OrderId.ToString();

            minPriceTextBox.Text = Filter.Price.Min.ToString();
            maxPriceTextBox.Text = Filter.Price.Max.ToString();

            minAmountTextBox.Text = Filter.Amount.Min.ToString();
            maxAmountTextBox.Text = Filter.Amount.Max.ToString();
        }

        private void idTextBox_TextChanged(object sender, TextChangedEventArgs e)
        {
            Utilities.ValidateTextBox(RegularExpressions.GetUnsignedIntRegex(), sender,
e);
        }

        private void orderIdTextBox_TextChanged(object sender, TextChangedEventArgs
e)
        {
```

```
            Utilities.ValidateTextBox(RegularExpressions.GetUnsignedIntRegex(), sender,
e);
        }

        private void componentIdTextBox_TextChanged(object sender,
TextChangedEventArgs e)
        {
            Utilities.ValidateTextBox(RegularExpressions.GetUnsignedIntRegex(), sender,
e);
        }

        private void warehouseIdTextBox_TextChanged(object sender,
TextChangedEventArgs e)
        {
            Utilities.ValidateTextBox(RegularExpressions.GetUnsignedIntRegex(), sender,
e);
        }

        private void minPriceTextBox_TextChanged(object sender, TextChangedEventArgs
e)
        {
            Utilities.ValidateTextBox(RegularExpressions.GetUnsignedFloatRegex(), sender,
e);
        }

        private void maxPriceTextBox_TextChanged(object sender,
TextChangedEventArgs e)
        {
            Utilities.ValidateTextBox(RegularExpressions.GetUnsignedFloatRegex(), sender,
e);
        }

        private void minAmountTextBox_TextChanged(object sender,
TextChangedEventArgs e)
        {
            Utilities.ValidateTextBox(RegularExpressions.GetUnsignedIntRegex(), sender,
e);
        }

        private void maxAmountTextBox_TextChanged(object sender,
TextChangedEventArgs e)
        {
            Utilities.ValidateTextBox(RegularExpressions.GetUnsignedIntRegex(), sender,
e);
        }

        private void filterButton_Click(object sender, RoutedEventArgs e)
        {
```

```
            OrderComponentsFilter NewFilter = new()
            {
                Price = (Utilities.TryParse(minPriceTextBox.Text, Filter.Price.Min),
Utilities.TryParse(maxPriceTextBox.Text, Filter.Price.Max)),
                Amount = (Utilities.TryParse(minAmountTextBox.Text, Filter.Amount.Min),
Utilities.TryParse(maxAmountTextBox.Text, Filter.Amount.Max)),
                Id = Utilities.TryParse(idTextBox.Text, -1),
                WarehouseId = Utilities.TryParse(warehouseIdTextBox.Text, -1),
                ComponentId = Utilities.TryParse(componentIdTextBox.Text, -1),
                OrderId = Utilities.TryParse(orderIdTextBox.Text, -1),
                IsFiltered = true
            };

            Filter = NewFilter;

            DialogResult = true;
        }
    }
}
```

Листинг А.30 – Код класса «FilterOrderComponentsWindow»

```
using HardwareStore.Models;
using HardwareStore.Windows.Supplies;
using Microsoft.EntityFrameworkCore;
using System;
using System.Collections.Generic;
using System.Configuration;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows;
using System.Windows.Controls;
using System.Windows.Data;
using System.Windows.Documents;
using System.Windows.Input;
using System.Windows.Media;
using System.Windows.Media.Imaging;
using System.Windows.Navigation;
using System.Windows.Shapes;

namespace HardwareStore.Windows.OrderComponents
{
    /// <summary>
    /// Логика взаимодействия для OrderComponentsUserControl.xaml
    /// </summary>
    public partial class OrderComponentsUserControl : UserControl
    {
        private StoreContext _context = new();

        private readonly CollectionViewSource orderComponentsViewSource;

        private OrderComponentsFilter Filter { get; set; }

        public OrderComponentsUserControl()
        {
            InitializeComponent();
            orderComponentsViewSource =
(CollectionViewSource)FindResource(nameof(orderComponentsViewSource));
            Filter = new(_context);
        }

        private void UserControl_Loaded(object sender, RoutedEventArgs e)
        {
            _context.OrderComponents.Load();

            orderComponentsViewSource.GroupDescriptions.Clear();
            orderComponentsViewSource.GroupDescriptions.Add(new
PropertyGroupDescription("OrderId"));
```

```
            orderComponentsViewSource.Source =
_context.OrderComponents.Local.ToObservableCollection();

            orderComponentsDataGrid.CanUserAddRows =
GroupPermissions.permissions[GroupPermissions.Group].OrderComponents.CanAdd;
            orderComponentsDataGrid.CanUserDeleteRows =
GroupPermissions.permissions[GroupPermissions.Group].OrderComponents.CanDelete;
            orderComponentsDataGrid.IsReadOnly =
!GroupPermissions.permissions[GroupPermissions.Group].OrderComponents.CanEdit;
        }

        private void updateOrderComponents_Click(object sender, RoutedEventArgs e)
        {
            Filter = new(_context);

            _context = new StoreContext();
            _context.OrderComponents.Load();

            orderComponentsViewSource.GroupDescriptions.Clear();
            orderComponentsViewSource.GroupDescriptions.Add(new
PropertyGroupDescription("OrderId"));
            orderComponentsViewSource.Source =
_context.OrderComponents.Local.ToObservableCollection();
        }

        private void saveOrderComponents_Click(object sender, RoutedEventArgs e)
        {
            Utilities.HandleSaveDataGrid(_context, orderComponentsDataGrid);
        }

        private void filterOrderComponents_Click(object sender, RoutedEventArgs e)
        {
            FilterOrderComponentsWindow filterSuppliesWindow = new()
            {
                Filter = Filter
            };

            if (filterSuppliesWindow.ShowDialog() == true)
            {
                Filter = filterSuppliesWindow.Filter;

CollectionViewSource.GetDefaultView(orderComponentsDataGrid.ItemsSource).Refres
h();
            }
        }

        private void CollectionViewSource_Filter(object sender, FilterEventArgs e)
        {
```

```
            if (e.Item is OrderComponent orderComponent)
            {
                bool IsPriceInRange = Filter.Price.Min <= orderComponent.Price &&
orderComponent.Price <= Filter.Price.Max,
                    IsAmountInRange = Filter.Amount.Min <= orderComponent.Amount &&
orderComponent.Amount <= Filter.Amount.Max;

                if (!Filter.IsFiltered)
                {
                    e.Accepted = true;
                    return;
                }

                e.Accepted = IsPriceInRange && IsAmountInRange
                    && (Filter.Id == -1 || Filter.Id == orderComponent.Id)
                    && (Filter.ComponentId == -1 || Filter.ComponentId ==
orderComponent.ComponentId)
                    && (Filter.WarehouseId == -1 || Filter.WarehouseId ==
orderComponent.WarehouseId)
                    && (Filter.OrderId == -1 || Filter.OrderId == orderComponent.OrderId);
            }
        }

        private void exportOrderComponents_Click(object sender, RoutedEventArgs e)
        {
            List<string[]> headerRow = new() { new string[] { "ИД", "ИД заказа", "ИД
склада", "ИД комплектующего", "Цена (руб)", "Количество (шт)", "Часть
конфигурации" } };
            List<object[]> cellData = new() { };

            foreach (OrderComponent? orderComponent in
_context.OrderComponents.ToList())
            {
                if (orderComponent == null)
                    continue;

                cellData.Add(new object[] { orderComponent.Id, orderComponent.OrderId,
orderComponent.WarehouseId, orderComponent.ComponentId, orderComponent.Price,
orderComponent.Amount, orderComponent.IsPartOfConfiguration });
            }

            Utilities.ExportToExcel("Содержимое заказов", headerRow, cellData);
        }
    }
}
```

Листинг А.31 – Код класса «OrderComponentsUserControl»

```
using HardwareStore.Models;
using HardwareStore.Windows.ComponentStorages;
using HardwareStore.Windows.Supplies;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows;
using System.Windows.Controls;
using System.Windows.Data;
using System.Windows.Documents;
using System.Windows.Input;
using System.Windows.Media;
using System.Windows.Media.Imaging;
using System.Windows.Shapes;

namespace HardwareStore.Windows.Orders
{
    public struct OrdersFilter
    {
        public OrdersFilter()
        {
            Id = -1;
            Status = -1;
            Date = -1;

            IsFiltered = false;
        }

        public int Id { get; set; }
        public int Status { get; set; }
        public long Date { get; set; }

        public bool IsFiltered { get; set; }
    }

    /// <summary>
    /// Логика взаимодействия для FilterOrdersWindow.xaml
    /// </summary>
    public partial class FilterOrdersWindow : Window
    {
        public OrdersFilter Filter;

        public FilterOrdersWindow()
        {
            InitializeComponent();
        }
```

```
        private void Window_Loaded(object sender, RoutedEventArgs e)
        {
            if (Filter.Id >= 1)
                idTextBox.Text = Filter.Id.ToString();

            if (Filter.Date >= 0)
                datePicker.SelectedDate = Utilities.UnixTimeStampToDateTime(Filter.Date);

            for (int i = 0; i < statusComboBox.Items.Count; i++)
            {
                ComboBoxItem? comboBoxItem = statusComboBox.Items[i] as
ComboBoxItem;

                if (comboBoxItem != null && comboBoxItem.Tag.ToString() ==
Filter.Status.ToString())
                {
                    statusComboBox.SelectedIndex = i;
                    break;
                }
            }
        }

        private void idTextBox_TextChanged(object sender, TextChangedEventArgs e)
        {
            Utilities.ValidateTextBox(RegularExpressions.GetUnsignedIntRegex(), sender,
e);
        }

        private void filterButton_Click(object sender, RoutedEventArgs e)
        {
            OrdersFilter NewFilter = new()
            {
                Id = Utilities.TryParse(idTextBox.Text, -1),
                Status =
int.Parse((string)((ComboBoxItem)statusComboBox.SelectedItem).Tag),
                IsFiltered = true
            };

            if (datePicker.SelectedDate != null)
            {
                DateTime selectedDateTime = (DateTime)datePicker.SelectedDate;
                NewFilter.Date = Utilities.DateTimeToUnixTimeStamp(new
DateTime(selectedDateTime.Year, selectedDateTime.Month, selectedDateTime.Day));
            }
            else
            {
                NewFilter.Date = -1;
```

```
            }

            Filter = NewFilter;

            DialogResult = true;
        }
    }
}
```

Листинг А.32 – Код класса «FilterOrdersWindow»

```
using HardwareStore.Models;
using HardwareStore.Windows.Supplies;
using Microsoft.EntityFrameworkCore;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows;
using System.Windows.Controls;
using System.Windows.Data;
using System.Windows.Documents;
using System.Windows.Input;
using System.Windows.Media;
using System.Windows.Media.Imaging;
using System.Windows.Navigation;
using System.Windows.Shapes;

namespace HardwareStore.Windows.Orders
{
    /// <summary>
    /// Логика взаимодействия для OrdersUserControl.xaml
    /// </summary>
    public partial class OrdersUserControl : UserControl
    {
        private StoreContext _context = new();

        private readonly CollectionViewSource ordersViewSource;

        private OrdersFilter Filter { get; set; }

        public OrdersUserControl()
        {
            InitializeComponent();
            ordersViewSource =
(CollectionViewSource)FindResource(nameof(ordersViewSource));
            Filter = new();
        }

        private void UserControl_Loaded(object sender, RoutedEventArgs e)
        {
            _context.Orders.Load();
            ordersViewSource.Source = _context.Orders.Local.ToObservableCollection();

            ordersDataGrid.CanUserAddRows =
GroupPermissions.permissions[GroupPermissions.Group].Orders.CanAdd;
            ordersDataGrid.CanUserDeleteRows =
GroupPermissions.permissions[GroupPermissions.Group].Orders.CanDelete;
```

```
            ordersDataGrid.IsReadOnly =
!GroupPermissions.permissions[GroupPermissions.Group].Orders.CanEdit;
        }

        private void updateOrders_Click(object sender, RoutedEventArgs e)
        {
            Filter = new();

            _context = new StoreContext();
            _context.Orders.Load();
            ordersViewSource.Source = _context.Orders.Local.ToObservableCollection();
        }

        private void saveOrders_Click(object sender, RoutedEventArgs e)
        {
            Utilities.HandleSaveDataGrid(_context, ordersDataGrid);
        }

        private void filterOrders_Click(object sender, RoutedEventArgs e)
        {
            FilterOrdersWindow filterOrdersWindow = new()
            {
                Filter = Filter
            };

            if (filterOrdersWindow.ShowDialog() == true)
            {
                Filter = filterOrdersWindow.Filter;

CollectionViewSource.GetDefaultView(ordersDataGrid.ItemsSource).Refresh();
            }
        }

        private void CollectionViewSource_Filter(object sender, FilterEventArgs e)
        {
            if (e.Item is Order order)
            {
                bool IsDateInRange = Filter.Date <= order.Date && order.Date <= Filter.Date
+ 86400;

                if (!Filter.IsFiltered)
                {
                    e.Accepted = true;
                    return;
                }

                e.Accepted = (Filter.Date == -1 || IsDateInRange)
                    && (Filter.Id == -1 || Filter.Id == order.Id)
```

```
                && (Filter.Status == -1 || Filter.Status == (int)order.Status);
            }
        }

        private void reportOrders_Click(object sender, RoutedEventArgs e)
        {
            long timeStampLastMoth =
Utilities.DateTimeToUnixTimeStamp(DateTime.Today.AddMonths(-1));

            if (!_context.Orders.Any(o => timeStampLastMoth <= o.Date && o.Date <=
Utilities.GetCurrentUnixTimeStamp()))
            {
                MessageBox.Show("Для создания отчета необходимо наличие, как
минимум, одного заказа за последний месяц!", "Ошибка");
                return;
            }

            ReportOrdersWindow reportOrdersWindow = new();
            reportOrdersWindow.ShowDialog();
        }

        private void exportOrders_Click(object sender, RoutedEventArgs e)
        {
            List<string[]> headerRow = new() { new string[] { "ИД", "Дата заказа", "Статус
заказа" } };
            List<object[]> cellData = new() { };

            foreach (Order? order in _context.Orders.ToList())
            {
                if (order == null)
                    continue;

                cellData.Add(new object[] { order.Id,
Utilities.UnixTimeStampToDateTime(order.Date).ToString("dd-MM-yyyy HH:mm"),
order.Status });
            }

            Utilities.ExportToExcel("Заказы", headerRow, cellData);
        }
    }
}
```

Листинг А.33 – Код класса «OrdersUserControl»

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows;
using System.Windows.Controls;
using System.Windows.Data;
using System.Windows.Documents;
using System.Windows.Input;
using System.Windows.Media;
using System.Windows.Media.Imaging;
using System.Windows.Shapes;
using OxyPlot;
using OxyPlot.Axes;
using OxyPlot.Series;
using OxyPlot.Wpf;

namespace HardwareStore.Windows.Orders
{
   public class OrdersPlotModel
   {
      private StoreContext _context = new();

      public PlotModel Model { get; private set; }

      public OrdersPlotModel()
      {
         Model = new()
         {
            Title = "Число заказов за последние 30 дней",
            DefaultColors = new List<OxyColor> { OxyColors.SteelBlue, },
         };

         BarSeries series = new();

         DateTime dateTimeLastMonth = DateTime.Today.AddMonths(-1);
         long timeStampLastMoth =
Utilities.DateTimeToUnixTimeStamp(dateTimeLastMonth);

         List<Models.Order> orders = _context.Orders.Where(o => timeStampLastMoth
<= o.Date && o.Date <= Utilities.GetCurrentUnixTimeStamp()).OrderBy(o =>
o.Date).ToList();

         if (orders.Count == 0)
            return;

         Dictionary<string, int> days = new();
```

```
        CategoryAxis categoryAxis = new()
        {
            Position = AxisPosition.Left,
            Title = "Дата поставки"
        };

        for (int i = 0; i < 30; i++)
        {
            string day = DateTime.Today.AddDays(-i).ToString("dd.MM");

            categoryAxis.Labels.Add(day);
            days[day] = i;
        }

        Dictionary<string, int> amounts = new();

        foreach (Models.Order order in orders)
        {
            DateTime orderDateTime = Utilities.UnixTimeStampToDateTime(order.Date);
            string day = orderDateTime.ToString("dd.MM");

            if (!amounts.ContainsKey(day))
                amounts[day] = 0;

            amounts[day]++;
        }

        foreach (KeyValuePair<string, int> amount in amounts)
        {
            series.Items.Add(new BarItem(amount.Value, days[amount.Key]));
        }

        LinearAxis amountAxis = new()
        {
            Position = AxisPosition.Bottom,
            MinimumMajorStep = 1,
            Title = "Число заказов"
        };

        Model.Axes.Add(categoryAxis);
        Model.Axes.Add(amountAxis);
        Model.Series.Add(series);
    }
}

/// <summary>
/// Логика взаимодействия для ReportOrdersWindow.xaml
```

```
    /// </summary>
    public partial class ReportOrdersWindow : Window
    {
        public ReportOrdersWindow()
        {
            InitializeComponent();
        }
    }
}
```

Листинг А.34 – Код класса «ReportOrdersWindow»

```
using HardwareStore.Models;
using HardwareStore.Windows.Warehouses;
using Microsoft.EntityFrameworkCore;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows;
using System.Windows.Controls;
using System.Windows.Data;
using System.Windows.Documents;
using System.Windows.Input;
using System.Windows.Media;
using System.Windows.Media.Imaging;
using System.Windows.Navigation;
using System.Windows.Shapes;

namespace HardwareStore.Windows.Suppliers
{
    /// <summary>
    /// Логика взаимодействия для SuppliersUserControl.xaml
    /// </summary>
    public partial class SuppliersUserControl : UserControl
    {
        private StoreContext _context = new();

        private readonly CollectionViewSource suppliersViewSource;

        public SuppliersUserControl()
        {
            InitializeComponent();
            suppliersViewSource =
(CollectionViewSource)FindResource(nameof(suppliersViewSource));
        }

        private void UserControl_Loaded(object sender, RoutedEventArgs e)
        {
            _context.Suppliers.Load();
            suppliersViewSource.Source =
_context.Suppliers.Local.ToObservableCollection();

            suppliersDataGrid.CanUserAddRows =
GroupPermissions.permissions[GroupPermissions.Group].Suppliers.CanAdd;
            suppliersDataGrid.CanUserDeleteRows =
GroupPermissions.permissions[GroupPermissions.Group].Suppliers.CanDelete;
            suppliersDataGrid.IsReadOnly =
!GroupPermissions.permissions[GroupPermissions.Group].Suppliers.CanEdit;
```

```
        }

        private void updateSuppliers_Click(object sender, RoutedEventArgs e)
        {
            _context = new StoreContext();
            _context.Suppliers.Load();
            suppliersViewSource.Source =
_context.Suppliers.Local.ToObservableCollection();
        }

        private void saveSuppliers_Click(object sender, RoutedEventArgs e)
        {
            Utilities.HandleSaveDataGrid(_context, suppliersDataGrid);
        }

        private void exportSuppliers_Click(object sender, RoutedEventArgs e)
        {
            List<string[]> headerRow = new() { new string[] { "ИД", "Наименование
организации" } };
            List<object[]> cellData = new() { };

            foreach (Supplier? supplier in _context.Suppliers.ToList())
            {
                if (supplier == null)
                    continue;

                cellData.Add(new object[] { supplier.Id, supplier.Name });
            }

            Utilities.ExportToExcel("Поставщики", headerRow, cellData);
        }
    }
}
```

Листинг А.35 – Код класса «SuppliersUserControl»

```
using HardwareStore.Models;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows;
using System.Windows.Controls;
using System.Windows.Data;
using System.Windows.Documents;
using System.Windows.Input;
using System.Windows.Media;
using System.Windows.Media.Imaging;
using System.Windows.Shapes;
using static System.Runtime.InteropServices.JavaScript.JSType;

namespace HardwareStore.Windows.Supplies
{
    public struct SuppliesFilter
    {
        public SuppliesFilter(StoreContext context)
        {
            Id = -1;
            SupplierId = -1;
            ComponentId = -1;
            WarehouseId = -1;

            Supply? maxSupplyPrice = context.Supplies.OrderByDescending(s =>
s.SupplyPrice).FirstOrDefault();
            Supply? maxAmount = context.Supplies.OrderByDescending(s =>
s.Amount).FirstOrDefault();

            SupplyPrice = (0, (maxSupplyPrice != null) ? maxSupplyPrice.SupplyPrice : 0);
            Amount = (0, (maxAmount != null) ? maxAmount.Amount : 0);

            Date = -1;

            IsFiltered = false;
        }

        public int Id { get; set; }
        public int SupplierId { get; set; }
        public int ComponentId { get; set; }
        public int WarehouseId { get; set; }

        public (float Min, float Max) SupplyPrice { get; set; }
        public (int Min, int Max) Amount { get; set; }
```

```
        public long Date { get; set; }
        public bool IsFiltered { get; set; }
    }

    /// <summary>
    /// Логика взаимодействия для FilterSuppliesWindow.xaml
    /// </summary>
    public partial class FilterSuppliesWindow : Window
    {
        public SuppliesFilter Filter;

        public FilterSuppliesWindow()
        {
            InitializeComponent();
        }

        private void Window_Loaded(object sender, RoutedEventArgs e)
        {
            if (Filter.Id >= 1)
                idTextBox.Text = Filter.Id.ToString();

            if (Filter.ComponentId >= 1)
                componentIdTextBox.Text = Filter.ComponentId.ToString();

            if (Filter.WarehouseId >= 1)
                warehouseIdTextBox.Text = Filter.WarehouseId.ToString();

            if (Filter.SupplierId >= 1)
                supplierIdTextBox.Text = Filter.SupplierId.ToString();

            if (Filter.Date >= 0)
                supplyDatePicker.SelectedDate =
Utilities.UnixTimeStampToDateTime(Filter.Date);

            minSupplyPriceTextBox.Text = Filter.SupplyPrice.Min.ToString();
            maxSupplyPriceTextBox.Text = Filter.SupplyPrice.Max.ToString();

            minAmountTextBox.Text = Filter.Amount.Min.ToString();
            maxAmountTextBox.Text = Filter.Amount.Max.ToString();
        }

        private void idTextBox_TextChanged(object sender, TextChangedEventArgs e)
        {
            Utilities.ValidateTextBox(RegularExpressions.GetUnsignedIntRegex(), sender,
e);
        }
```

```
        private void supplierIdTextBox_TextChanged(object sender,
TextChangedEventArgs e)
        {
            Utilities.ValidateTextBox(RegularExpressions.GetUnsignedIntRegex(), sender,
e);
        }

        private void componentIdTextBox_TextChanged(object sender,
TextChangedEventArgs e)
        {
            Utilities.ValidateTextBox(RegularExpressions.GetUnsignedIntRegex(), sender,
e);
        }

        private void warehouseIdTextBox_TextChanged(object sender,
TextChangedEventArgs e)
        {
            Utilities.ValidateTextBox(RegularExpressions.GetUnsignedIntRegex(), sender,
e);
        }

        private void minSupplyPriceTextBox_TextChanged(object sender,
TextChangedEventArgs e)
        {
            Utilities.ValidateTextBox(RegularExpressions.GetUnsignedFloatRegex(), sender,
e);
        }

        private void maxSupplyPriceTextBox_TextChanged(object sender,
TextChangedEventArgs e)
        {
            Utilities.ValidateTextBox(RegularExpressions.GetUnsignedFloatRegex(), sender,
e);
        }

        private void minAmountTextBox_TextChanged(object sender,
TextChangedEventArgs e)
        {
            Utilities.ValidateTextBox(RegularExpressions.GetUnsignedIntRegex(), sender,
e);
        }

        private void maxAmountTextBox_TextChanged(object sender,
TextChangedEventArgs e)
        {
            Utilities.ValidateTextBox(RegularExpressions.GetUnsignedIntRegex(), sender,
e);
        }
```

```
        private void filterButton_Click(object sender, RoutedEventArgs e)
        {
            SuppliesFilter NewFilter = new()
            {
                SupplyPrice = (Utilities.TryParse(minSupplyPriceTextBox.Text,
Filter.SupplyPrice.Min), Utilities.TryParse(maxSupplyPriceTextBox.Text,
Filter.SupplyPrice.Max)),
                Amount = (Utilities.TryParse(minAmountTextBox.Text, Filter.Amount.Min),
Utilities.TryParse(maxAmountTextBox.Text, Filter.Amount.Max)),
                Id = Utilities.TryParse(idTextBox.Text, -1),
                WarehouseId = Utilities.TryParse(warehouseIdTextBox.Text, -1),
                ComponentId = Utilities.TryParse(componentIdTextBox.Text, -1),
                SupplierId = Utilities.TryParse(supplierIdTextBox.Text, -1),
                IsFiltered = true
            };

            if (supplyDatePicker.SelectedDate != null)
            {
                DateTime selectedDateTime = (DateTime)supplyDatePicker.SelectedDate;
                NewFilter.Date = Utilities.DateTimeToUnixTimeStamp(new
DateTime(selectedDateTime.Year, selectedDateTime.Month, selectedDateTime.Day));
            }
            else
            {
                NewFilter.Date = -1;
            }

            Filter = NewFilter;

            DialogResult = true;
        }
    }
}
```

Листинг А.36 – Код класса «FilterSuppliesWindow»

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows;
using System.Windows.Controls;
using System.Windows.Data;
using System.Windows.Documents;
using System.Windows.Input;
using System.Windows.Media;
using System.Windows.Media.Imaging;
using System.Windows.Shapes;
using System.Xml.Linq;
using Microsoft.EntityFrameworkCore;
using OxyPlot;
using OxyPlot.Axes;
using OxyPlot.Series;
using OxyPlot.Wpf;

namespace HardwareStore.Windows.Supplies
{
   public class SuppliesPlotModel
   {
      private StoreContext _context = new();

      public PlotModel Model { get; private set; }

      public SuppliesPlotModel()
      {
         Model = new()
         {
            Title = "Число поставок за последние 30 дней",
            DefaultColors = new List<OxyColor> { OxyColors.SteelBlue, },
         };

         BarSeries series = new();

         DateTime dateTimeLastMonth = DateTime.Today.AddMonths(-1);
         long timeStampLastMoth =
Utilities.DateTimeToUnixTimeStamp(dateTimeLastMonth);

         List<Models.Supply> supplies = _context.Supplies.Where(o =>
timeStampLastMoth <= o.Date && o.Date <=
Utilities.GetCurrentUnixTimeStamp()).OrderBy(o => o.Date).ToList();

         if (supplies.Count == 0)
            return;
```

```
        Dictionary<string, int> days = new();

        CategoryAxis categoryAxis = new()
        {
            Position = AxisPosition.Left,
            Title = "Дата поставки"
        };

        for (int i = 0; i < 30; i++)
        {
            string day = DateTime.Today.AddDays(-i).ToString("dd.MM");

            categoryAxis.Labels.Add(day);
            days[day] = i;
        }

        Dictionary<string, int> amounts = new();

        foreach (Models.Supply supply in supplies)
        {
            DateTime supplyDateTime =
Utilities.UnixTimeStampToDateTime(supply.Date);
            string day = supplyDateTime.ToString("dd.MM");

            if (!amounts.ContainsKey(day))
                amounts[day] = 0;

            amounts[day]++;
        }

        foreach (KeyValuePair<string, int> amount in amounts)
        {
            series.Items.Add(new BarItem(amount.Value, days[amount.Key]));
        }

        LinearAxis amountAxis = new()
        {
            Position = AxisPosition.Bottom,
            MinimumMajorStep = 1,
            Title = "Число поставок"
        };

        Model.Axes.Add(categoryAxis);
        Model.Axes.Add(amountAxis);
        Model.Series.Add(series);
    }
}
```

```
    /// <summary>
    /// Логика взаимодействия для ReportSuppliesWindow.xaml
    /// </summary>
    public partial class ReportSuppliesWindow : Window
    {
        public ReportSuppliesWindow()
        {
            InitializeComponent();
        }
    }
}
```

Листинг А.37 – Код класса «ReportSuppliesWindow»

```
using HardwareStore.Models;
using HardwareStore.Windows.Users;
using Microsoft.EntityFrameworkCore;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows;
using System.Windows.Controls;
using System.Windows.Data;
using System.Windows.Documents;
using System.Windows.Input;
using System.Windows.Media;
using System.Windows.Media.Imaging;
using System.Windows.Navigation;
using System.Windows.Shapes;

namespace HardwareStore.Windows.Supplies
{
    /// <summary>
    /// Логика взаимодействия для SuppliesUserControl.xaml
    /// </summary>
    public partial class SuppliesUserControl : UserControl
    {
        private StoreContext _context = new();

        private readonly CollectionViewSource suppliesViewSource;

        private SuppliesFilter Filter { get; set; }

        public SuppliesUserControl()
        {
            InitializeComponent();
            suppliesViewSource =
(CollectionViewSource)FindResource(nameof(suppliesViewSource));
            Filter = new(_context);
        }

        private void UserControl_Loaded(object sender, RoutedEventArgs e)
        {
            _context.Supplies.Load();
            suppliesViewSource.Source =
_context.Supplies.Local.ToObservableCollection();

            suppliesDataGrid.CanUserAddRows =
GroupPermissions.permissions[GroupPermissions.Group].Supplies.CanAdd;
```

```
            suppliesDataGrid.CanUserDeleteRows =
GroupPermissions.permissions[GroupPermissions.Group].Supplies.CanDelete;
            suppliesDataGrid.IsReadOnly =
!GroupPermissions.permissions[GroupPermissions.Group].Supplies.CanEdit;
        }

        private void updateSupplies_Click(object sender, RoutedEventArgs e)
        {
            Filter = new(_context);

            _context = new StoreContext();
            _context.Supplies.Load();
            suppliesViewSource.Source =
_context.Supplies.Local.ToObservableCollection();
        }

        private void saveSupplies_Click(object sender, RoutedEventArgs e)
        {
            Utilities.HandleSaveDataGrid(_context, suppliesDataGrid);
        }

        private void filterSupplies_Click(object sender, RoutedEventArgs e)
        {
            FilterSuppliesWindow filterSuppliesWindow = new()
            {
                Filter = Filter
            };

            if (filterSuppliesWindow.ShowDialog() == true)
            {
                Filter = filterSuppliesWindow.Filter;

CollectionViewSource.GetDefaultView(suppliesDataGrid.ItemsSource).Refresh();
            }
        }

        private void CollectionViewSource_Filter(object sender, FilterEventArgs e)
        {
            if (e.Item is Supply supply)
            {
                bool IsSupplyPriceInRange = Filter.SupplyPrice.Min <= supply.SupplyPrice
&& supply.SupplyPrice <= Filter.SupplyPrice.Max,
                    IsAmountInRange = Filter.Amount.Min <= supply.Amount &&
supply.Amount <= Filter.Amount.Max,
                    IsDateInRange = Filter.Date <= supply.Date && supply.Date <= Filter.Date
+ 86400;

                if (!Filter.IsFiltered)
```

```
            {
                e.Accepted = true;
                return;
            }

            e.Accepted = IsSupplyPriceInRange && IsAmountInRange
                && (Filter.Date == -1 || IsDateInRange)
                && (Filter.Id == -1 || Filter.Id == supply.Id)
                && (Filter.ComponentId == -1 || Filter.ComponentId ==
supply.ComponentId)
                && (Filter.WarehouseId == -1 || Filter.WarehouseId ==
supply.WarehouseId)
                && (Filter.SupplierId == -1 || Filter.SupplierId == supply.SupplierId);
        }
    }

    private void reportSupplies_Click(object sender, RoutedEventArgs e)
    {
        long timeStampLastMoth =
Utilities.DateTimeToUnixTimeStamp(DateTime.Today.AddMonths(-1));

        if (!_context.Supplies.Any(s => timeStampLastMoth <= s.Date && s.Date <=
Utilities.GetCurrentUnixTimeStamp()))
        {
            MessageBox.Show("Для создания отчета необходима, как минимум, одна
поставка за последний месяц!", "Ошибка");
            return;
        }

        ReportSuppliesWindow reportSuppliesWindow = new();
        reportSuppliesWindow.ShowDialog();
    }

    private void exportSupplies_Click(object sender, RoutedEventArgs e)
    {
        List<string[]> headerRow = new() { new string[] { "ИД", "ИД поставщика",
"ИД комплектующего", "ИД склада", "Закупочная цена (руб)", "Количество товара
(шт)", "Дата поставки" } };
        List<object[]> cellData = new() { };

        foreach (Supply? supply in _context.Supplies.ToList())
        {
            if (supply == null)
                continue;

            cellData.Add(new object[] { supply.Id, supply.SupplierId,
supply.ComponentId, supply.WarehouseId, supply.SupplyPrice, supply.Amount,
```

```
Utilities.UnixTimeStampToDateTime(supply.Date).ToString("dd-MM-yyyy HH:mm")
});
            }

            Utilities.ExportToExcel("Поставки", headerRow, cellData);
        }
    }
}
```

Листинг А.38 – Код класса «SuppliesUserControl»

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Security.Cryptography;
using System.Text;
using System.Threading.Tasks;
using System.Windows;
using System.Windows.Controls;
using System.Windows.Data;
using System.Windows.Documents;
using System.Windows.Input;
using System.Windows.Media;
using System.Windows.Media.Imaging;
using System.Windows.Shapes;
using HardwareStore.Models;

namespace HardwareStore.Windows.Users
{
    /// <summary>
    /// Логика взаимодействия для AddUserWindow.xaml
    /// </summary>
    public partial class AddUserWindow : Window
    {
        public int UserID { get; private set; }

        public AddUserWindow()
        {
            InitializeComponent();
        }

        private void addButton_Click(object sender, RoutedEventArgs e)
        {
            if (passwordBox.Password.Length == 0)
            {
                MessageBox.Show("Введите пароль!", "Ошибка");
                return;
            }

            using StoreContext context = new();

            Group group =
(Group)int.Parse((string)((TextBlock)groupComboBox.SelectedItem).Tag);

            User user = new() { Password = Utilities.GetHash(SHA256.Create(),
passwordBox.Password), Group = group, RegistrationDate =
DateTimeOffset.Now.ToUnixTimeSeconds() };

            context.Users.Add(user);
```

```
            context.SaveChanges();

            UserID = context.Users.OrderBy(u => u.Id).Last().Id;

            DialogResult = true;
        }
    }
}
```

Листинг А.39 – Код класса «AddUserWindow»

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows;
using System.Windows.Controls;
using System.Windows.Data;
using System.Windows.Documents;
using System.Windows.Input;
using System.Windows.Media;
using System.Windows.Media.Imaging;
using System.Windows.Shapes;
using HardwareStore.Models;

namespace HardwareStore.Windows.Users
{
    /// <summary>
    /// Логика взаимодействия для RemoveUserWindow.xaml
    /// </summary>
    public partial class RemoveUserWindow : Window
    {
        public RemoveUserWindow()
        {
            InitializeComponent();
        }

        private void removeButton_Click(object sender, RoutedEventArgs e)
        {
            if (idTextBox.Text.Length == 0)
            {
                MessageBox.Show("Введите ИД пользователя!", "Ошибка");
                return;
            }

            using StoreContext context = new();
            User? user = context.Users.Where(u => u.Id ==
int.Parse(idTextBox.Text)).FirstOrDefault();
            if (user != null)
            {
                context.Users.Remove(user);
                context.SaveChanges();
                DialogResult = true;
            }
            else
            {
                MessageBox.Show("Данного пользователя не существует!", "Ошибка");
            }
```

```
        }

        private void idTextBox_TextChanged(object sender, TextChangedEventArgs e)
        {
            Utilities.ValidateTextBox(RegularExpressions.GetUnsignedIntRegex(), sender,
e);
        }
    }
}
```

Листинг А.40 – Код класса «RemoveUserWindow»

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows;
using System.Windows.Controls;
using System.Windows.Data;
using System.Windows.Documents;
using System.Windows.Input;
using System.Windows.Media;
using System.Windows.Media.Imaging;
using System.Windows.Shapes;
using System.Security.Cryptography;
using System.Text.RegularExpressions;
using HardwareStore.Models;

namespace HardwareStore.Windows.Users
{
    /// <summary>
    /// Логика взаимодействия для SetUserWindow.xaml
    /// </summary>
    public partial class SetUserWindow : Window
    {
        public SetUserWindow()
        {
            InitializeComponent();
        }

        private void setButton_Click(object sender, RoutedEventArgs e)
        {
            if (idTextBox.Text.Length == 0)
            {
                MessageBox.Show("Введите ИД пользователя!", "Ошибка");
                return;
            }

            using StoreContext context = new();
            User? user = context.Users.Where(u => u.Id ==
int.Parse(idTextBox.Text)).FirstOrDefault();
            if (user != null)
            {
                if (passwordBox.Password.Length > 0)
                    user.Password = Utilities.GetHash(SHA256.Create(),
passwordBox.Password);

                user.Group =
(Models.Group)int.Parse((string)((TextBlock)groupComboBox.SelectedItem).Tag);
```

```
            context.SaveChanges();

            DialogResult = true;
        }
        else
        {
            MessageBox.Show("Данного пользователя не существует!", "Ошибка");
        }
    }

    private void idTextBox_TextChanged(object sender, TextChangedEventArgs e)
    {
        Utilities.ValidateTextBox(RegularExpressions.GetUnsignedIntRegex(), sender,
e);

        using StoreContext context = new();

        User? user = context.Users.Where(u => u.Id ==
int.Parse(idTextBox.Text)).FirstOrDefault();
        if (user != null)
        {
            groupComboBox.SelectedIndex = (int)user.Group;
        }
    }
  }
}
```

Листинг А.41 – Код класса «SetUserWindow»

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows;
using System.Windows.Controls;
using System.Windows.Data;
using System.Windows.Documents;
using System.Windows.Input;
using System.Windows.Media;
using System.Windows.Media.Imaging;
using System.Windows.Navigation;
using System.Windows.Shapes;
using HardwareStore.Models;
using Microsoft.EntityFrameworkCore;

namespace HardwareStore.Windows.Users
{
    /// <summary>
    /// Логика взаимодействия для UsersUserControl.xaml
    /// </summary>
    public partial class UsersUserControl : UserControl
    {
        private StoreContext _context = new();

        private readonly CollectionViewSource usersViewSource;

        public UsersUserControl()
        {
            InitializeComponent();
            usersViewSource =
(CollectionViewSource)FindResource(nameof(usersViewSource));
        }

        private void UserControl_Loaded(object sender, RoutedEventArgs e)
        {
            _context.Users.Load();
            usersViewSource.Source = _context.Users.Local.ToObservableCollection();

            usersDataGrid.CanUserDeleteRows =
GroupPermissions.permissions[GroupPermissions.Group].Users.CanDelete;
            usersDataGrid.IsReadOnly =
!GroupPermissions.permissions[GroupPermissions.Group].Users.CanEdit;

            addUser.IsEnabled =
GroupPermissions.permissions[GroupPermissions.Group].Users.CanAdd;
```

```
            removeUser.IsEnabled =
GroupPermissions.permissions[GroupPermissions.Group].Users.CanDelete;
            setUser.IsEnabled =
GroupPermissions.permissions[GroupPermissions.Group].Users.CanEdit;
        }

        private void updateUsers_Click(object sender, RoutedEventArgs e)
        {
            _context = new StoreContext();
            _context.Users.Load();
            usersViewSource.Source = _context.Users.Local.ToObservableCollection();
        }

        private void saveUsers_Click(object sender, RoutedEventArgs e)
        {
            Utilities.HandleSaveDataGrid(_context, usersDataGrid);
        }

        private void addUser_Click(object sender, RoutedEventArgs e)
        {
            AddUserWindow addUserWindow = new();

            if (addUserWindow.ShowDialog() == true)
            {
                MessageBox.Show($"Пользователь с ИД {addUserWindow.UserID}
успешно создан!", "Успех");
                updateUsers_Click(sender, e);
            }
        }

        private void setUser_Click(object sender, RoutedEventArgs e)
        {
            SetUserWindow setUserWindow = new();

            if (setUserWindow.ShowDialog() == true)
            {
                MessageBox.Show($"Данные о пользователе успешно изменены!",
"Успех");
                updateUsers_Click(sender, e);
            }
        }

        private void removeUser_Click(object sender, RoutedEventArgs e)
        {
            RemoveUserWindow removeUserWindow = new();

            if (removeUserWindow.ShowDialog() == true)
            {
```

```
            MessageBox.Show($"Данные о пользователе удалены!", "Успех");
            updateUsers_Click(sender, e);
        }
    }

    private void exportUsers_Click(object sender, RoutedEventArgs e)
    {
        List<string[]> headerRow = new() { new string[] { "ИД", "Группа прав", "Дата
последнего входа", "Дата регистрации" } };
        List<object[]> cellData = new() { };

        foreach (User? user in _context.Users.ToList())
        {
            if (user == null)
                continue;

            cellData.Add(new object[] { user.Id, user.Group, user.LastVisitDateText,
user.RegistrationDateText });
        }

        Utilities.ExportToExcel("Пользователи", headerRow, cellData);
    }
  }
}
```

Листинг А.42 – Код класса «UsersUserControl»

```
using HardwareStore.Windows.Supplies;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows;
using System.Windows.Controls;
using System.Windows.Data;
using System.Windows.Documents;
using System.Windows.Input;
using System.Windows.Media;
using System.Windows.Media.Imaging;
using System.Windows.Shapes;

namespace HardwareStore.Windows.Warehouses
{
    public struct WarehousesFilter
    {
        public WarehousesFilter()
        {
            Id = -1;
            Zip = -1;

            City = "";
            Street = "";
            House = "";

            IsFiltered = false;
        }

        public int Id { get; set; }
        public string City { get; set; }
        public string Street { get; set; }
        public string House { get; set; }
        public int Zip { get; set; }
        public bool IsFiltered { get; set; }
    }

    /// <summary>
    /// Логика взаимодействия для FilterWarehousesWindow.xaml
    /// </summary>
    public partial class FilterWarehousesWindow : Window
    {
        public WarehousesFilter Filter { get; set; }

        public FilterWarehousesWindow()
        {
```

```
            InitializeComponent();
        }

        private void Window_Loaded(object sender, RoutedEventArgs e)
        {
            if (Filter.Id >= 1)
                idTextBox.Text = Filter.Id.ToString();

            if (Filter.Zip >= 0)
                zipTextBox.Text = Filter.Zip.ToString();

            cityTextBox.Text = Filter.City.ToString();
            streetTextBox.Text = Filter.Street.ToString();
            houseTextBox.Text = Filter.House.ToString();
        }

        private void idTextBox_TextChanged(object sender, TextChangedEventArgs e)
        {
            Utilities.ValidateTextBox(RegularExpressions.GetUnsignedIntRegex(), sender,
e);
        }

        private void cityTextBox_TextChanged(object sender, TextChangedEventArgs e)
        {

        }

        private void streetTextBox_TextChanged(object sender, TextChangedEventArgs e)
        {

        }

        private void houseTextBox_TextChanged(object sender, TextChangedEventArgs e)
        {

        }

        private void zipTextBox_TextChanged(object sender, TextChangedEventArgs e)
        {
            Utilities.ValidateTextBox(RegularExpressions.GetUnsignedIntRegex(), sender,
e);
        }

        private void filterButton_Click(object sender, RoutedEventArgs e)
        {
            WarehousesFilter NewFilter = new()
            {
                Id = Utilities.TryParse(idTextBox.Text, -1),
```

```
            City = cityTextBox.Text,
            Street = streetTextBox.Text,
            House = houseTextBox.Text,
            Zip = Utilities.TryParse(zipTextBox.Text, -1),
            IsFiltered = true
        };

        Filter = NewFilter;

        DialogResult = true;
    }
  }
}
```

Листинг А.43 – Код класса «FilterWarehousesWindow»

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows;
using System.Windows.Controls;
using System.Windows.Data;
using System.Windows.Documents;
using System.Windows.Input;
using System.Windows.Media;
using System.Windows.Media.Imaging;
using System.Windows.Navigation;
using System.Windows.Shapes;
using HardwareStore.Models;
using HardwareStore.Windows.Supplies;
using HardwareStore.Windows.Users;
using Microsoft.EntityFrameworkCore;

namespace HardwareStore.Windows.Warehouses
{
    /// <summary>
    /// Логика взаимодействия для WarehousesUserControl.xaml
    /// </summary>
    public partial class WarehousesUserControl : UserControl
    {
        private StoreContext _context = new();

        private readonly CollectionViewSource warehousesViewSource;

        private WarehousesFilter Filter { get; set; }

        public WarehousesUserControl()
        {
            InitializeComponent();
            warehousesViewSource =
(CollectionViewSource)FindResource(nameof(warehousesViewSource));
            Filter = new();
        }

        private void UserControl_Loaded(object sender, RoutedEventArgs e)
        {
            _context.Warehouses.Load();
            warehousesViewSource.Source =
_context.Warehouses.Local.ToObservableCollection();

            warehousesDataGrid.CanUserAddRows =
GroupPermissions.permissions[GroupPermissions.Group].Warehouses.CanAdd;
```

```
            warehousesDataGrid.CanUserDeleteRows =
GroupPermissions.permissions[GroupPermissions.Group].Warehouses.CanDelete;
            warehousesDataGrid.IsReadOnly =
!GroupPermissions.permissions[GroupPermissions.Group].Warehouses.CanEdit;
        }

        private void updateWarehouses_Click(object sender, RoutedEventArgs e)
        {
            Filter = new();

            _context = new StoreContext();
            _context.Warehouses.Load();
            warehousesViewSource.Source =
_context.Warehouses.Local.ToObservableCollection();
        }

        private void saveWarehouses_Click(object sender, RoutedEventArgs e)
        {
            Utilities.HandleSaveDataGrid(_context, warehousesDataGrid);
        }

        private void filterWarehouses_Click(object sender, RoutedEventArgs e)
        {
            FilterWarehousesWindow filterWarehousesWindow = new()
            {
                Filter = Filter
            };

            if (filterWarehousesWindow.ShowDialog() == true)
            {
                Filter = filterWarehousesWindow.Filter;

CollectionViewSource.GetDefaultView(warehousesDataGrid.ItemsSource).Refresh();
            }
        }

        private void CollectionViewSource_Filter(object sender, FilterEventArgs e)
        {
            if (e.Item is Warehouse warehouse)
            {
                if (!Filter.IsFiltered)
                {
                    e.Accepted = true;
                    return;
                }

                e.Accepted = (Filter.Id == -1 || Filter.Id == warehouse.Id)
                    && (Filter.Zip == -1 || Filter.Zip == warehouse.Zip)
```

```
                && (Filter.City.Length == 0 || Filter.City == warehouse.City)
                && (Filter.Street.Length == 0 || Filter.Street == warehouse.Street)
                && (Filter.House.Length == 0 || Filter.House == warehouse.House);
            }
        }

        private void exportWarehouses_Click(object sender, RoutedEventArgs e)
        {
            List<string[]> headerRow = new() { new string[] { "ИД", "Город", "Улица",
"Дом", "Индекс" } };
            List<object[]> cellData = new() { };

            foreach (Warehouse? warehouse in _context.Warehouses.ToList())
            {
                if (warehouse == null)
                    continue;

                cellData.Add(new object[] { warehouse.Id, warehouse.City, warehouse.Street,
warehouse.House, warehouse.Zip });
            }

            Utilities.ExportToExcel("Склады", headerRow, cellData);
        }
    }
}
```

Листинг А.44 – Код класса «WarehousesUserControl»

ПРИЛОЖЕНИЕ Б.
Техническое задание

ПРАВИТЕЛЬСТВО РОССИЙСКОЙ ФЕДЕРАЦИИ
НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ
«ВЫСШАЯ ШКОЛА ЭКОНОМИКИ»

Факультет социально-экономических и компьютерных наук Образовательная программа бакалавриата «Программная инженерия»

СОГЛАСОВАНО	УТВЕРЖДАЮ
Руководитель проекта, старший преподаватель кафедры ИТБ НИУ ВШЭ	Академический руководитель образовательной программы «Программная инженерия»
_____ Г.И. Рустамханова	_____ В.В. Ланин
«___» _____ 2023 г.	«___» _____ 2023 г.

Подп. и дата	
Инв. № дубл	
Взаим. инв. №	
Подп. и дата	
Инв. № подл.	RU.17701729.05.15-01 ТЗ 01-1

ИНФОРМАЦИОННАЯ СИСТЕМА ДЛЯ ПЕРСОНАЛА МАГАЗИНА КОМПЬЮТЕРНЫХ КОМПЛЕКТУЮЩИХ

Техническое задание

ЛИСТ УТВЕРЖДЕНИЯ

RU.17701729.05.15-01 ТЗ 01-1-ЛУ

Исполнитель:
студент группы ПИ21-1
_____ А.В. Кривов
«____» _____ 2023 г.

Пермь 2023

УТВЕРЖДЁН
RU.17701729.05.15-01 ТЗ 01-1-ЛУ

ИНФОРМАЦИОННАЯ СИСТЕМА ДЛЯ ПЕРСОНАЛА МАГАЗИНА КОМПЬЮТЕРНЫХ КОМПЛЕКТУЮЩИХ

Техническое задание

RU.17701729.05.15-01 ТЗ 01-1

Листов 14

Инв. № подл.	*Подп. и дата*	*Взаим. инв. №*	*Инв. № дубл*	*Подп. и дата*
RU.17701729.05.15-01 ТЗ 01-1				

Пермь 2023

1 Введение

1.1 Наименование программы

Наименование: «Информационная система для персонала магазина компьютерных комплектующих».

1.2 Краткая характеристика области применения программы

Информационная система предназначена для автоматизации работы персонала магазина компьютерных комплектующих. Она обеспечивает доступ к данным о продукции, складах, заказах и персонале, а также позволяет управлять процессом продажи компьютерных комплектующих и вести учет товаров на складах. Система может использоваться как в малых, так и в крупных магазинах.

2 Основания для разработки

2.1 Основание для проведения разработки

Программа выполнена в рамках курсовой работы по теме «Разработка информационной системы для персонала магазина компьютерных комплектующих» (Факультет социально-экономических и компьютерных наук), в соответствии с учебным планом подготовки бакалавров по направлению «Программная инженерия».

2.2 Наименование темы разработки

Наименование темы разработки: «Разработка информационной системы для персонала магазина компьютерных комплектующих» или «Development of an Information Support System for a Hardware Store Staff».

3 Назначение разработки

3.1 Функциональное назначение программы

Программа предоставляет удобное и эффективное решение для управления персоналом, продажами и складским учетом.

3.2 Эксплуатационное назначение программы

Эксплуатационное назначение информационной системы для персонала магазина компьютерных комплектующих заключается в упрощении и оптимизации работы с базой данных магазина, автоматизации процессов взаимодействия с клиентами, управлении заказами и складским учетом, а также в повышении общей эффективности работы персонала магазина.

4 Требования к программе

4.1 Требования к функциональным характеристикам

Данная программа имеет клиентскую часть, при этом база данных хранится на удаленном сервере.

Клиентская часть должна быть реализована в виде десктопного приложения с пользовательским интерфейсом.

Приложение должно предоставлять следующие возможности:

- управление персоналом;
- управление складами;
- управление комплектующими;
- управление складским учетом;
- управление заказами;
- управление поставками и поставщиками.

Входные данные со стороны пользователя должны вводиться последовательно с использованием элементов интерфейса.

Выходные данные организованы в виде таблиц.

4.2 Требования к надёжности

Программа должна безотказно работать (не завершаться аварийно) в течение времени, требуемого пользователю, вне зависимости от действий пользователя.

В случае если приложение прекратило работу по причине, не зависящей от разработчика, восстановление должно происходить в течение времени, достаточного для перезапуска приложения.

В случае если приложение прекратило работу по причине отказа используемого устройства, восстановление должно происходить в течение времени, достаточного для перезапуска данного устройства.

4.3 Условия эксплуатации

Программа должна функционировать в любой среде, в которой возможно функционирование используемого устройства.

Очное обслуживание не требуется, характер выполняемых задач не требует регулярных обновлений. Программа не должна терять функционал при отсутствии обновлений, за исключением вопросов оптимизации времени/памяти.

Для пользования программой необходимы базовые знания по работе с ЭВМ.

4.4 Требования к составу и параметрам технических средств

Для функционирования приложения необходима система, поддерживающая функционирование приложений платформы .NET 7.

4.5 Требования к информационной и программной совместимости

Для функционирования приложения необходима среда, поддерживаемая платформой .NET 7.

Исходный код программы должен быть написан на языке C# в среде Microsoft Visual Studio с использованием библиотек и средств WPF для ОС Windows.

4.6 Требования к маркировке и упаковке

Требования к маркировке и упаковке не предъявляются, приложение распространяется в виде исполняемого файла со съёмного носителя.

5 Требования к программной документации

5.1 Состав программной документации

Состав программной документации:

- «Информационная система для персонала магазина компьютерных комплектующих»
  Техническое задание системы (ГОСТ 19.201-78);
- «Информационная система для персонала магазина компьютерных комплектующих»
  Руководство оператора системы (ГОСТ 19.505-79);
- «Информационная система для персонала магазина компьютерных комплектующих»
  Текст программы системы (ГОСТ 19.401-78).

5.2 Специальные требования к программной документации

Документы к программе должны быть выполнены в соответствии с ГОСТ 19.106-78 и ГОСТами к каждому виду документа (см. п. 5.1).

Пояснительная записка должна быть загружена в систему Антиплагиат через LMS «НИУ ВШЭ». Лист, подтверждающий загрузку пояснительной записки, сдается в учебный офис вместе со всеми материалами не позже, чем за день до защиты курсовой работы.

Техническое задание и пояснительная записка, титульные листы других документов должны быть напечатаны, подписаны академическим руководителем образовательной программы 09.03.04 «Программная инженерия», руководителем разработки и исполнителем перед сдачей курсовой работы в учебный офис не позже одного дня до защиты.

Документация и программа также сдается в электронном виде в формате .pdf или .docx, в архиве формата .zip или .rar.

Материалы курсовой работы должны быть загружены одним или несколькими архивами в проект дисциплины «Курсовая работа 2022-2023» в личном

кабинете в информационной образовательной среде «LMS» («Learning Management System») НИУ ВШЭ.

6 Технико-экономические показатели

В рамках данной работы расчёт экономической эффективности не предусмотрен, так как она выполняется в рамках курсового проекта с открытым исходным кодом.

Использование разрабатываемого инструмента предназначено для автоматизации складского учета, управления персоналом, комплектующими, заказами и поставками.

7 Стадии и этапы разработки

Стадии разработки:

1) выбор темы курсовой работы, анализ задачи и путей её решения;
2) установка требований к Приложению, создание графика работ, проектирование Приложения и плана тестирования, написание Технического задания;
3) разработка и тестирование прототипа и итогового варианта Приложения;
4) сдача курсовой работы и получение подтверждения о его приёмке.

Этапы курсовой работы, входящие в них задачи, сроки их выполнения, ресурсы и ожидаемые результаты описаны в таблице 1.

Таблица 1 – Этапы Проекта

Номер ИСР	Описание задачи	Сроки выполнения	Исполнитель	Результаты
1	Анализ и планирование проекта			
1.1	Выявить требования Заказчика	11.01.2023	А.В. Кривов	Протоколы общения с Заказчиком
1.2	Формализовать требования Заказчика	12.01.2023	А.В. Кривов	Реестр требований
1.3	Протестировать требования Заказчика	13.01.2023	А.В. Кривов	План тестирования
1.4	Утвердить требования Заказчика	14.01.2023	А.В. Кривов	Утвержденный реестр требований
2	Проектирование и создание прототипа продукта			
2.1	Спроектировать архитектуру Приложения	15.01.2023	А.В. Кривов	Техническое задание, проект Приложения
2.2	Разработать прототип Приложения	20.01.2023	А.В. Кривов	Прототип Приложения, руководство пользователя, руководство программиста
2.3	Протестировать прототип Приложения	26.01.2023	А.В. Кривов	План тестирования
2.4	Актуализировать требования Заказчика	29.01.2023	А.В. Кривов	Прототип Приложения
2.5	Актуализировать проектные документы	01.02.2023	А.В. Кривов	Техническое задание
2.5.1	Сформировать уточненный список требований	05.02.2023	А.В. Кривов	Реестр требований
2.5.2	Протестировать требования	10.02.2023	А.В. Кривов	План тестирования
2.5.3	Провести демонстрацию прототипа Приложения Заказчику	14.02.2023	А.В. Кривов	Документ о принятии работ

Номер ИСР	Описание задачи	Сроки выполнения	Исполнитель	Результаты
3	Реализация и тестирование			
3.1	Актуализировать требования Заказчика	15.02.2023	А.В. Кривов	Реестр требований
3.2	Обновить документацию в соответствии с новыми требованиями	24.02.2023	А.В. Кривов	Техническое задание, руководство пользователя, руководство программиста
3.3	Разработать Приложение	01.03.2023	А.В. Кривов	Итоговый вариант Приложения
3.4	Протестировать Приложение	20.03.2023	А.В. Кривов	Приемосдаточные испытания Приложения, План тестирования
3.5	Провести приемосдаточные мероприятия Приложения	26.03.2023	А.В. Кривов	Документ о принятии работ

ПРИЛОЖЕНИЕ В.
Руководство оператора

ПРАВИТЕЛЬСТВО РОССИЙСКОЙ ФЕДЕРАЦИИ
НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ
«ВЫСШАЯ ШКОЛА ЭКОНОМИКИ»

Факультет социально-экономических и компьютерных наук Образовательная программа бакалавриата «Программная инженерия»

СОГЛАСОВАНО	УТВЕРЖДАЮ
Руководитель проекта, преподаватель кафедры ИТБ НИУ ВШЭ	Академический руководитель образовательной программы «Программная инженерия»
_____ Г.И. Рустамханова	_____ В.В. Ланин
«___» _____ 2023 г.	«___» _____ 2023 г.

Подп. и дата	
Инв. № дубл	
Взаим. инв. №	
Подп. и дата	
Инв. № подл.	RU.17701729.05.15-01 РО 01-1

ИНФОРМАЦИОННАЯ СИСТЕМА ДЛЯ ПЕРСОНАЛА МАГАЗИНА КОМПЬЮТЕРНЫХ КОМПЛЕКТУЮЩИХ

Руководство оператора

ЛИСТ УТВЕРЖДЕНИЯ

RU.17701729.05.15-01 РО 01-1-ЛУ

Исполнитель:
студент группы ПИ21-1
_____ А.В. Кривов
«____» _____ 2023 г.

Пермь 2023

УТВЕРЖДЁН
RU.17701729.05.15-01 РО 01-1-ЛУ

ИНФОРМАЦИОННАЯ СИСТЕМА ДЛЯ ПЕРСОНАЛА МАГАЗИНА КОМПЬЮТЕРНЫХ КОМПЛЕКТУЮЩИХ

Руководство оператора

RU.17701729.05.15-01 РО 01-1

Листов 7

Инв. № подл.	*Подп. и дата*	*Взаим. инв. №*	*Инв. № дубл*	*Подп. и дата*
RU.17701729.10.03-01 РО 01-1				

Пермь 2023

1 Назначение программы

Наименование программы: «Информационная система для персонала магазина компьютерных комплектующих».

Программа предназначена для функционирования на ЭВМ клиента, имеет клиентскую часть и базу данных на удаленном сервере.

Клиентская часть должна быть реализована в виде десктопного приложения с пользовательским интерфейсом.

Приложение должно предоставлять следующие возможности:

- управление персоналом;
- управление складами;
- управление комплектующими;
- управление складским учетом;
- управление заказами;
- управление поставками и поставщиками.

Входные данные со стороны пользователя должны вводиться последовательно с использованием элементов интерфейса.

Выходные данные организованы в виде таблиц.

2 Условия выполнения программы

Для функционирования приложения необходима система, поддерживающая функционирование приложений платформы .NET 7.

Программа должна функционировать в любой среде, в которой возможно функционирование используемого устройства.

Очное обслуживание не требуется, характер выполняемых задач не требует регулярных обновлений. Программа не должна терять функционал при отсутствии обновлений, за исключением вопросов оптимизации времени/памяти.

Для пользования программой необходимы базовые знания по работе с ЭВМ.

3 Выполнение программы

Перед запуском программы необходимо поместить исполняемый файл в удобную для пользователя директорию.

После запуска приложения открывается окно интерфейса. Пользователю предоставляется возможность ввести логин (идентификатор сотрудника) и пароль для авторизации. В случае ввода неправильных данных пользователю необходимо будет ввести их заново. Если данные корректны, то окно авторизации закрывается и открывается главное меню.

В главном меню пользователь может выбрать любую доступную вкладку для взаимодействия с определенной таблицей из базы данных. Набор инструментов зависит строго от группы прав пользователя. Максимальная группа прав предоставляет полный CRUD-функционал для всех таблиц. Данные из каждой таблицы можно экспортировать в Excel-файл формата xlsx.

4 Сообщения оператору

Существуют различные сообщения, выдаваемые в ходе выполнения программы. Описание их содержания и соответствующие действия операторы будут приведены ниже:

1. При запуске программы возможно появление ошибки о невозможности подключения к базе данных. Для решения этой проблемы необходимо обратиться к системному администратору.
2. При вводе неверных данных в окне авторизации пользователь получает ошибку входа. Для авторизации необходимо ввести корректные данные.
3. При построении графика (визуализация данных) может появиться сообщение, что недостаточно данных. Для этого нужно добавить в базу данных информацию о необходимой сущности.