МИНИСТЕРСТВО ОБРАЗОВАНИЯ КИРОВСКОЙ ОБЛАСТИ

Кировское областное государственное профессиональное образовательное

бюджетное учреждение

«Слободской колледж педагогики и социальных отношений»

**ОТЧЕТ**

**по учебной практике**

**ПМ01. Разработка модулей программного обеспечения для компьютерных систем**

**Тема: «Разработка программного модуля «Автоматизация складского учета»**

Студент

Лутошкин Андрей Юрьевич

Группа 21П-1

Специальность 09.02.07 Информационные системы и программирование

Руководитель практики от колледжа:

*Махнев Александр Анатольевич*

/

Подпись расшифровка

2024-2025 уч. год

**СОДЕРЖАНИЕ**

1. СВЕДЕНИЯ ОБ ОРГАНИЗАЦИИ (БАЗЕ ПРАКТИКИ)
2. РАЗРАБОТКА И АНАЛИЗ ТЕХНИЧЕСКОГО ЗАДАНИЯ
3. РАЗРАБОТКА ПРОГРАММНОГО МОДУЛЯ ПО АЛГОРИТМУ В СООТВЕТСТВИИ ТЕХНИЧЕСКОМУ ЗАДАНИЮ
4. ОФОРМЛЕНИЕ ДОКУМЕНТАЦИИ НА ПРОГРАММНЫЙ МОДУЛЬ
5. ВЫПОЛНЕНИЕ ОТЛАДКИ ПРОГРАММНОГО МОДУЛЯ
6. РАЗРАБОТКА ПРОГРАММНОГО МОДУЛЯ ДЛЯ МОБИЛЬНОГО УСТРОЙСТВА
7. ПРОВЕРКА РАБОТОСПОСОБНОСТИ ПРОГРАММНОГО МОДУЛЯ НА МОБИЛЬНОМ УСТРОЙСТВЕ ИЛИ ЭМУЛЯТОРЕ
8. ЗАКЛЮЧЕНИЕ.
9. ПРИЛОЖЕНИЯ К ОТЧЕТУ.
10. **СВЕДЕНИЯ ОБ ОРГАНИЗАЦИИ (БАЗЕ ПРАКТИКИ)**

Наименование базы практики – Кировское областное государственное

профессиональное образовательное бюджетное учреждение «Слободской колледж педагогики и социальных отношений» (КОГПОБУ СКПиСО).

* Адрес – Кировская область, г. Слободской, ул. Рождественская, д. 69.
* ФИО руководителя (директора) – Шеренцова Ольга Михайловна.
* Телефон директора +79014791707.

Образование в КОГПОБУ СКПиСО ориентировано на массовое обучение специалистов с рабочими навыками. Половина учебного времени отводится на практические работы в лабораториях. Это помогает выпускникам органично и быстро включаться в трудовые процессы на новых местах работы.

Схема организационной структуры представлена на рисунке 1.

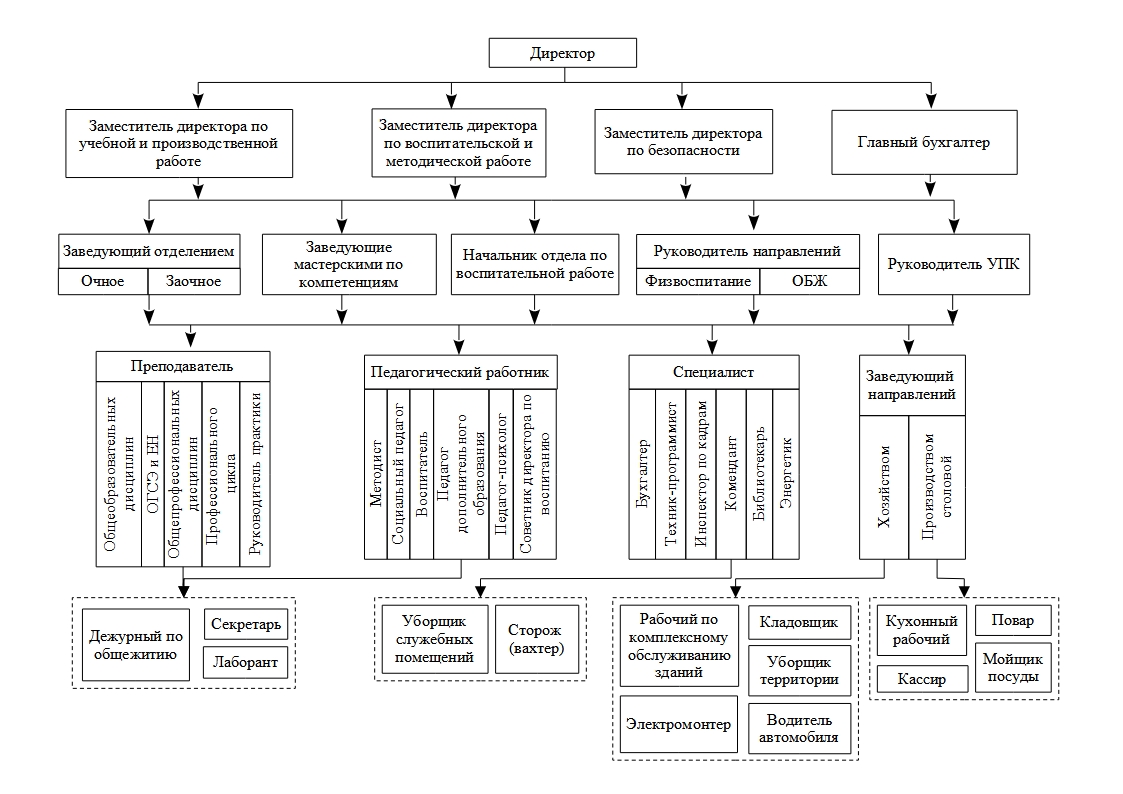


Рисунок 1 - схема организационной структуры

1. **РАЗРАБОТКА И АНАЛИЗ ТЕХНИЧЕСКОГО ЗАДАНИЯ**

На данном этапе было проведено детальное изучение предоставленного технического задания (ТЗ). В рамках анализа ТЗ были уточнены ключевые требования к базе данных, определены основные сущности, их атрибуты и взаимосвязи. Для визуализации процессов и структуры данных были разработаны следующие диаграммы:

**Диаграмма вариантов использования** — для отображения основных сценариев взаимодействия пользователей с системой (рисунок 2.1).

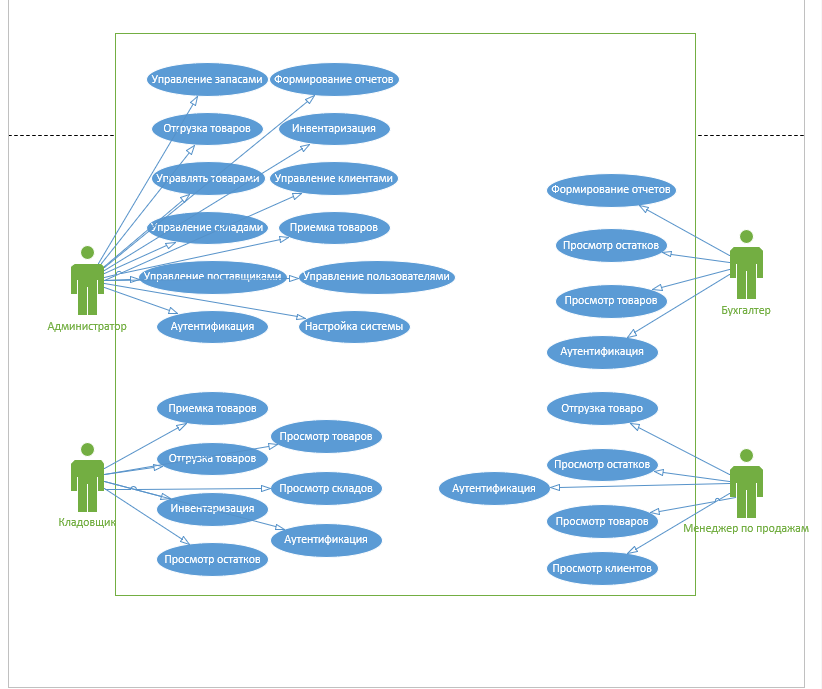


Рисунок 2.1 - Диаграмма вариантов использования

**Диаграмма последовательности** — для демонстрации последовательности действий и взаимодействия между компонентами системы (рисунок 2.2).

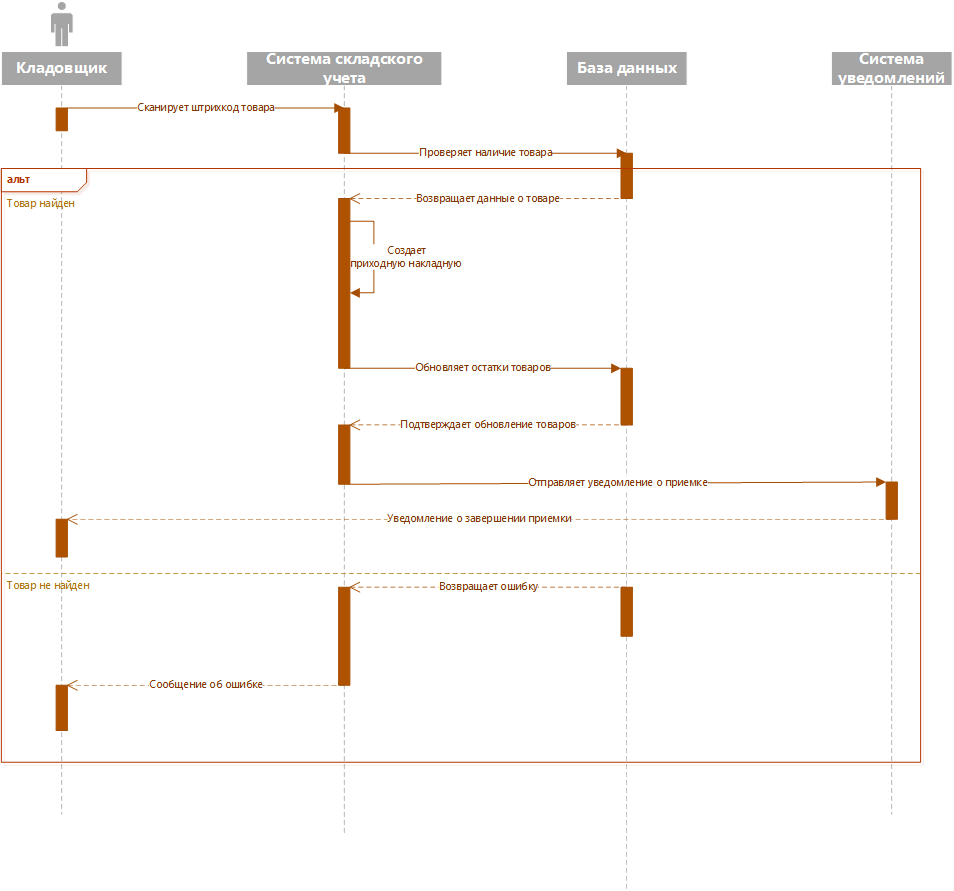


Рисунок 2.2 - Диаграмма последовательности (приемка товаров)

**Диаграмма видов деятельности** — для описания бизнес-процессов и workflow (рисунок 2.3).

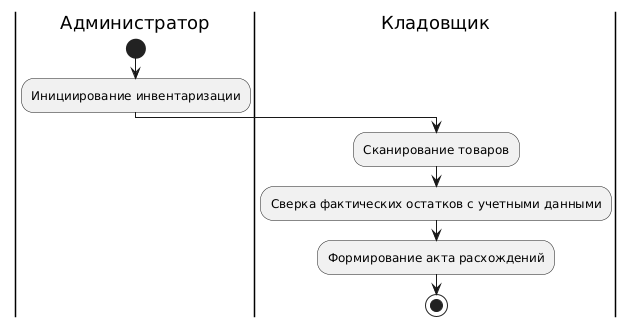


Рисунок 2.3 - Диаграмма видов деятельности (инвентаризация)

**ER-диаграмма** — для проектирования структуры базы данных, включая сущности, их атрибуты и связи (рисунок 2.4).

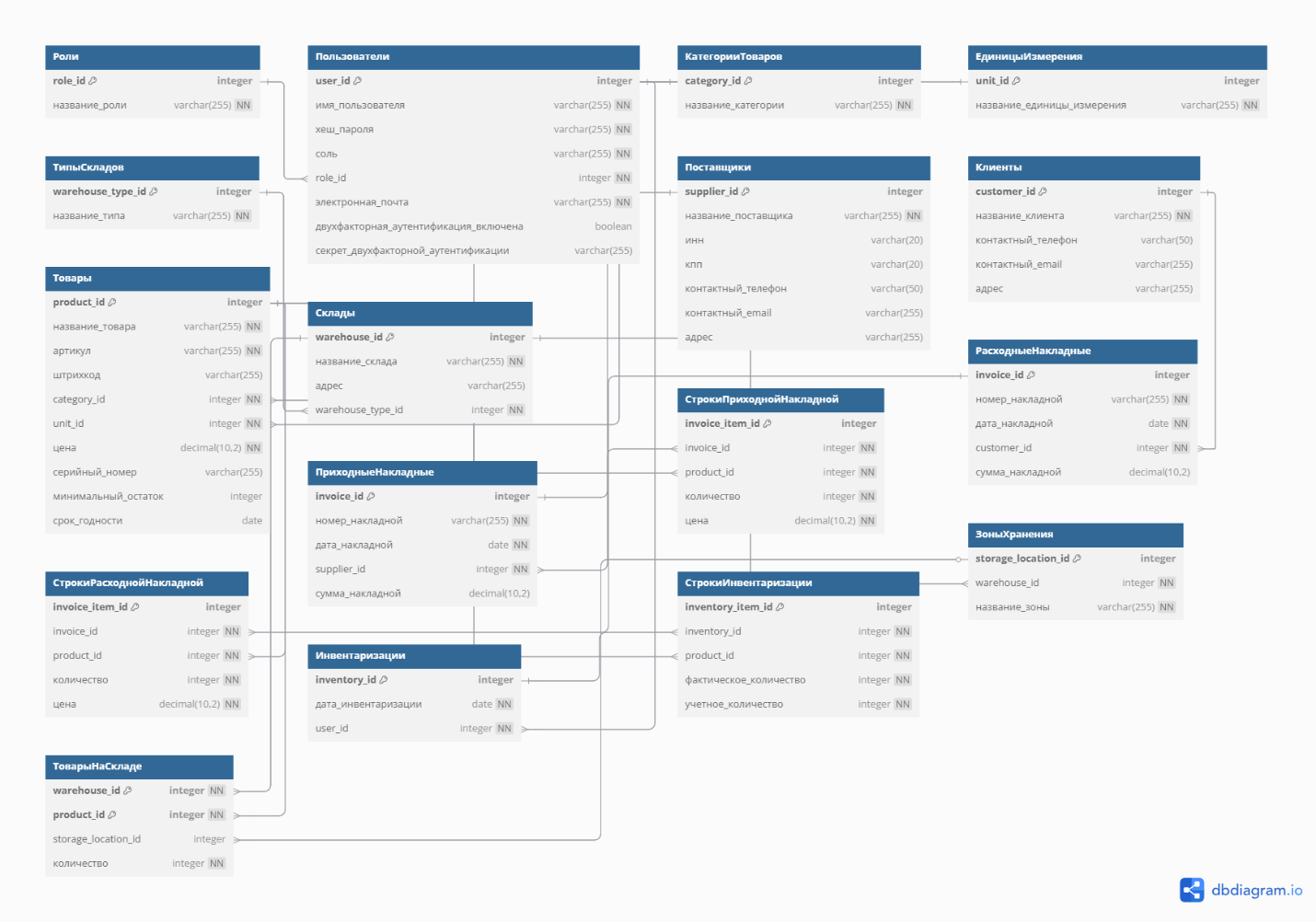


Рисунок 2.4 - Диаграмма базы данных

На основе анализа ТЗ и разработанных диаграмм были сформулированы уточненные требования к реализации базы данных. Также был произведен выбор СУБД (MSServer) и подготовлена основа для следующего этапа — непосредственного создания базы данных.

1. **РАЗРАБОТКА ПРОГРАММНОГО МОДУЛЯ ПО АЛГОРИТМУ В СООТВЕТСТВИИ ТЕХНИЧЕСКОМУ ЗАДАНИЮ**

На данном этапе была выполнена разработка программного модуля, включающего API, библиотеку для работы с данными и настольное приложение, в строгом соответствии с техническим заданием (ТЗ). Основной задачей было реализовать функционал, который обеспечивает корректную работу системы и ее интеграцию с другими компонентами.

**Разработка API**

**Взаимодействие с данными осуществляется напрямую через API, предоставляемый платформой Supabase. API построен на основе стандарта REST и использует JSON для обмена данными. Для взаимодействия с API из кода C# используется библиотека Supabase.Client.**

Были реализованы и используются следующие возможности API Supabase:

* **Авторизация:** Используется встроенная аутентификация Supabase, основанная на JWT (JSON Web Tokens). Поддерживаются следующие способы аутентификации: через email/пароль и через токен сессии. Библиотека Supabase.Client предоставляет удобные методы для управления сессиями пользователей.
* **Получение списка складов:** Для получения списка складов используется метод From<Warehouse>().Get(), предоставляемый Supabase.Client. Этот метод отправляет GET-запрос к соответствующей таблице (warehouses) в Supabase и возвращает данные в виде коллекции объектов C#.
* **Работа с товарами:** Для работы с товарами используются следующие методы Supabase.Client:
  + From<Product>().Insert(product) – для добавления нового товара (отправляет POST-запрос к таблице products). Объект product представляет собой C#-класс, соответствующий структуре таблицы товаров.
  + From<Product>().Get() – для получения списка всех товаров (отправляет GET-запрос к таблице products).

**Важно:** Поскольку API Supabase автоматически генерируется на основе структуры таблиц, отдельные методы API, такие как /warehouses или /products, в коде C# напрямую не используются. Вместо этого взаимодействие происходит через объектно-ориентированную обертку, предоставляемую библиотекой Supabase.Client.

* Supabase.Client.Auth – управление аутентификацией
* Supabase.Client.From<Warehouse>() – взаимодействие с таблицей "Склады"
* Supabase.Client.From<Product>() – взаимодействие с таблицей "Товары"

*(Можно добавить схему, иллюстрирующую поток данных: C# приложение -> Supabase.Client -> Supabase API -> База данных Supabase)*

**Библиотека для работы с данными (Supabase)**

Для упрощения и инкапсуляции логики работы с данными складов была разработана собственная библиотека на языке C#. Эта библиотека *использует* Supabase.Client для взаимодействия с API Supabase, предоставляя более высокоуровневый интерфейс для выполнения специфичных для приложения задач. Библиотека включает следующие методы:

* **Подсчет количества товаров:**
  + CountAllProducts(): Использует Supabase.Client.From<Product>().Count() для получения общего количества товаров во всех складах.
  + CountProductsInWarehouse(int warehouseId): Использует Supabase.Client.From<Product>().Filter("warehouse\_id", Operator.Equals, warehouseId).Count() для получения количества товаров на конкретном складе.
* **Подсчет суммы стоимости товаров:**
  + CalculateTotalProductValue(): Использует Supabase.Client и *пользовательскую функцию (Postgres function)* в Supabase (например, calculate\_total\_product\_value()) для расчета общей стоимости товаров. *Это необходимо, так как Supabase.Client напрямую не поддерживает агрегатные функции, такие как SUM.*
  + CalculateWarehouseProductValue(int warehouseId): Использует Supabase.Client и *пользовательскую функцию* в Supabase (например, calculate\_warehouse\_product\_value(warehouse\_id)) для расчета стоимости товаров на конкретном складе.
* **Подсчет товаров по категориям:**
  + CountProductsByCategory(string category): Использует Supabase.Client.From<Product>().Filter("category", Operator.Equals, category).Count() для подсчета товаров по всем складам.
  + CountProductsInCategoryInWarehouse(string category, int warehouseId): Использует Supabase.Client.From<Product>().Filter("category", Operator.Equals, category).Filter("warehouse\_id", Operator.Equals, warehouseId).Count() для подсчета на конкретном складе.

Методы библиотеки реализованы с использованием перегрузки, где это применимо, для обеспечения гибкости вызова.

**Создание настольного приложения**

Настольное приложение было разработано с использованием WPF для C#. Основные функции приложения:

* Разграничение по ролям — реализованы роли Администратора, Кладовщика, Менеджера по продажам и Бухгалтера с соответствующими правами доступа:
* Администратор — управление складами, товарами, клиентами, поставщиками и пользователями системы.
* Кладовщик — оформление приходных и расходных накладных, отправка отчетов по инвентаризации, формирование штрихкодов (QR-кодов) для товаров.
* Менеджер по продажам — формирование заказов поставщикам и клиентам, просмотр содержимого складов, контроль накладных и заказов, регистрация клиентов и поставщиков.
* Бухгалтер — формирование отчетов по складам (сумма, количество, категория), отчетов по оборотам и остаткам, проведение инвентаризации.
* Функции поиска и сортировки — реализованы для работы с данными (заказы, накладные, товары).
* Подключение к базе данных — приложение работает через подключение к БД по технологии ADO NET.EDM.
* Профиль пользователя — реализовано окно профиля с возможностью редактирования данных пользователя.

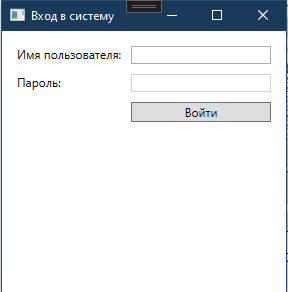


Рисунок 3.1 – Настольное приложение

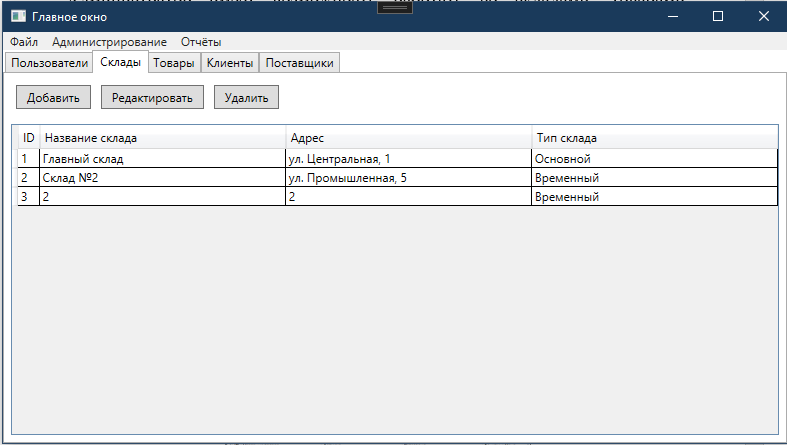


Рисунок 3.2 – Настольное приложение (окно Администратора)

Результатом этапа стал полностью функциональный программный модуль, включающий API, библиотеку для работы с данными и настольное приложение, соответствующие всем требованиям ТЗ. Модуль готов к дальнейшему тестированию и отладке.

1. **ОФОРМЛЕНИЕ ДОКУМЕНТАЦИИ НА ПРОГРАММНЫЙ МОДУЛЬ**

На данном этапе была выполнена разработка и оформление документации на программный модуль в соответствии с требованиями технического задания (ТЗ) и стандартами разработки. Документация включает в себя описание всех компонентов системы, инструкции по использованию и технические спецификации.

**Руководство оператора для настольного приложения**

**Функциональным назначением программы** является автоматизация складских процессов.

Программа должна обеспечивать возможность выполнения перечисленных ниже функций:

* Двухфакторную аутентификацию и разграничение по ролям: Администратор, Кладовщик, Менеджер по продажам, Бухгалтер.
* Управление складами, товарами, клиентами, поставщиками и пользователями.
* Оформление накладных, отчеты по инвентаризации, генерация штрихкодов.
* Формирование заказов, просмотр складов, управление клиентами и поставщиками.
* Формирование отчетов по складам, оборотам, остаткам и проведение инвентаризации.
* Поиск и сортировка данных: заказы, накладные, товары.
* Профиль пользователя: редактирование данных и смена пароля.

**Условия выполнения программы**

Климатические условия эксплуатации, при которых должны обеспечиваться заданные характеристики, должны удовлетворять требованиям, предъявляемым к техническим средства в части условий их эксплуатации.

**Минимальный состав технических средств**

В состав технических средств должен входить IBM-совместимый персональный компьютер (ПЭВМ), включающий в себя:

* процессор с тактовой частотой, 1 ГГц, не менее;
* оперативную память объемом, 512 Мб, не менее;
* жесткий диск со свободным местом 500 Мб, не менее;
* монитор, с разрешением экрана 1024\*768, не менее;
* компьютерная мышь;
* клавиатура;
* принтер;
* CD – привод.

**Минимальный состав программных средств**

Системные программные средства, используемые программой, должны быть представлены лицензионной локализованной версией операционной системы Windows 7 и выше.

**Требования к персоналу (пользователю)**

Минимальное количество персонала, требуемого для работы программы, должно составлять не менее 2 штатных единиц – пользователь программы, оператор.

Пользователь программы должен обладать практическими навыками работы с графическим пользовательским интерфейсом операционной системы семейства Windows.

**Выполнение программы**

Для установки программы необходимо открыть файл setup.exe от имени администратора с CD–диска. При установке выбрать установку для всех пользователей. После установки ярлыки приложений отобразиться на рабочем столе. Для запуска клиентской программы необходимо открыть ярлык либо файл Автоматизация складского учета.exe в папке установки, после пользователю отобразится окно авторизации программы (Рисунок 1).

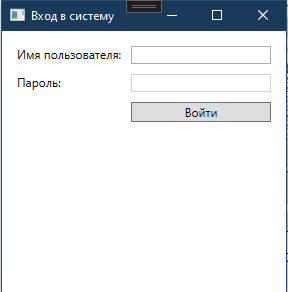


Рисунок 4.1 – Окно авторизации

После ввода вы получаете уведомление об успешной авторизации (Рисунок 2), и производится переход на окно с нужным функционалом, соответствующим вашей роли в системе: Администратор, Бухгалтер, Кладовщик, Менеджер по продажам.

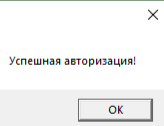


Рисунок 4.2 – Уведомление об успешной авторизации

**Функционал Администратора**

Для администратора доступны следующие функции:

Администратор имеет возможность работать со складами, товарами, клиентами, поставщиками и пользователями системы (редактирование, удаление, добавление). Также реализована функция поиска нужной информации на каждой вкладке. (Рисунок 3-7).

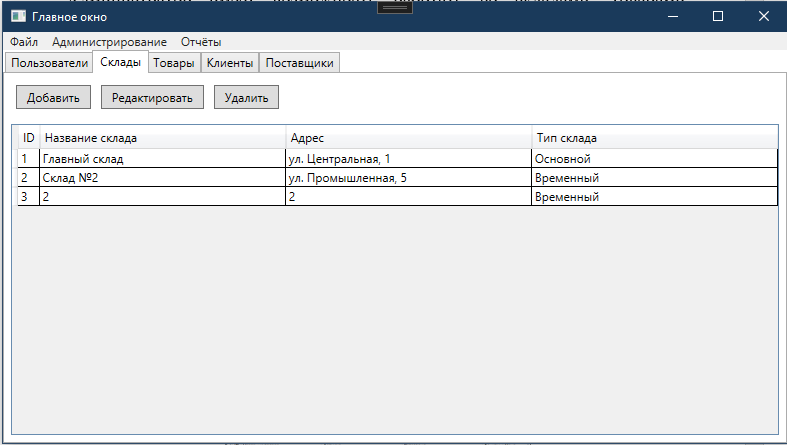


Рисунок 4.3 – Раздел администрирования складами

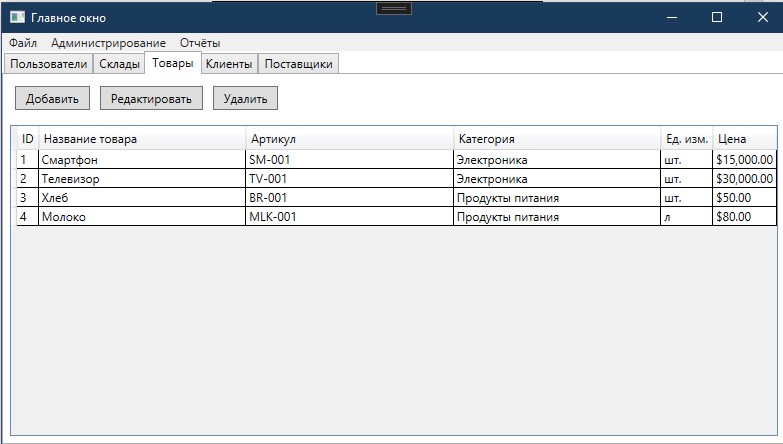


Рисунок 4.4 – Раздел администрирования товарами

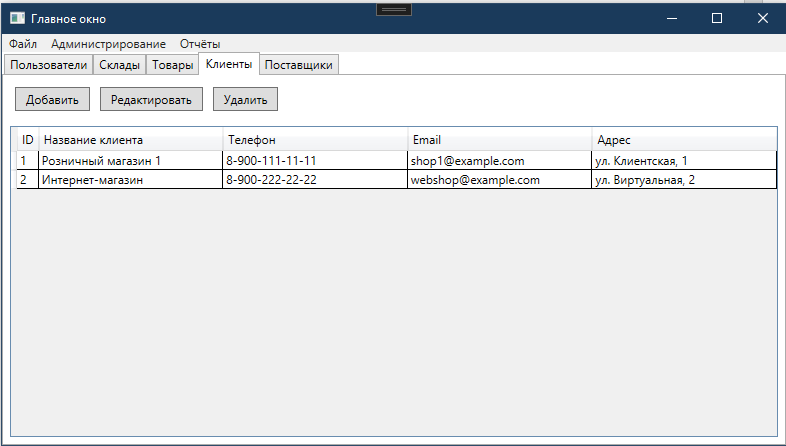


Рисунок 4.5 – Раздел администрирования клиентами

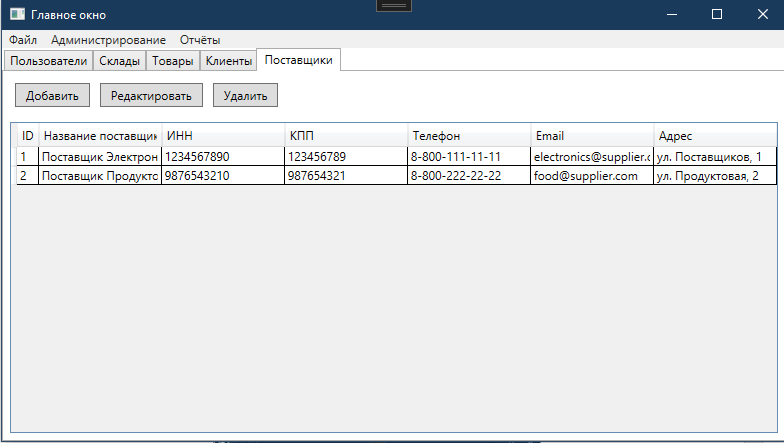


Рисунок 4.6 – Раздел администрирования поставщиками

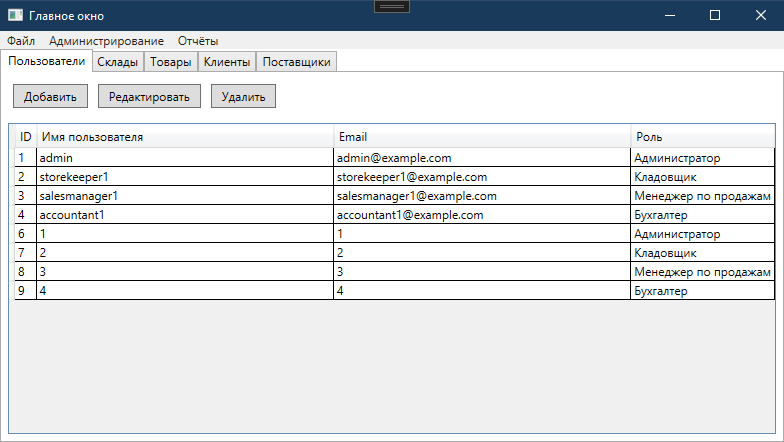


Рисунок 4.7 – Раздел администрирования пользователями

**Функционал Кладовщика**

Для кладовщика доступны следующие функции:

Кладовщик имеет возможность работать с накладными (расходными, приходными), формировать отчет по инвентаризации (экспорт в Excel), формирование штрих-кода для товара и вывод его на печать. (Рисунок 8-10).

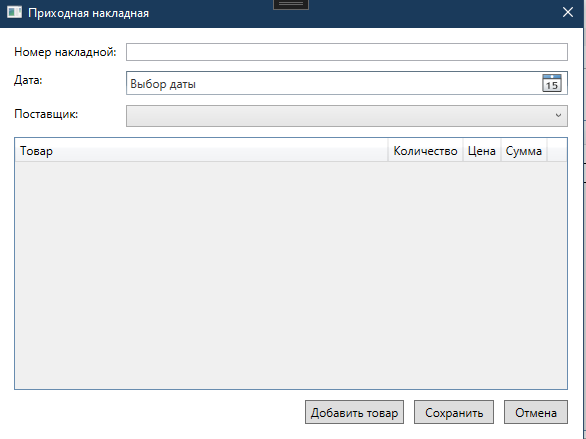


Рисунок 4.8 – Оформление приходной накладной

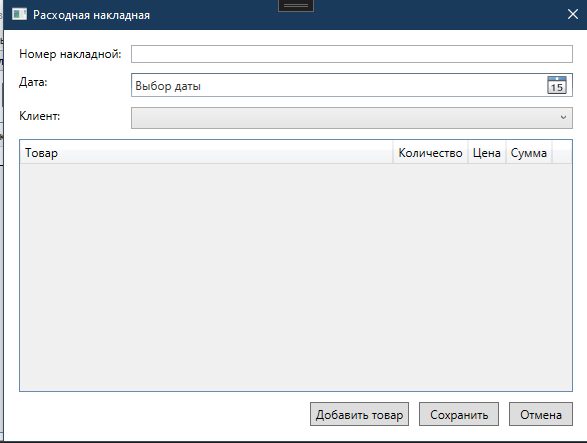


Рисунок 4.9 – Оформление расходной накладной

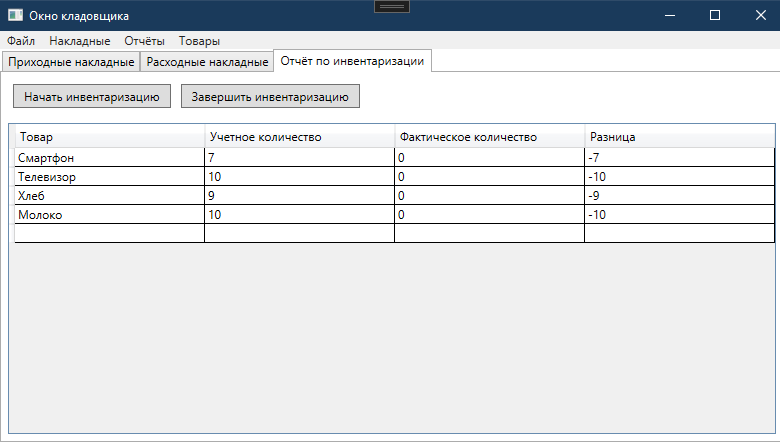


Рисунок 4.10 – Оформление отчета по инвентаризации

**Функционал Бухгалтера**

Для бухгалтера доступны следующие функции:

Бухгалтер имеет возможность формировать отчеты по инвентаризации (проверка фактических и ожидаемых значений склада, вывод расхождений), по складам (сумма, количество, категория), по оборотам, по остаткам. (Рисунок 12-15).

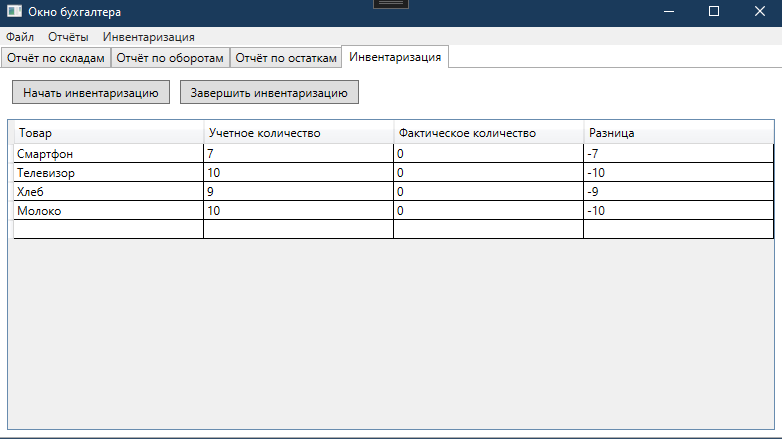


Рисунок 4.11 – Отчет по инвентаризации

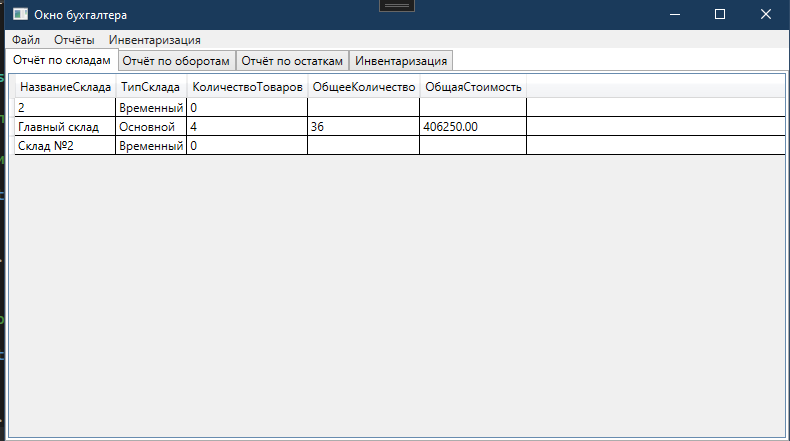


Рисунок 4.12 – Отчет по складам

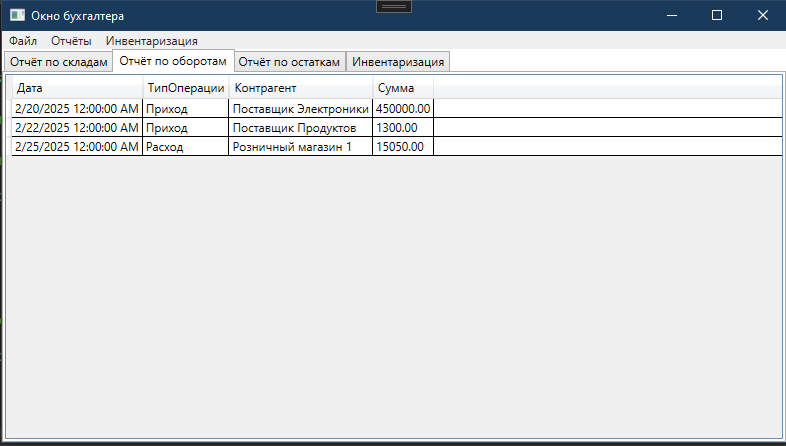


Рисунок 4.13 – Отчет по оборотам

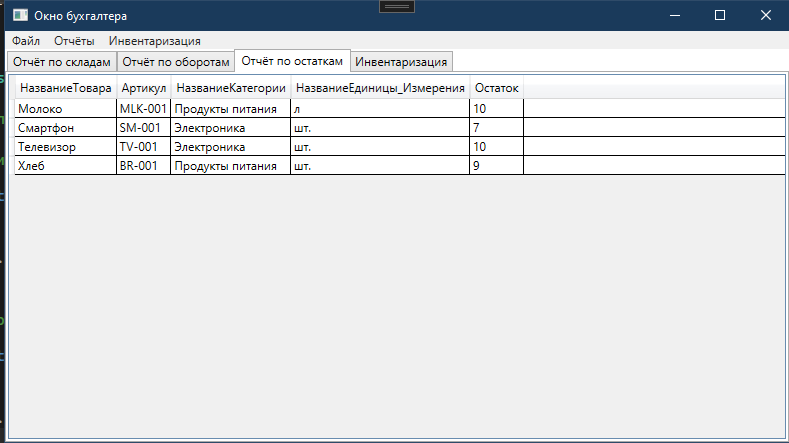


Рисунок 4.14 – Отчет по остаткам

**Функционал Менеджера по продажам**

Для менеджера по продажам доступны следующие функции:

Менеджер по продажам имеет возможность формировать заказы и контролировать их (клиенту, поставщику), просматривать и контролировать склады, накладные, товары, регистрировать клиентов и поставщиков. (Рисунок 16-23).

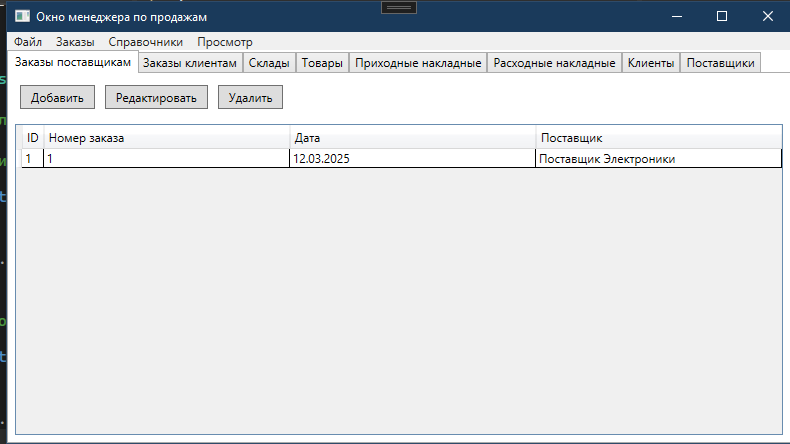


Рисунок 4.15 – Контроль заказов поставщикам

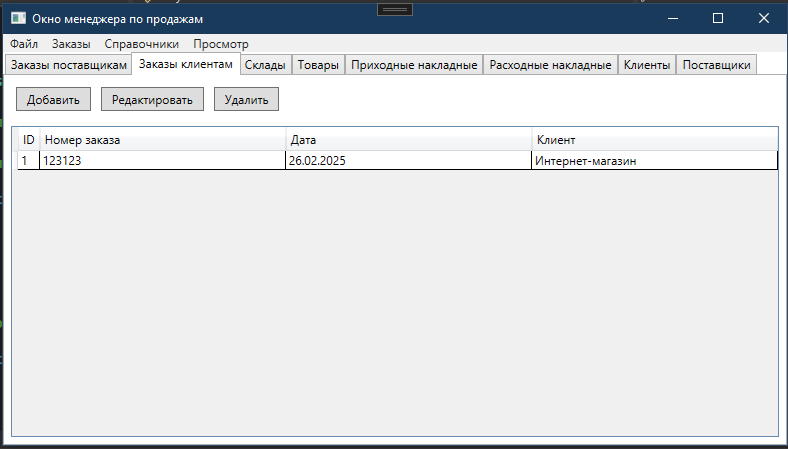


Рисунок 4.16 – Контроль заказов клиентам

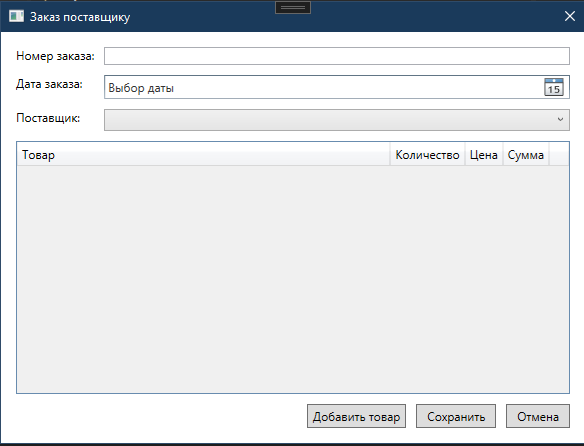


Рисунок 4.17 – Формирование заказа клиенту/поставщику

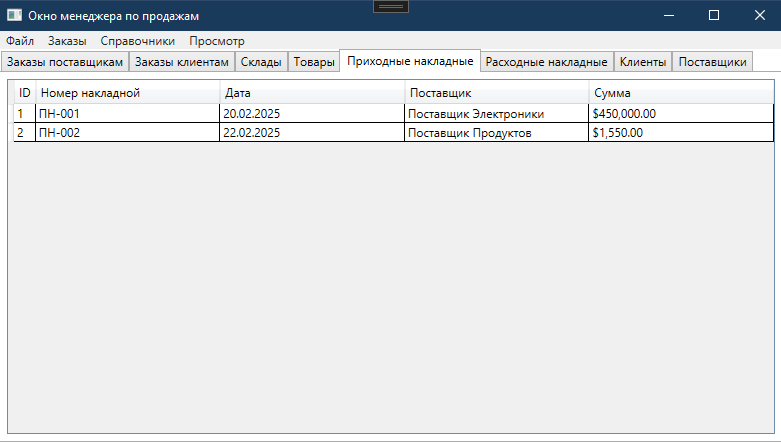


Рисунок 4.18 – Контроль накладных

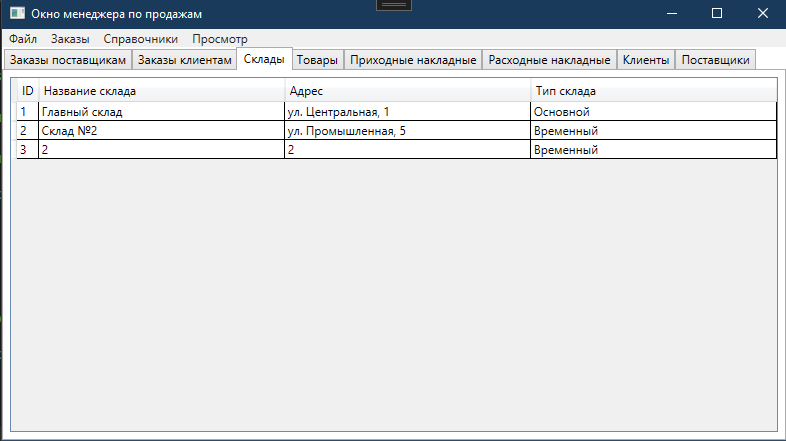


Рисунок 4.19 – Контроль складов

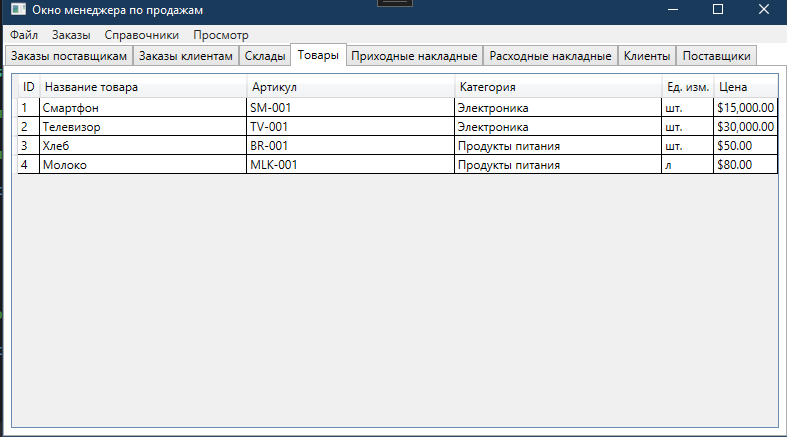


Рисунок 4.20 – Контроль товаров

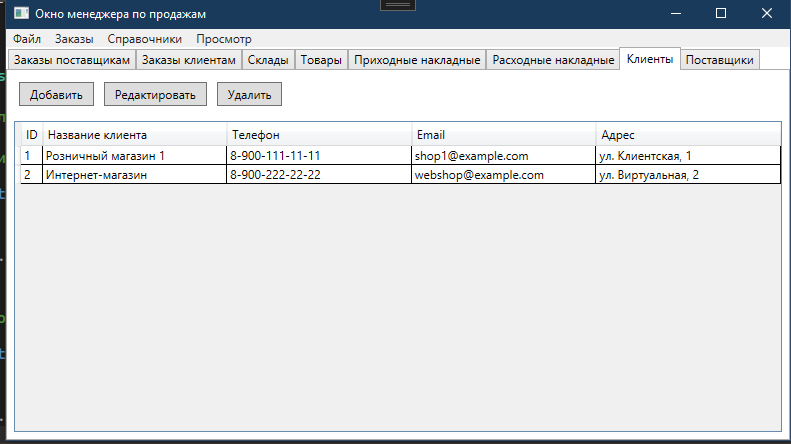


Рисунок 4.21 – Контроль клиентов

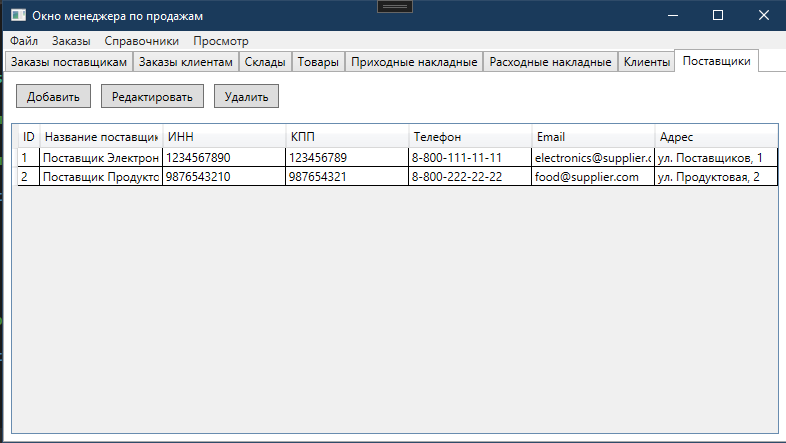


Рисунок 4.22 – Контроль поставщиков

1. **ВЫПОЛНЕНИЕ ОТЛАДКИ ПРОГРАММНОГО МОДУЛЯ**

На данном этапе была проведена отладка программного модуля, в ходе которой были исправлены ошибки в коде, оптимизирована работа API и проверена корректность функционирования всех компонентов системы. Основной задачей было обеспечить стабильную работу модуля в соответствии с требованиями технического задания (ТЗ).

**Основные этапы отладки**

**Исправление ошибок в коде**

Проведен анализ кода на наличие синтаксических, логических и runtime-ошибок.

Устранены ошибки, связанные с обработкой данных, работой с базой данных и взаимодействием между компонентами системы.

Исправлены проблемы, выявленные в процессе тестирования (например, некорректная обработка исключений, ошибки в расчетах и т.д.).

**Оптимизация работы API**

Проведена оптимизация методов API для повышения производительности и уменьшения времени обработки запросов.

Улучшена обработка ошибок (например, при неверных входных данных или отсутствии доступа).

Проверена корректность работы всех методов:

* Авторизация (два способа).
* Получение списка складов.
* Отправка и получение данных о товарах.

**Проверка работы всех функций настольного приложения**

Проведено тестирование функционала настольного приложения для каждой роли:

* Администратор: проверено управление складами, товарами, пользователями и поставщиками.
* Кладовщик: протестировано оформление приходных и расходных накладных, формирование штрихкодов и отправка отчетов по инвентаризации.
* Менеджер по продажам: проверено формирование заказов, регистрация клиентов и поставщиков, а также контроль заказов и накладных
* Бухгалтер: протестировано формирование отчетов по складам, оборотам и остаткам, а также проведение инвентаризации.

Проверена работа функций поиска и сортировки данных (заказы, накладные, товары).

Убедились в корректности работы окна профиля пользователя и возможности редактирования данных.

**Используемые инструменты**

* Отладчики (Visual Studio).
* Логирование — для отслеживания выполнения кода и выявления ошибок.
* Тестовые данные — для проверки всех возможных сценариев.
* Инструменты для тестирования API (Postman).

**Результаты отладки**

* Все выявленные ошибки были успешно исправлены.
* Работа API оптимизирована, что повысило производительность системы.
* Подтверждена корректность работы всех функций приложения.
* Обеспечено соответствие модуля требованиям ТЗ.

Программный модуль был успешно отлажен, все компоненты системы работают стабильно и готовы к дальнейшему тестированию.

1. **РАЗРАБОТКА ПРОГРАММНОГО МОДУЛЯ ДЛЯ МОБИЛЬНОГО УСТРОЙСТВА**

На данном этапе было разработано мобильное приложение, предназначенное в первую очередь для роли кладовщика, но с возможностью просмотра данных другими ролями (Администратор, Менеджер по продажам, Бухгалтер). Приложение предоставляет удобный интерфейс для работы со складами и товарами, а также поддерживает двухфакторную аутентификацию для обеспечения безопасности.

**Основные функции мобильного приложения:**

**Меню приложения**

Приложение включает удобное меню с несколькими фрагментами:

* Список складов и — отображение всех складов.
* Список товаров по складам — просмотр товаров, хранящихся на каждом складе, с возможностью фильтрации и поиска.
* Сканирование штрихкода (QR-кода) — функция для проверки товара путем сканирования штрихкода или QR-кода.
* Профиль с пропуском — раздел для просмотра и редактирования данных пользователя, включая информацию о пропуске.

**Работа с локальной базой данных**

Для хранения данных используется локальная база данных SQLite, которая интегрирована в приложение через NuGet-пакет.

**Технологии и инструменты разработки:**

* Среда разработки: Xamarin (кроссплатформенная разработка для iOS и Android).
* База данных: SQLite (легковесная и производительная локальная БД).

**NuGet-пакеты:**

* SQLite-net-pcl — для работы с SQLite.
* ZXing.Net.Mobile — для сканирования и генерации штрихкодов и QR-кодов.
* Xamarin.Forms.Maps — для работы с картами и геолокацией.

**Особенности реализации:**

* Интерфейс: Приложение имеет интуитивно понятный и отзывчивый интерфейс, адаптированный под мобильные устройства.

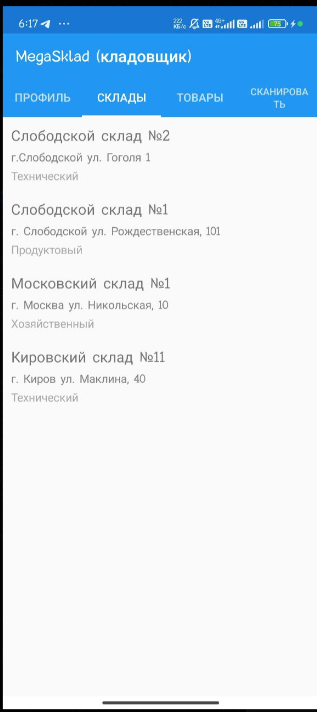


Рисунок 6.1 – Скриншоты интерфейса

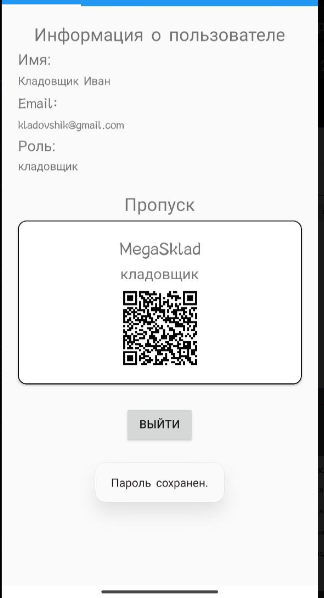


Рисунок 6.2 – Скриншоты интерфейса

* Офлайн-режим: Возможность работы с данными без подключения к интернету благодаря локальной базе данных.
* Сканирование штрихкодов: Реализована функция сканирования с использованием камеры устройства, что упрощает процесс проверки товаров.

Разработано мобильное приложение, которое соответствует всем требованиям технического задания. Обеспечена удобная работа для роли кладовщика, а также возможность просмотра данных другими ролями. Приложение готово к тестированию на реальных устройствах и эмуляторах.

1. **ПРОВЕРКА РАБОТОСПОСОБНОСТИ ПРОГРАММНОГО МОДУЛЯ НА МОБИЛЬНОМ УСТРОЙСТВЕ ИЛИ ЭМУЛЯТОРЕ**

**Функциональным назначением программы** является обеспечение для кладовщиков доступа к информации о складах, товарах сканированию штрих-кодов и управлению профилем.

Программа должна обеспечивать возможность выполнения перечисленных ниже функций:

* Двухфакторная аутентификация;
* Просмотр списка складов и их местоположения;
* Просмотр списка товаров по складам;
* Сканирование штрихкода (QR кода) товара;
* Возможность изменения данных через профиль пользователя;
* Создание пропуска в профиле пользователя.

**Условия выполнения программы**

Климатические условия эксплуатации, при которых должны обеспечиваться заданные характеристики, должны удовлетворять требованиям, предъявляемым к техническим средства в части условий их эксплуатации.

**Минимальный состав технических средств**

Для работы мобильного приложения необходимо устройство, соответствующее следующим минимальным требованиям:

* **Мобильное устройство**: Смартфон или планшет с операционной системой Android 8.0 (или выше) или iOS 12 (или выше).
* **Процессор**: Минимум 4 ядра с тактовой частотой 1,5 ГГц.
* **Оперативная память**: Не менее 2 ГБ.
* **Встроенная память**: Не менее 500 МБ свободного места для установки приложения.
* **Дисплей**: Минимальное разрешение экрана 720x1280 пикселей.
* **Камера**: Наличие камеры для сканирования штрих-кодов и QR-кодов.
* **Интернет-соединение**: Поддержка Wi-Fi или мобильного интернета (3G/4G/5G) для подключения к серверу.
* **Доступ к геопозиции**: Доступ к геопозиции устройства для отображения местоположения складов на карте.

**Минимальный состав программных средств**

Системные программные средства, используемые программой, должны быть представлены лицензионной локализованной версией операционной системы Android 8.0 (или выше) или iOS 12 (или выше).

**Требования к персоналу (пользователю)**

Минимальное количество персонала, требуемого для работы программы, должно составлять не менее 2 штатных единиц – пользователь программы, оператор.

Пользователь программы должен обладать базовыми навыками работы с мобильными устройствами и операционными системами Android или iOS.

**Выполнение программы**

Для установки программы необходимо открыть файл setup.apk на мобильном устройстве, нажмите кнопку "Установить".

После завершения установки ярлык приложения появится на рабочем столе.

Для запуска программы необходимо открыть ярлык либо файл WarehouseMob.apk в папке установки, после пользователю отобразиться окно авторизации (Рисунок 1).

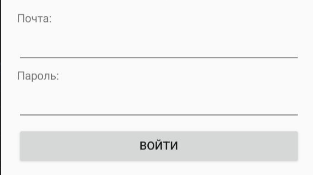


Рисунок 7.1 – Окно авторизации

Далее после успешной авторизации открывается меню навигации по разделам приложения. (Рисунок 2).



Рисунок 7.2 – Навигация

Первая страница представляет собой профиль пользователя с возможностью редактирования данных и генерацией пропуска (QR-кода) (Рисунок 3)

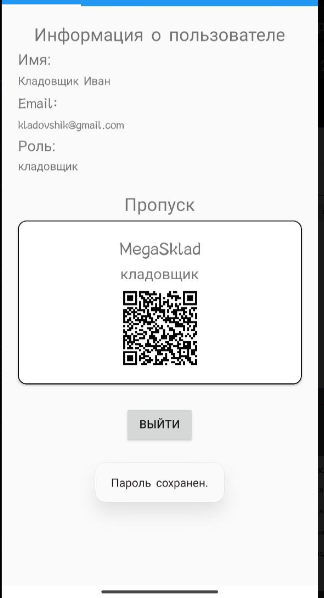


Рисунок 7.3 – Профиль пользователя

Вторая страница в разделе навигации представляет список складов с подробной информацией (Рисунок 4).

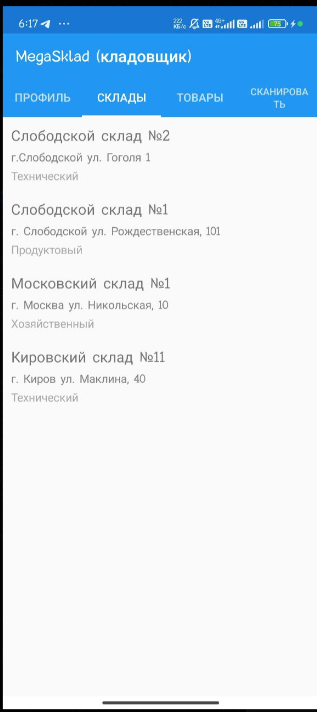


Рисунок 7.4 – Список складов

Третья страница представляет возможность просмотра списка товаров по складам, сам список появляется в виде сплывающего окна. ( Рисунок 5-6)

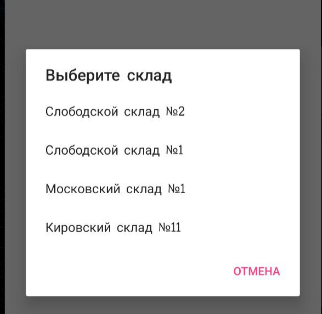


Рисунок 7.5 – Просмотр товаров на складах

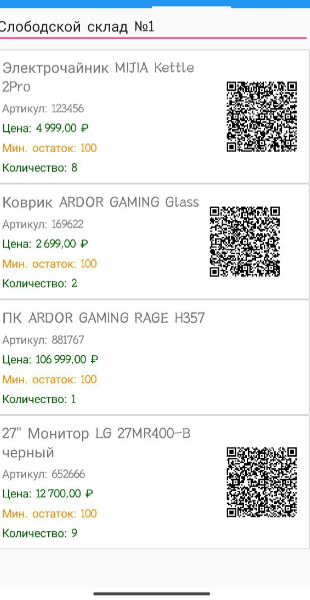


Рисунок 7.6 –Окно с информацией

Четвертая страница представляет возможность сканирования штрих-кодов товаров: как через камеру. (Рисунок 7)



Рисунок 7.7 – Сканирование штрих-кодов

**8. ЗАКЛЮЧЕНИЕ**

Учебная практика предоставила мне ценный опыт, который помог расширить мои профессиональные знания и навыки. В процессе работы над проектом я углубил свои знания в области разработки программного обеспечения, включая проектирование архитектуры системы, работу с базами данных (в частности, с использованием Supabase), создание API и разработку клиентских приложений на C#.

Я освоил современные инструменты и технологии, такие как C#, библиотека Supabase.Client для взаимодействия с Supabase, а также приобрел практический опыт работы с JSON и REST API. Это значительно повысило мою техническую компетентность.

Учебная практика позволила мне применить теоретические знания на практике, что помогло мне лучше понять их применение в реальных рабочих ситуациях. Благодаря работе над практическим проектом я приобрел ценный опыт в решении реальных проблем и разработке работающего решения.

Например, я научился анализировать требования технического задания, проектировать структуру базы данных в Supabase, создавать пользовательские функции (Postgres functions) для сложных вычислений, а также проводить тестирование и отладку программного обеспечения для обеспечения его стабильной работы.

Этот практический опыт дал мне преимущество, поскольку я уже обладаю практическими навыками и опытом, необходимыми для выполнения рабочих задач. Я научился эффективно распределять время и ресурсы, а также находить оптимальные решения для сложных задач. Эти навыки, безусловно, будут полезны в моей будущей профессиональной деятельности.

Я уверен, что полученные знания и опыт помогут мне сделать успешную карьеру в выбранной области. Учебная практика не только укрепила мои технические навыки, но и развила такие важные качества, как ответственность, самостоятельность и умение адаптироваться к новым условиям. Я готов применять эти навыки в реальных проектах и продолжать развиваться как профессионал в сфере информационных технологий.

* + 1. **ПРИЛОЖЕНИЯ К ОТЧЕТУ.**

Все материалы, программные модули и документация, созданные в рамках проекта, были систематизированы и загружены в репозиторий на GitHub.

**Ссылка на репозиторий:**

https://github.com/sv1xt/PracktikaLA4K