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

Кротова Ксения Николаевна

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. ВЫПОЛНЕНИЕ ТЕСТИРОВАНИЯ ПРОГРАММНОГО МОДУЛЯ
7. ОФОРМЛЕНИЕ РЕЗУЛЬТАТОВ ТЕСТИРОВАНИЯ
8. РАЗРАБОТКА ПРОГРАММНОГО МОДУЛЯ ДЛЯ МОБИЛЬНОГО УСТРОЙСТВА
9. ПРОВЕРКА РАБОТОСПОСОБНОСТИ ПРОГРАММНОГО МОДУЛЯ НА МОБИЛЬНОМ УСТРОЙСТВЕ ИЛИ ЭМУЛЯТОРЕ
10. ЗАКЛЮЧЕНИЕ.
11. ПРИЛОЖЕНИЯ К ОТЧЕТУ.
12. **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.

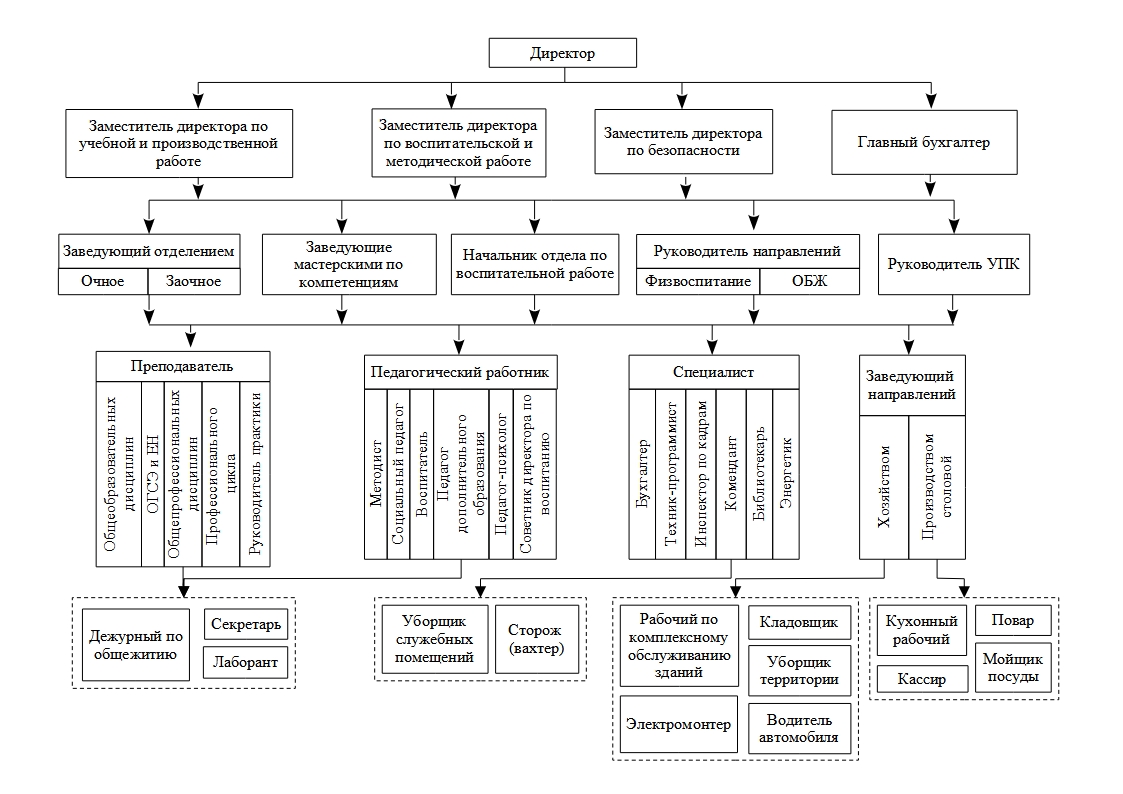


Рисунок 1 - схема организационной структуры

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).

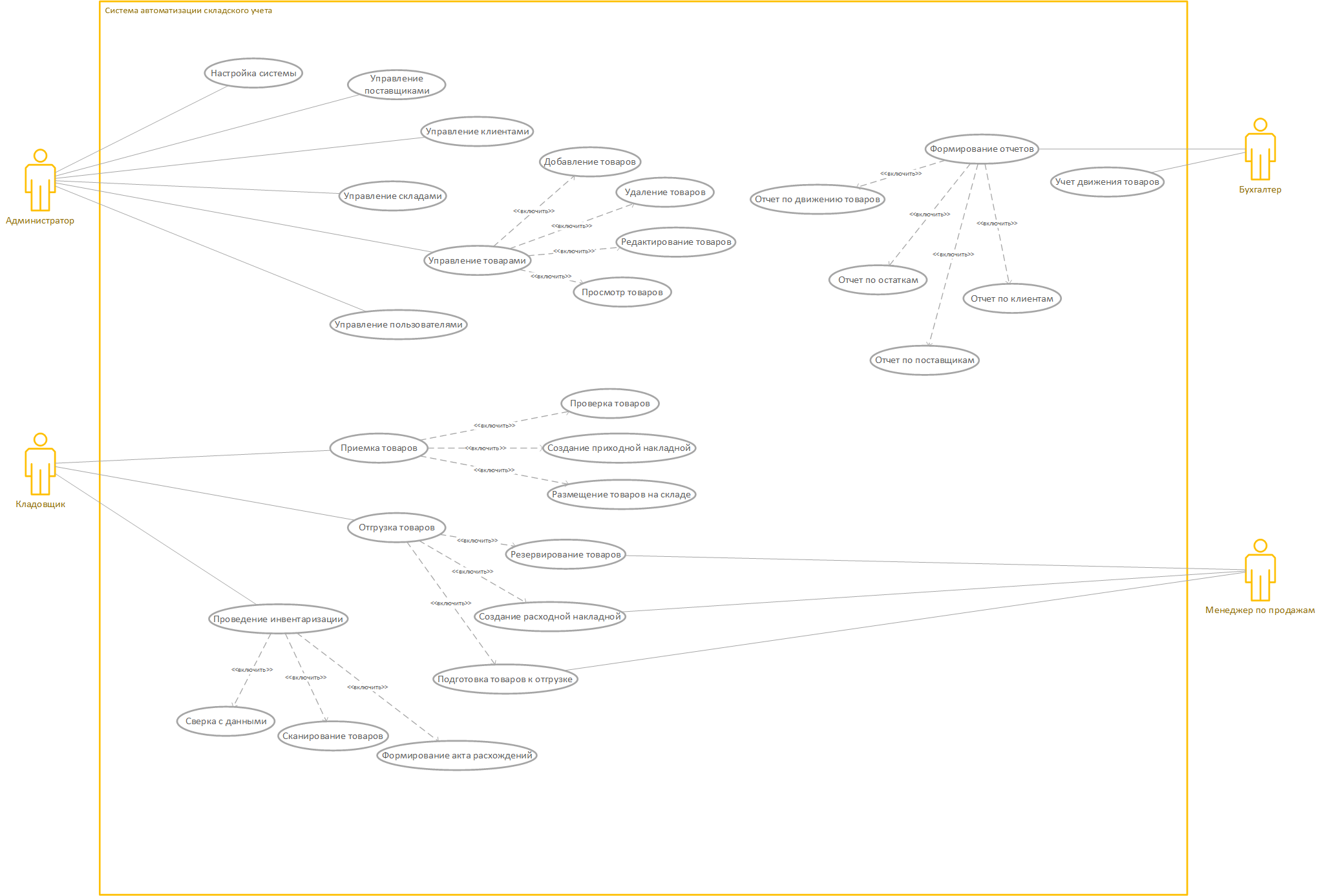


Рисунок 2.1 - Диаграмма вариантов использования

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

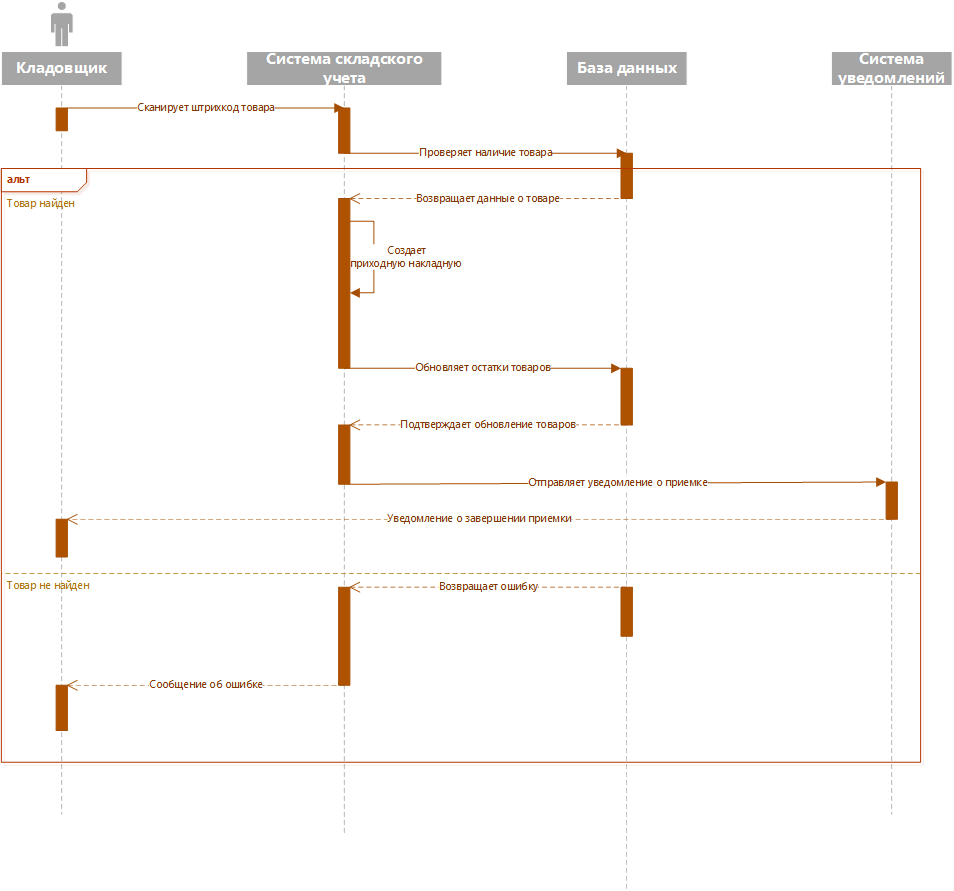


Рисунок 2.2 - Диаграмма последовательности (приемка товаров)

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

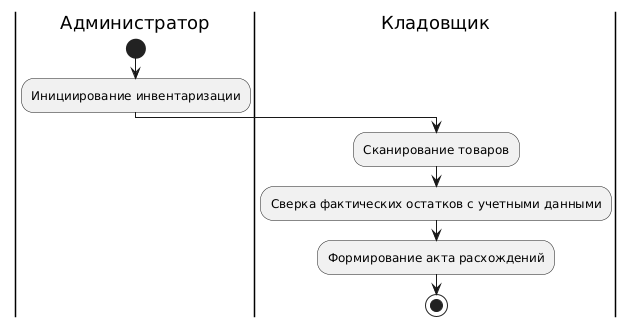


Рисунок 2.3 - Диаграмма видов деятельности (инвентаризация)

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

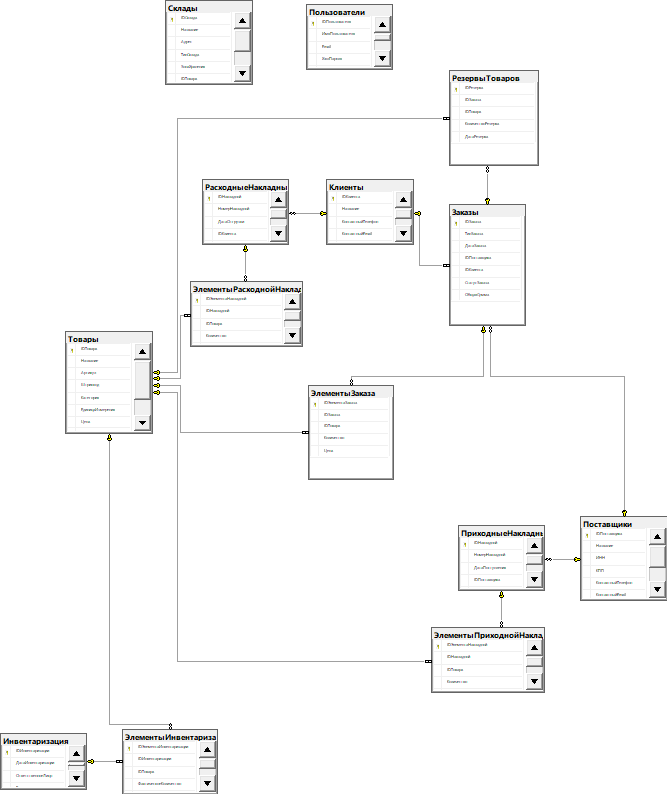


Рисунок 2.4 - Диаграмма базы данных

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**

API был разработан с использованием ASP.NET для C# и с использованием БД по технологии ADO NET.EDM. Были реализованы следующие методы:

* Авторизация — поддержка двух способов аутентификации: через логин/пароль и через токен.
* Получение списка складов — метод GET, возвращающий список всех складов с их основными характеристиками.
* Работа с товарами — методы POST для отправки данных о товаре и GET для получения списка товаров.

API обеспечивает взаимодействие, предоставляя необходимые данные в формате JSON.



Рисунок 3.1 – Методы API



Рисунок 3.2 – Методы API

**Создание библиотеки**

Библиотека для работы с данными со складов была разработана на языке программирования C# и с использованием БД по технологии ADO NET.EDM. Она включает следующие методы:

* Подсчет количества товаров — общее количество товаров по всем складам и на конкретном складе.
* Подсчет суммы стоимости товаров — общая стоимость товаров на складе и по всем складам.
* Подсчет товаров по категориям — количество товаров определенной категории на складе и по всем складам.

Методы библиотеки реализованы с использованием перегрузки, что позволяет гибко работать с различными типами данных.

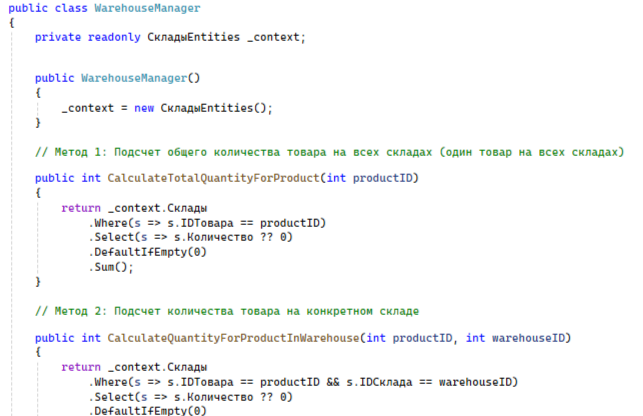


Рисунок 3.3 – Библиотека классов

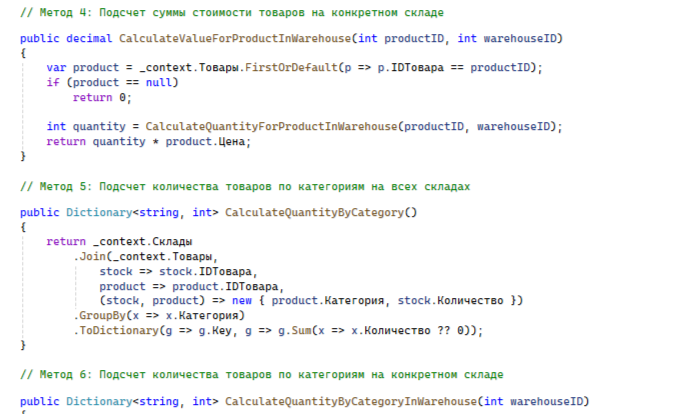


Рисунок 3.4 – Библиотека классов

**Создание настольного приложения**

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

* Двухфакторная аутентификация — для повышения безопасности системы.
* 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.

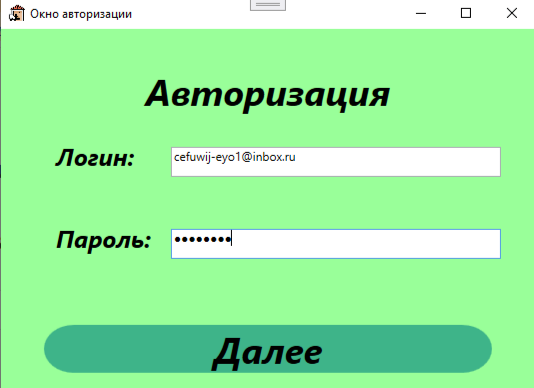


Рисунок 3.5 – Настольное приложение

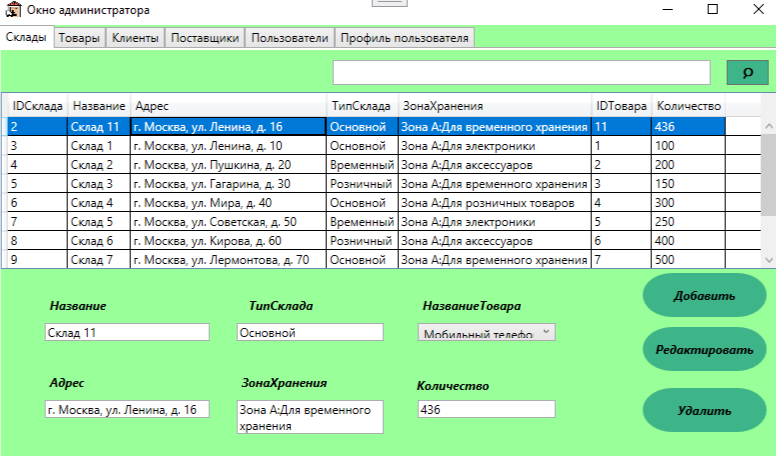


Рисунок 3.6 – Настольное приложение (окно Администратора)

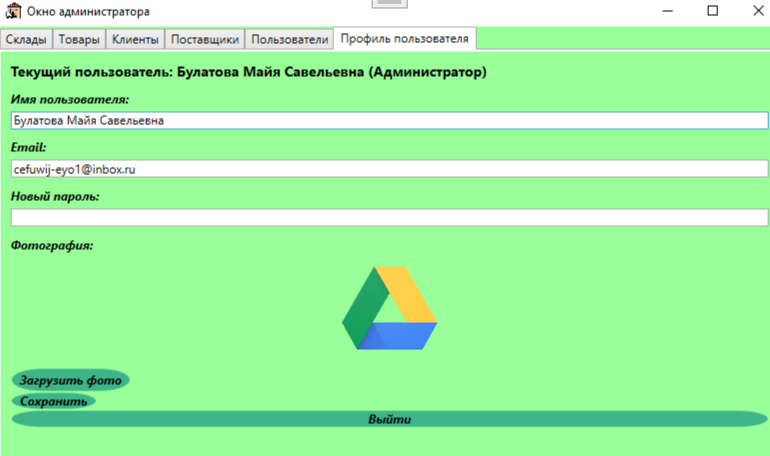


Рисунок 3.7 – Настольное приложение (профиль пользователя)

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 4.1).

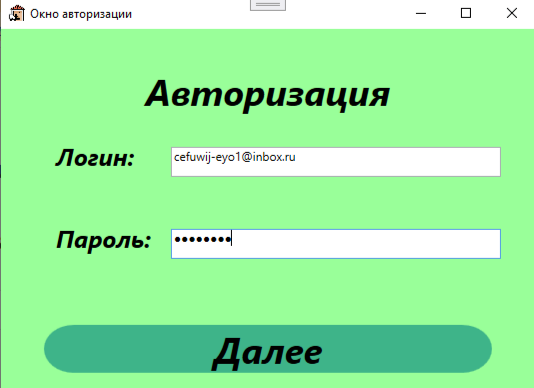


Рисунок 4.1 – Окно авторизации

Для того что бы начать использовать приложение вам необходимо авторизоваться. Для этого, необходимо ввести в поля Логин и Пароль соответствующие данные и нажать на кнопку «Далее». После нажатия на окне авторизации сменятся элементы, и необходимо будет ввести код 2FA для дальнейшей авторизации. (Рисунок 4.2).

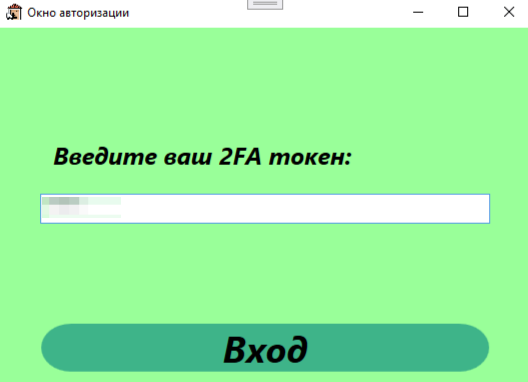


Рисунок 4.2 – Двухфакторная аутентификация

После ввода вы получаете уведомление об успешной авторизации (Рисунок 4.3), и производится переход на окно с нужным функционалом, соответствующим вашей роли в системе: Администратор, Бухгалтер, Кладовщик, Менеджер по продажам.



Рисунок 4.3 – 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 4.4 - 4.8).

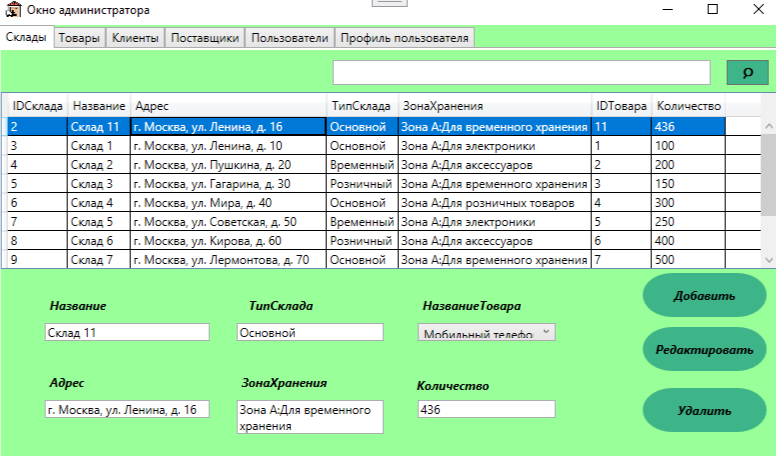


Рисунок 4.4 – Раздел администрирования складами

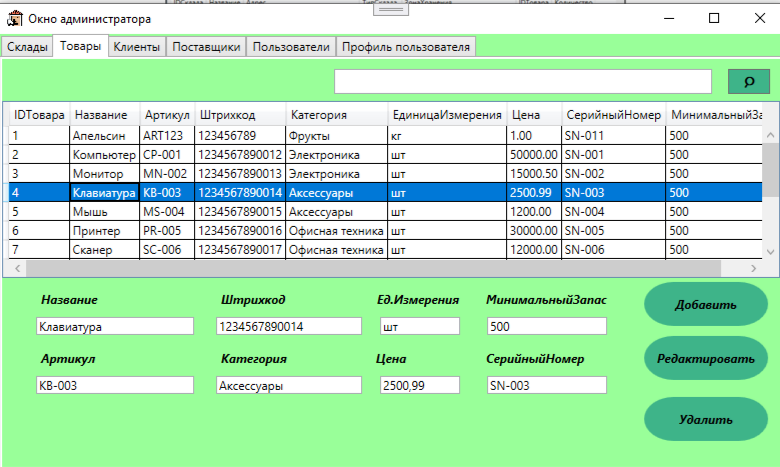


Рисунок 4.5 – Раздел администрирования товарами

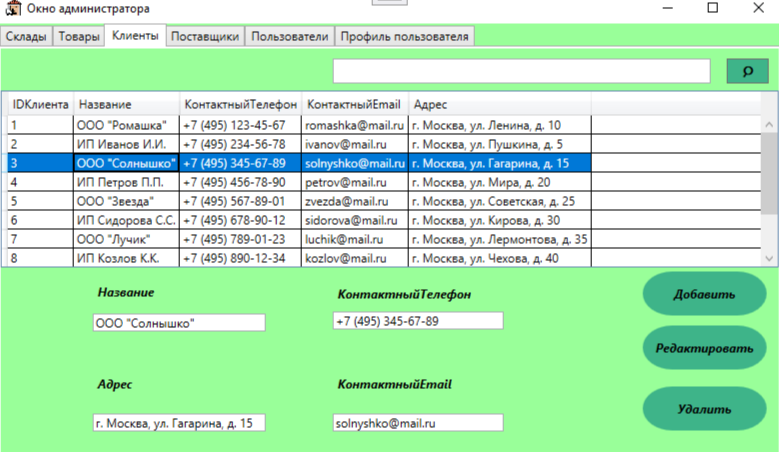


Рисунок 4.6 – Раздел администрирования клиентами

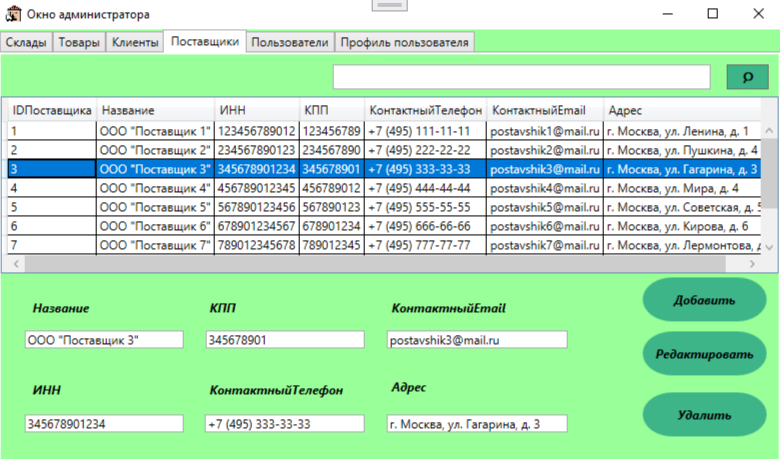


Рисунок 4.7 – Раздел администрирования поставщиками

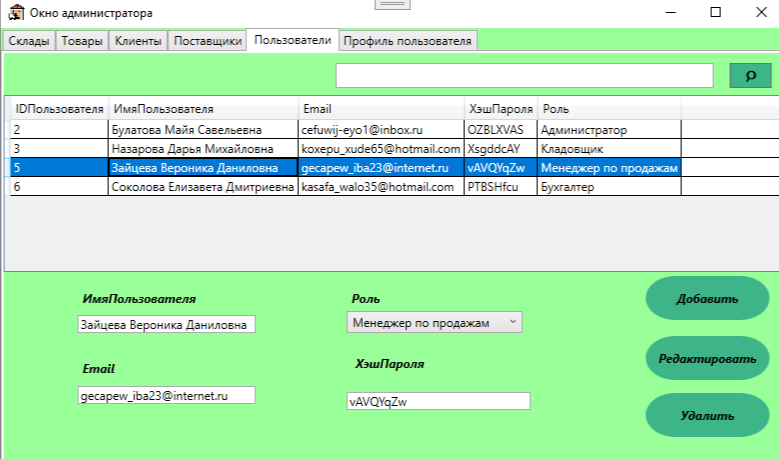


Рисунок 4.8 – Раздел администрирования пользователями

Также реализована возможность редактирования своих данных через профиль пользователя (Рисунок 4.9). Можно изменить пароль, электронную почту, фотографию.

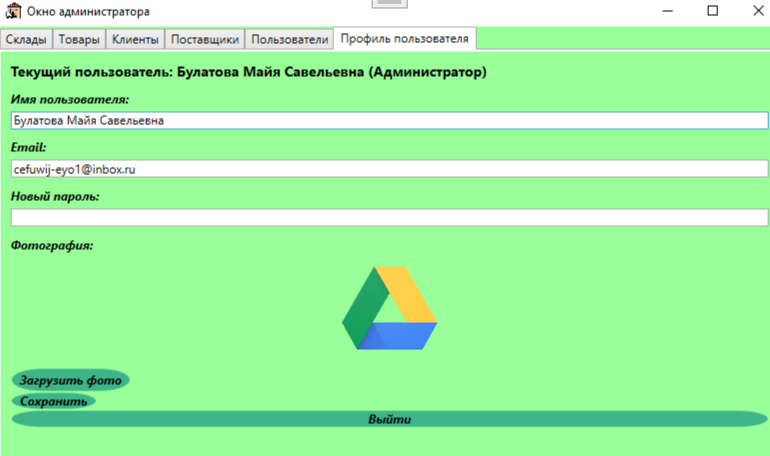


Рисунок 4.9 – Профиль пользователя

**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. Также реализован профиль пользователя. (Рисунок 4.10 – 4.14).

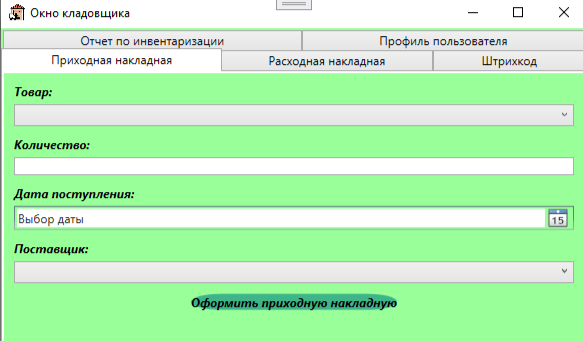


Рисунок 4.10 – Оформление приходной накладной

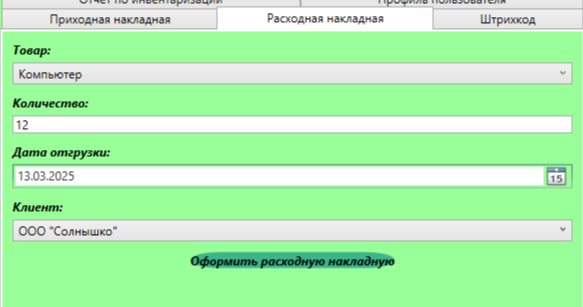


Рисунок 4.11 – Оформление расходной накладной

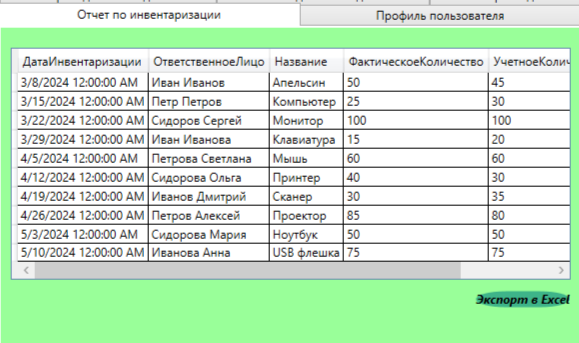


Рисунок 4.12 – Оформление отчета по инвентаризации

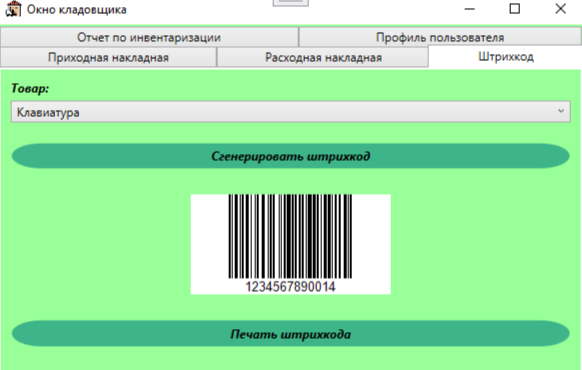


Рисунок 4.13 – Создание штрих-кода

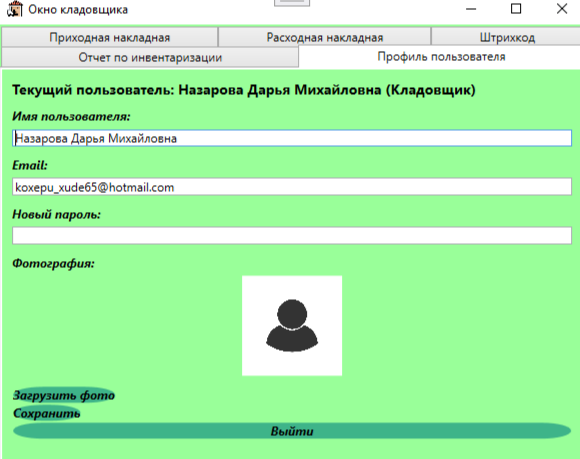


Рисунок 4.14 – Профиль пользователя

**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. Также реализован профиль пользователя. (Рисунок 4.15 – 4.19).



Рисунок 4.15 – Отчет по инвентаризации

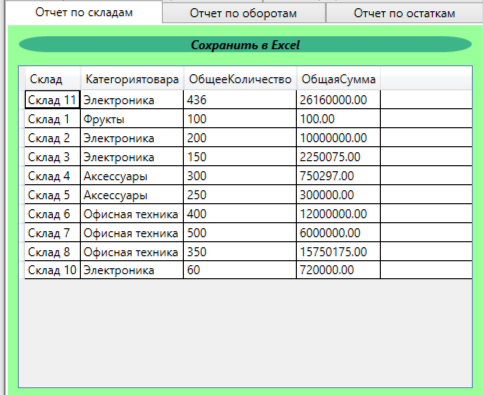


Рисунок 4.16 – Отчет по складам

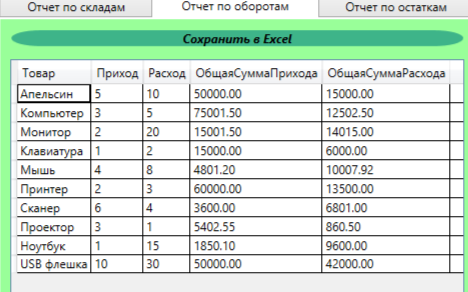


Рисунок 4.17 – Отчет по оборотам



Рисунок 4.18 – Отчет по остаткам

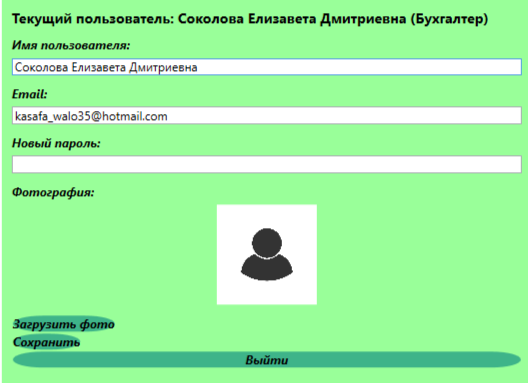


Рисунок 4.19 – Профиль пользователя

**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. Также реализован профиль пользователя возможность поиска информации. (Рисунок 4.20 – 4.27).



Рисунок 4.20 – Контроль заказов

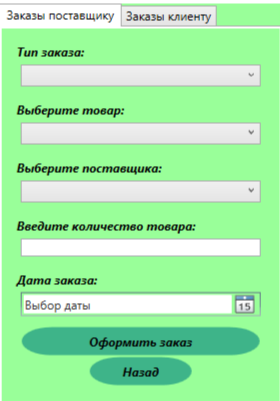


Рисунок 4.21 – Формирование заказа клиенту/поставщику

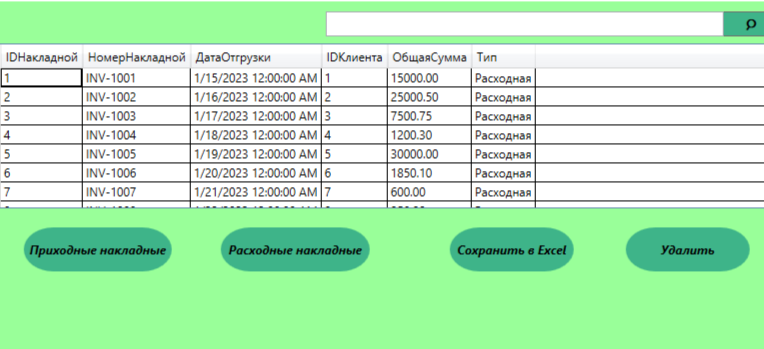


Рисунок 4.22 – Контроль накладных

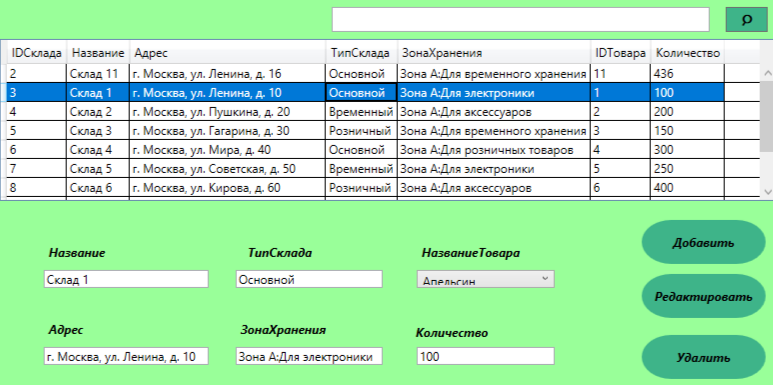


Рисунок 4.23 – Контроль складов

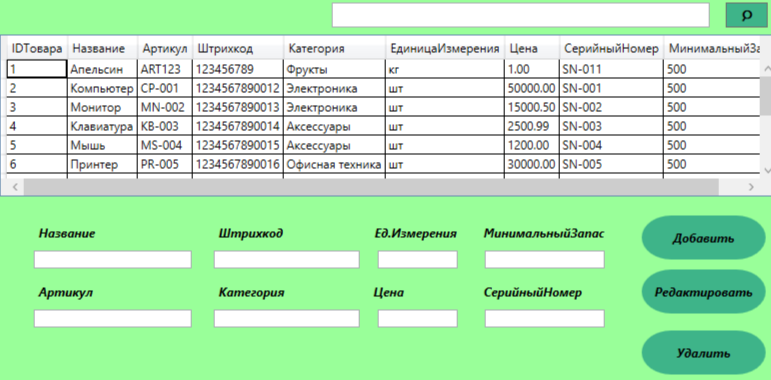


Рисунок 4.24 – Контроль товаров



Рисунок 4.25 – Контроль клиентов

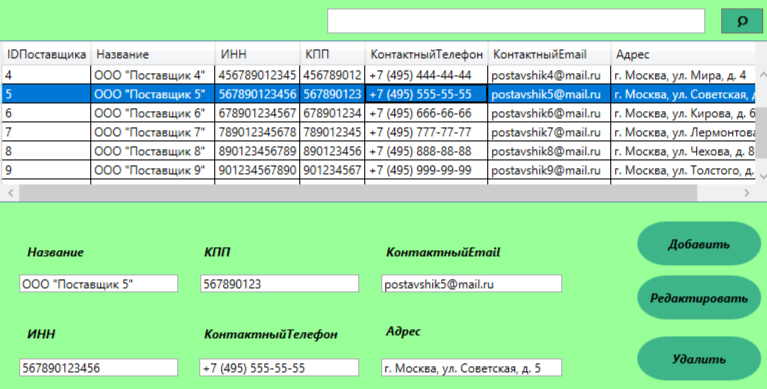


Рисунок 4.26 – Контроль поставщиков

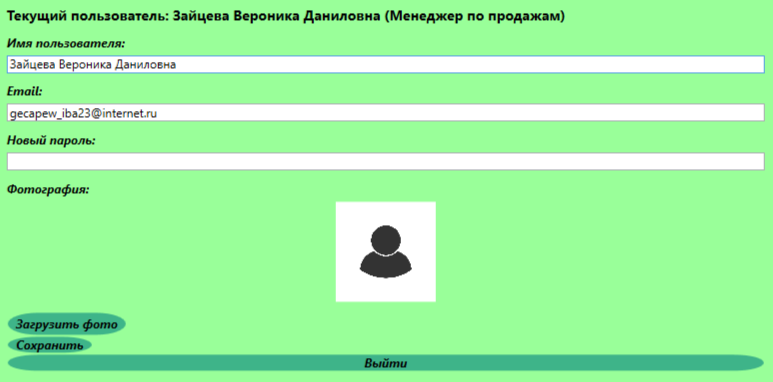


Рисунок 4.27 – Профиль пользователя

**Руководство программиста для API**

Данное руководство предназначено для разработчиков, которые будут использовать API для взаимодействия с системой управления складом и пользователями. API предоставляет методы для аутентификации пользователей, управления товарами и складами.

**Общие сведения**

API использует стандартные HTTP-методы (GET, POST) для выполнения операций. Все запросы и ответы передаются в формате JSON. Базовый URL для всех запросов: <http://localhost:62328/api>

**Аутентификация и авторизация**

**Обычная авторизация**

**Метод: POST**

**URL:** [http://localhost:62328/api**/**auth/login](http://localhost:62328/api/auth/login)

**Тело запроса:**

{

"email": "kasafa\_walo35@hotmail.com",

"password": "PTBSHfcu"

}

**Ответ:**

**Успешная аутентификация:**

{

"IDПользователя": 6,

"ИмяПользователя": "Соколова Елизавета Дмитриевна",

"Email": "kasafa\_walo35@hotmail.com",

"ХэшПароля": "PTBSHfcu",

"Роль": "Бухгалтер",

"Фото": "Строка двоичных данных (из бд)",

"TwoFactorToken": "687737"

}

**Ошибка аутентификации:** 401 Unauthorized

**Запрос токена для 2FA**

**Метод: POST**

**URL:** [http://localhost:62328/api**/**auth/request-2fa](http://localhost:62328/api/auth/request-2fa)

**Тело запроса:**

{

"email": "cefuwij-eyo1@inbox.ru"

}

**Ответ:**

**Успешный запрос:**

{

"Message": "2FA token generated."

}

**Ошибка:** 400 Bad Request или 500 Internal Server Error

**Вход с 2FA**

**Метод: POST**

**URL:** [http://localhost:62328/api**/**auth/login-2fa](http://localhost:62328/api/auth/login-2fa)

**Тело запроса:**

{

"email": "kasafa\_walo35@hotmail.com",

"token": "687737"

}

**Ответ:**

**Успешная аутентификация:**

{

"IDПользователя": 6,

"ИмяПользователя": "Соколова Елизавета Дмитриевна",

"Email": "kasafa\_walo35@hotmail.com",

"ХэшПароля": "PTBSHfcu",

"Роль": "Бухгалтер",

"Фото": "Строка двоичных данных (из бд)",

"TwoFactorToken": "687737"

}

**Ошибка аутентификации:** 401 Unauthorized

**Управление товарами**

**Получение списка товаров**

**Метод: GET**

**URL:** <http://localhost:62328/api/Products>

**Ответ:**

Выводит весь список товаров, хранящихся в базе данных.

**Добавление нового товара**

**Метод: POST**

**URL:** <http://localhost:62328/api/Products>

**Тело запроса:**

{

"IDТовара":

"Название":

"Артикул":

"Штрихкод":

"Категория":

"ЕдиницаИзмерения":

"Цена":

"СерийныйНомер":

"МинимальныйЗапас":

}

**Ответ:**

**Успешное добавление:** при успешном запросе, данные, представленные в виде JSON файла, заносятся в базу данных.

**Ошибка:** 400 Bad Request

**Управление складами**

**Получение списка складов**

**Метод: GET**

**URL:** <http://localhost:62328/api/Warehouse>

**Ответ:** Выводит весь список складов, хранящихся в базе данных.

**Модели данных**

**Response\_ Products:**

public class Response\_Products

{

public Response\_Products(Товары товары)

{

IDТовара = товары.IDТовара;

Название = товары.Название;

Артикул = товары.Артикул;

Штрихкод = товары.Штрихкод;

Категория = товары.Категория;

ЕдиницаИзмерения = товары.ЕдиницаИзмерения;

Цена = товары.Цена;

СерийныйНомер = товары.СерийныйНомер;

МинимальныйЗапас = товары.МинимальныйЗапас;

}

public int IDТовара { get; set; }

public string Название { get; set; }

public string Артикул { get; set; }

public string Штрихкод { get; set; }

public string Категория { get; set; }

public string ЕдиницаИзмерения { get; set; }

public decimal Цена { get; set; }

public string СерийныйНомер { get; set; }

public Nullable<int> МинимальныйЗапас { get; set; }

}

**Response\_Warehouse:**

public class Response\_Warehouse

{

public Response\_Warehouse(Склады склады)

{

IDСклада = склады.IDСклада;

Название = склады.Название;

Адрес = склады.Адрес;

ТипСклада = склады.ТипСклада;

ЗонаХранения = склады.ЗонаХранения;

IDТовара = склады.IDТовара;

Количество = склады.Количество;

}

public int IDСклада { get; set; }

public string Название { get; set; }

public string Адрес { get; set; }

public string ТипСклада { get; set; }

public string ЗонаХранения { get; set; }

public Nullable<int> IDТовара { get; set; }

public Nullable<int> Количество { get; set; }

}

**Response\_Пользователи:**

public class Response\_Пользователи

{

public Response\_Пользователи(Пользователи юзер)

{

IDПользователя = юзер.IDПользователя;

ИмяПользователя = юзер.ИмяПользователя;

Email = юзер.Email;

ХэшПароля = юзер.ХэшПароля;

Роль = юзер.Роль;

Фото = юзер.Фото;

TwoFactorToken = юзер.TwoFactorToken;

}

public int IDПользователя { get; set; }

public string ИмяПользователя { get; set; }

public string Email { get; set; }

public string ХэшПароля { get; set; }

public string Роль { get; set; }

public byte[] Фото { get; set; }

public string TwoFactorToken { get; set; }

}

**LoginRequest:**

public class LoginRequest

{

public string Email { get; set; }

public string Password { get; set; }

}

**Login2FARequest:**

public class Login2FARequest

{

public string Email { get; set; }

public string Token { get; set; }

}

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.

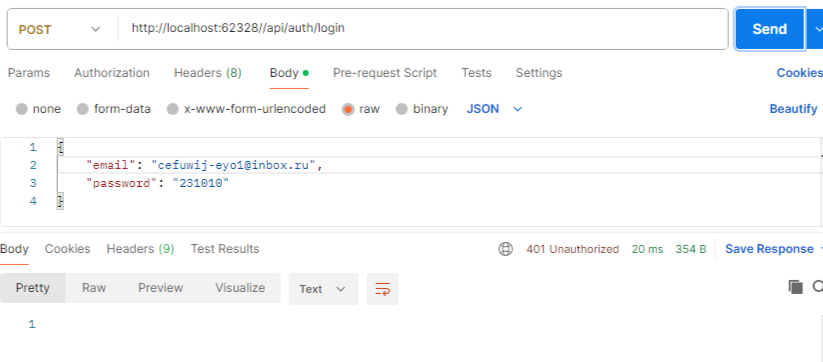
1. **ВЫПОЛНЕНИЕ ТЕСТИРОВАНИЯ ПРОГРАММНОГО МОДУЛЯ**

На этапе тестирования была проведена комплексная проверка программного модуля для обеспечения его корректной работы и соответствия требованиям технического задания (ТЗ).

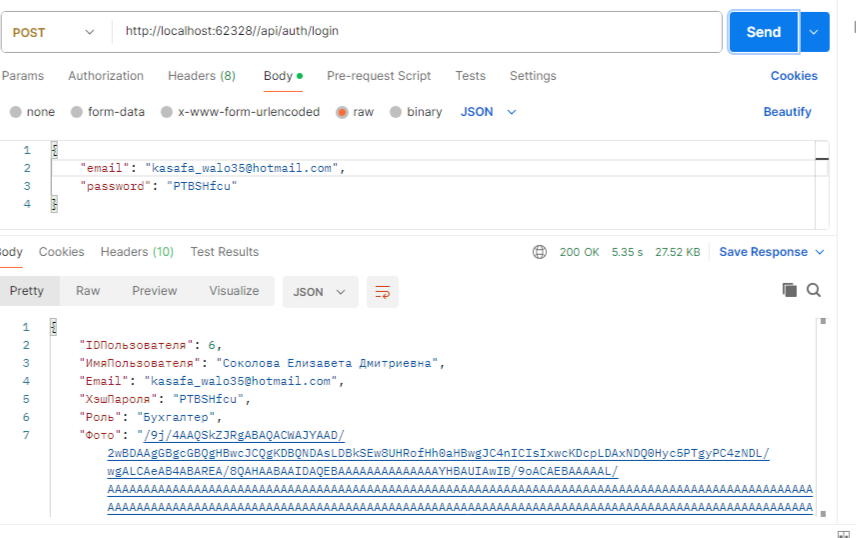
**Нагрузочное тестирование API**

**Postman-запросы:**

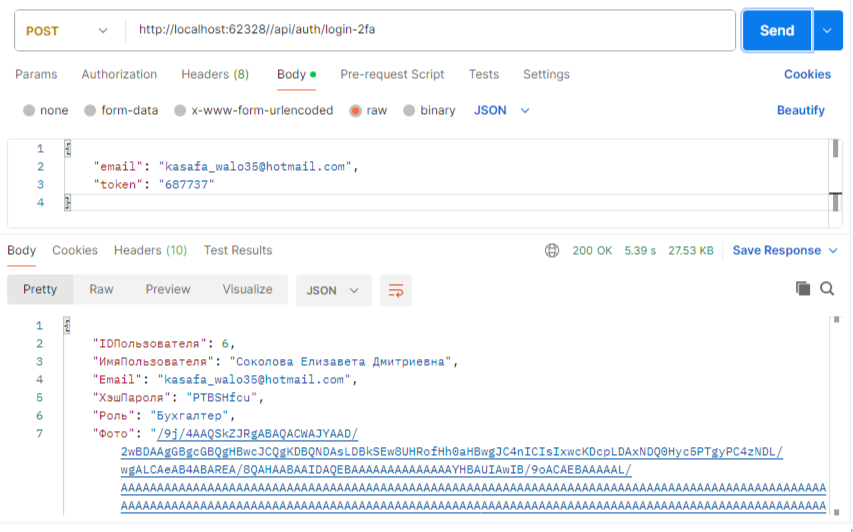
**Авторизация (неправильный пароль) (Рисунок 6.1):**



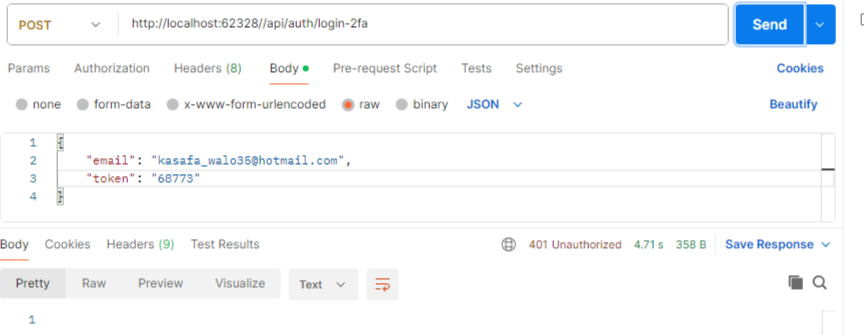
**Авторизация (успешная) (Рисунок 6.2):**

****

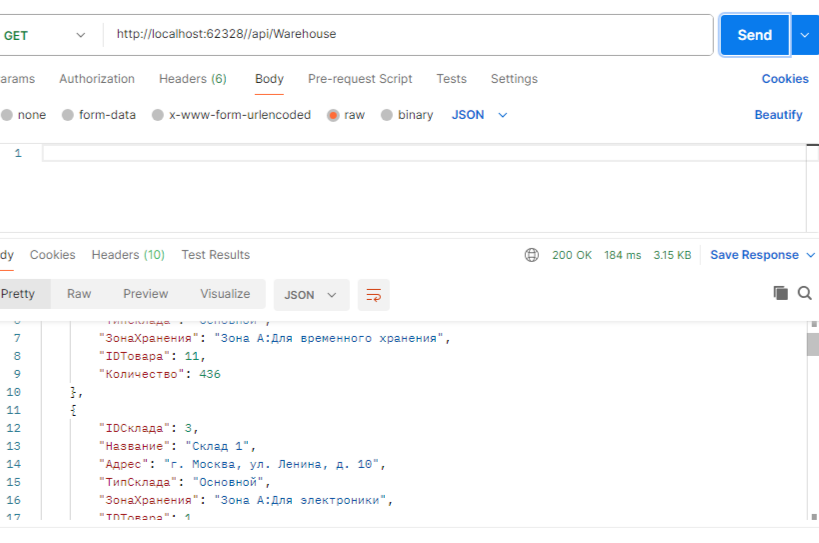
**Авторизация через токен (Рисунок 6.3):**

****

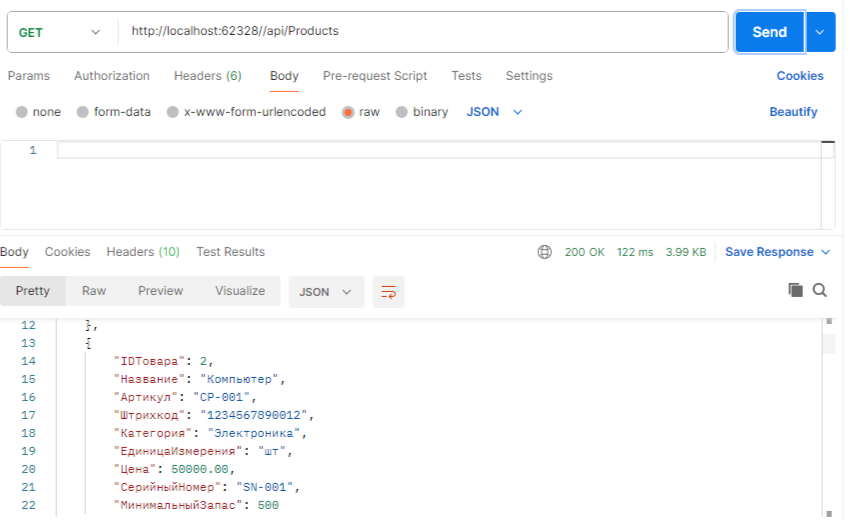
**Авторизация через неправильный токен (Рисунок 6.4):**

****

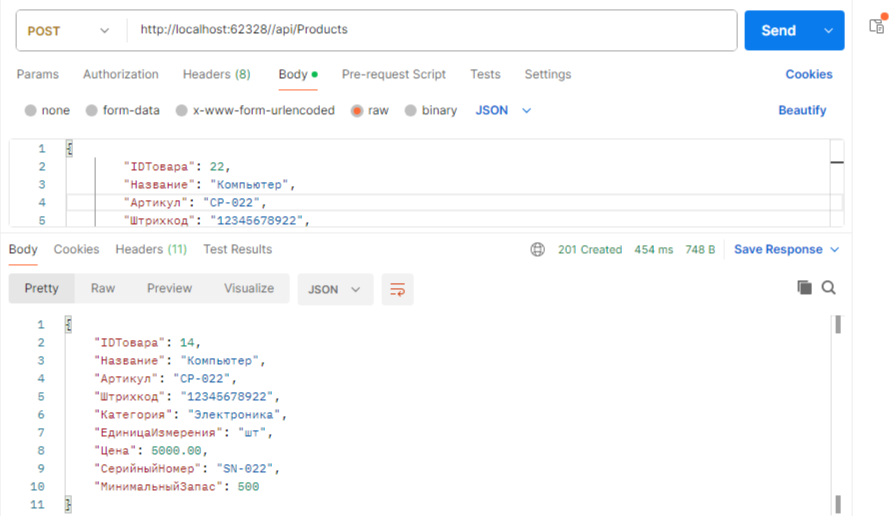
**Получить список складов (Рисунок 6.5):**

****

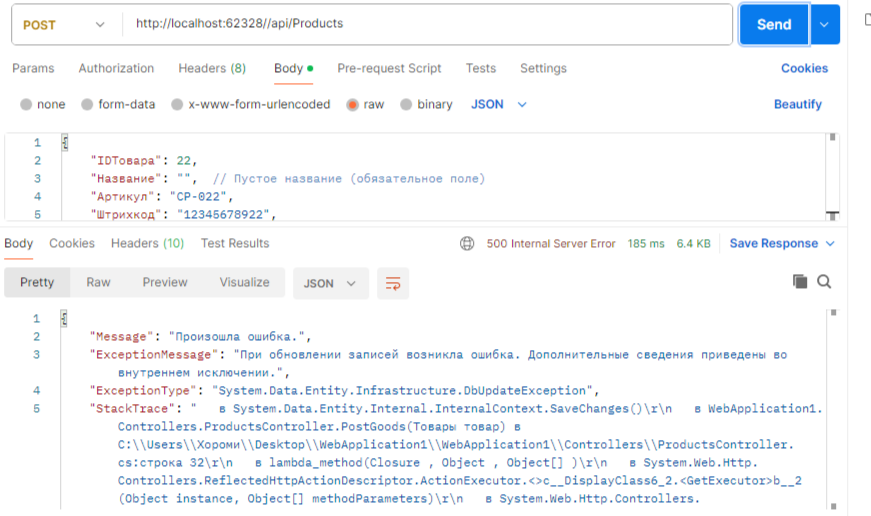
**Получить список товаров (Рисунок 6.6):**

****

**Добавить товар (с правильными данными) (Рисунок 6.7):**



**Добавить товар (с неправильными данными) (Рисунок 6.8):**

****

**Интеграционное тестирование библиотеки**

**Код программы для тестирования:**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using WarehouseLibrary;

namespace ConsoleApp1

{

internal class Program

{

static void Main(string[] args)

{

// Создаем экземпляр WarehouseManager

var warehouseManager = new WarehouseManager();

// Тест 1: Подсчет общего количества товара на всех складах

int productID = 1; // Пример ID товара

int totalQuantity = warehouseManager.CalculateTotalQuantityForProduct(productID);

Console.WriteLine($"Общее количество товара с ID {productID} на всех складах: {totalQuantity}");

// Тест 2: Подсчет количества товара на конкретном складе

int warehouseID = 1; // Пример ID склада

int quantityInWarehouse = warehouseManager.CalculateQuantityForProductInWarehouse(productID, warehouseID);

Console.WriteLine($"Количество товара с ID {productID} на складе {warehouseID}: {quantityInWarehouse}");

// Тест 3: Подсчет суммы стоимости товаров на всех складах

decimal totalValue = warehouseManager.CalculateTotalValueForProduct(productID);

Console.WriteLine($"Общая стоимость товара с ID {productID} на всех складах: {totalValue}");

// Тест 4: Подсчет суммы стоимости товаров на конкретном складе

decimal valueInWarehouse = warehouseManager.CalculateValueForProductInWarehouse(productID, warehouseID);

Console.WriteLine($"Стоимость товара с ID {productID} на складе {warehouseID}: {valueInWarehouse}");

// Тест 5: Подсчет количества товаров по категориям на всех складах

Dictionary<string, int> quantityByCategory = warehouseManager.CalculateQuantityByCategory();

Console.WriteLine("Количество товаров по категориям на всех складах:");

foreach (var category in quantityByCategory)

{

Console.WriteLine($"{category.Key}: {category.Value}");

}

// Тест 6: Подсчет количества товаров по категориям на конкретном складе

Dictionary<string, int> quantityByCategoryInWarehouse = warehouseManager.CalculateQuantityByCategoryInWarehouse(warehouseID);

Console.WriteLine($"Количество товаров по категориям на складе {warehouseID}:");

foreach (var category in quantityByCategoryInWarehouse)

{

Console.WriteLine($"{category.Key}: {category.Value}");

}

// Отрицательные тесты

// Тест 7: Подсчет общего количества для несуществующего товара

int nonExistentProductID = 999;

int totalQuantityNonExistent = warehouseManager.CalculateTotalQuantityForProduct(nonExistentProductID);

Console.WriteLine($"Общее количество несуществующего товара с ID {nonExistentProductID}: {totalQuantityNonExistent}");

// Тест 8: Подсчет количества для несуществующего товара на складе

int quantityNonExistentInWarehouse = warehouseManager.CalculateQuantityForProductInWarehouse(nonExistentProductID, warehouseID);

Console.WriteLine($"Количество несуществующего товара с ID {nonExistentProductID} на складе {warehouseID}: {quantityNonExistentInWarehouse}");

// Тест 9: Подсчет суммы стоимости для несуществующего товара

decimal totalValueNonExistent = warehouseManager.CalculateTotalValueForProduct(nonExistentProductID);

Console.WriteLine($"Общая стоимость несуществующего товара с ID {nonExistentProductID}: {totalValueNonExistent}");

// Тест 10: Подсчет суммы стоимости для несуществующего товара на складе

decimal valueNonExistentInWarehouse = warehouseManager.CalculateValueForProductInWarehouse(nonExistentProductID, warehouseID);

Console.WriteLine($"Стоимость несуществующего товара с ID {nonExistentProductID} на складе {warehouseID}: {valueNonExistentInWarehouse}");

}

}

}

**Результат тестирования (Рисунок 6.9):**



1. **ОФОРМЛЕНИЕ РЕЗУЛЬТАТОВ ТЕСТИРОВАНИЯ**

Результаты тестирования были систематизированы и оформлены в виде отчетов, которые включают:

**Описание тестовых сценариев**

В данном разделе представлен подробный перечень всех проверенных функций и сценариев использования программного модуля. Например, были проверены сценарии авторизации, работы с товарами, формирования отчетов и другие ключевые функции, описанные в техническом задании.

**Отчеты были оформлены в виде онлайн документации:**

Все результаты тестирования, включая отчеты, были загружены в репозиторий на GitHub. Это обеспечивает удобное хранение, версионность и доступ к документам для всех участников проекта. Каждый отчет был размещен в папке репозитория, что позволяет легко находить нужные материалы и отслеживать изменения.

Таким образом, результаты тестирования были тщательно проанализированы и оформлены в удобном для использования виде, что позволяет эффективно работать с ними на всех этапах жизненного цикла проекта.

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:**

**Двухфакторная аутентификация**

* Реализована система входа с использованием двухфакторной аутентификации для повышения безопасности.
* Пользователь вводит логин и пароль, после чего вводит код для двухфакторной аутентификации.

**Меню приложения**

Приложение включает удобное меню с несколькими фрагментами:

* Список складов и их местоположение — отображение всех складов с возможностью просмотра их местоположения на карте.
* 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.

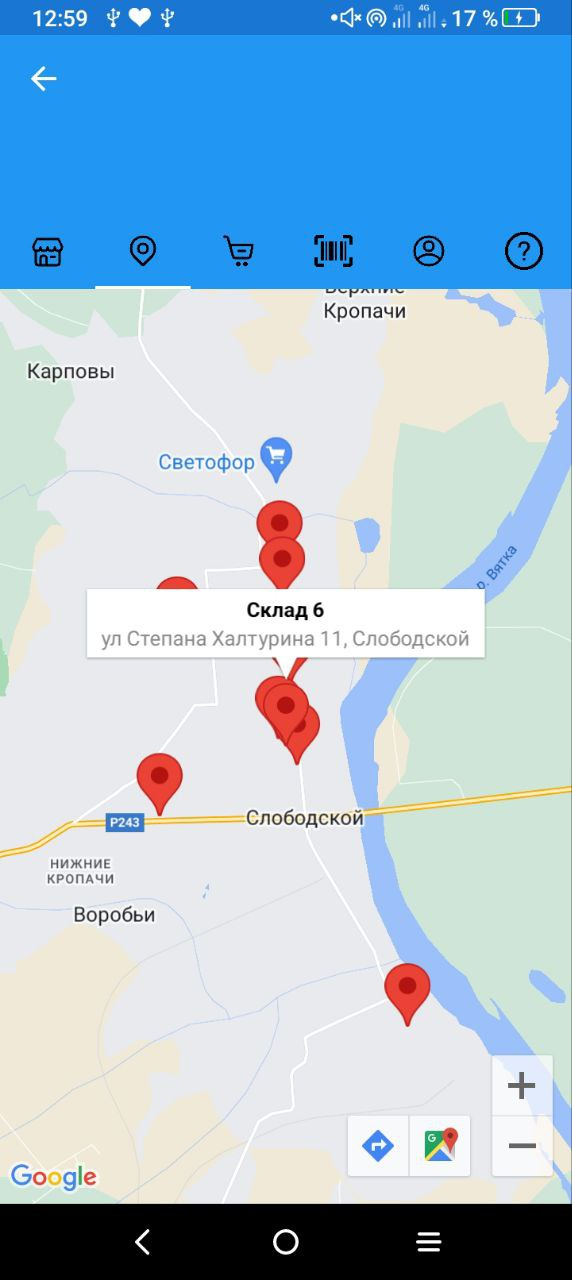


Рисунок 8.1 – Скриншоты интерфейса

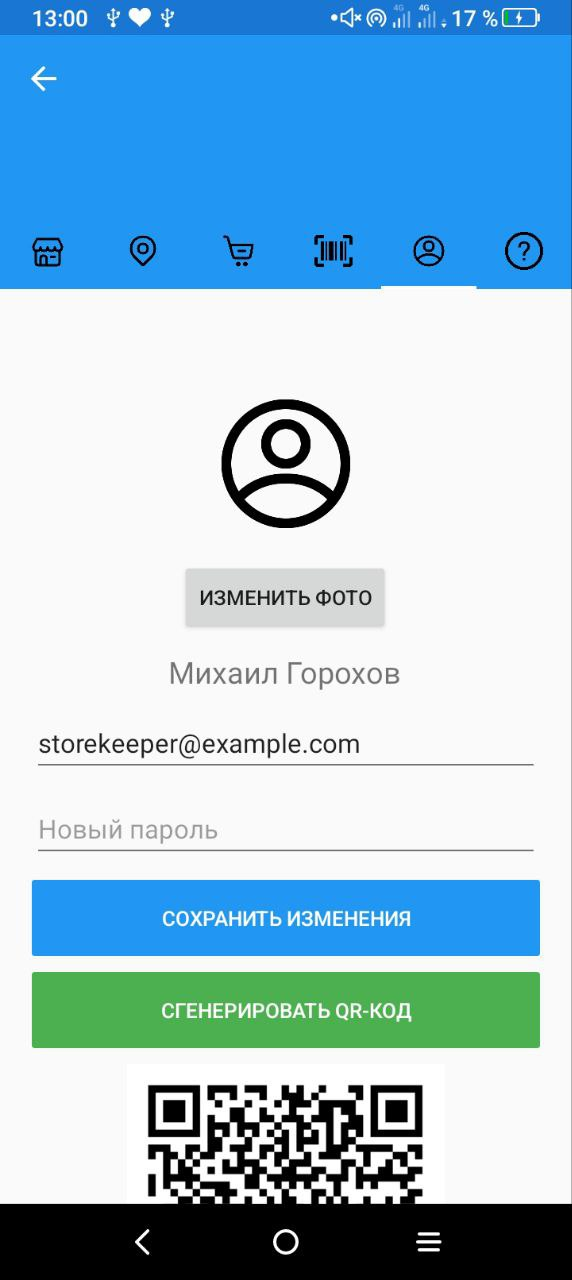


Рисунок 8.2 – Скриншоты интерфейса

* 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**

На данном этапе было проведено тестирование мобильного приложения для проверки его работоспособности и корректности работы на реальных устройствах и эмуляторах. Основной целью было убедиться, что приложение функционирует стабильно, интерфейс удобен для пользователя, а все функции работают в соответствии с техническим заданием.

**Тестирование интерфейса мобильного приложения**

|  |  |
| --- | --- |
| TestCase # | 1 |
| Приоритет теста | **Высокий** |
| Название тестирования/Имя | **Успешная двухфакторная аутентификация** |
| Резюме испытания | **Проверка успешной аутентификации пользователя с использованием двухфакторной аутентификации** |
| Шаги тестирования | * **Открыть приложение** * **Ввести логин и пароль** * **Ввести роль в системе** * **Нажать "Войти"** |
| Данные тестирования | **Логин:** [**storekeeper@example.com**](mailto:storekeeper@example.com)  **Пароль: store1234** |
| Ожидаемый результат | **Пользователь успешно авторизован, открывается главное меню приложения** |
| Фактический результат |  |
| Предпосылки | **Пользователь зарегистрирован в системе** |
| Постусловия | **Пользователь находится в главном меню** |
| Статус  (Pass/Fail) | **Pass** |
| Комментарии | **Тест пройден** |

|  |  |
| --- | --- |
| TestCase # | 2 |
| Приоритет теста | **Высокий** |
| Название тестирования/Имя | **Просмотр списка складов** |
| Резюме испытания | **Проверка отображения списка складов** |
| Шаги тестирования | * **Авторизоваться в приложении.** * **Перейти в раздел "Склады".** * **Проверить, правильно ли отображается информация** |
| Данные тестирования | **Список складов в базе данных** |
| Ожидаемый результат | **Отображается список складов с их названиями и адресами** |
| Фактический результат |  |
| Предпосылки | **Пользователь авторизован в системе** |
| Постусловия | **Пользователь видит список складов** |
| Статус  (Pass/Fail) | **Pass** |
| Комментарии | **Тест пройден** |

|  |  |
| --- | --- |
| TestCase # | 3 |
| Приоритет теста | **Высокий** |
| Название тестирования/Имя | **Просмотр списка товаров на складе** |
| Резюме испытания | **Проверка отображения списка товаров на выбранном складе** |
| Шаги тестирования | * **Авторизоваться в приложении.** * **Перейти в раздел "Товары на складах".** * **Проверить, правильно ли отображается информация** |
| Данные тестирования | **Список товаров на выбранном складе** |
| Ожидаемый результат | **Отображается список товаров с их названиями, количеством и единицами измерения** |
| Фактический результат |  |
| Предпосылки | **Пользователь авторизован в системе** |
| Постусловия | **Пользователь видит список товаров на складе** |
| Статус  (Pass/Fail) | **Pass** |
| Комментарии | **Тест пройден** |

|  |  |
| --- | --- |
| TestCase # | 4 |
| Приоритет теста | **Высокий** |
| Название тестирования/Имя | **Сканирование штрих кода** |
| Резюме испытания | **Проверка успешного сканирования штрих кода и отображения информации о товаре** |
| Шаги тестирования | * **Авторизоваться в приложении.** * **Перейти в раздел "Сканирование".** * **Сканировать штрих код товара.** |
| Данные тестирования | **Действительный штрих код товара** |
| Ожидаемый результат | **Отображается информация о товаре** |
| Фактический результат | **Отображается информация о товаре** |
| Предпосылки | **Пользователь авторизован в системе** |
| Постусловия | **Пользователь видит информацию о товаре** |
| Статус  (Pass/Fail) | **Fail** |
| Комментарии | **Тест не пройден** |

|  |  |
| --- | --- |
| TestCase # | 5 |
| Приоритет теста | **Высокий** |
| Название тестирования/Имя | **Ввод штрих кода вручную** |
| Резюме испытания | **Проверка штрих кода при вводе вручную и отображения информации о товаре** |
| Шаги тестирования | * **Авторизоваться в приложении.** * **Перейти в раздел "Сканирование".** * **Нажать «Ввести штрих-код вручную»** * **Ввести штрих код вручную и проверить, отображается ли информация о товаре** |
| Данные тестирования | **Действительный штрих код товара** |
| Ожидаемый результат | **Отображается информация о товаре** |
| Фактический результат |  |
| Предпосылки | **Пользователь авторизован в системе** |
| Постусловия | **Пользователь видит информацию о товаре** |
| Статус  (Pass/Fail) | **Pass** |
| Комментарии | **Тест пройден** |

|  |  |
| --- | --- |
| TestCase # | 6 |
| Приоритет теста | **Высокий** |
| Название тестирования/Имя | **Просмотр профиля** |
| Резюме испытания | **Проверка отображения профиля пользователя с пропуском** |
| Шаги тестирования | * **Авторизоваться в приложении.** * **Перейти в раздел "Профиль".** * **Проверить, все ли отображается** |
| Данные тестирования | **Данные профиля пользователя** |
| Ожидаемый результат | **Отображается информация о пользователе и его пропуск** |
| Фактический результат |  |
| Предпосылки | **Пользователь авторизован в системе** |
| Постусловия | **Пользователь видит свой профиль** |
| Статус  (Pass/Fail) | **Pass** |
| Комментарии | **Тест пройден** |

|  |  |
| --- | --- |
| TestCase # | 7 |
| Приоритет теста | **Высокий** |
| Название тестирования/Имя | **Проверка генерации пропуска (QR-кода)** |
| Резюме испытания | **Проверка генерации пропуска (QR-кода)** |
| Шаги тестирования | * **Авторизоваться в приложении.** * **Перейти в раздел "Профиль".** * **Нажать «Сгенерировать QR-код»** |
| Данные тестирования | **Данные профиля пользователя** |
| Ожидаемый результат | **QR-код генерируется правильно** |
| Фактический результат |  |
| Предпосылки | **Пользователь авторизован в системе** |
| Постусловия | **Пользователь видит свой профиль и пропуск** |
| Статус  (Pass/Fail) | **Pass** |
| Комментарии | **Тест пройден** |

|  |  |
| --- | --- |
| TestCase # | 8 |
| Приоритет теста | **Высокий** |
| Название тестирования/Имя | **Проверка изменения фотографии пользователя** |
| Резюме испытания | **Проверка изменения фотографии в профиле пользователя** |
| Шаги тестирования | * **Авторизоваться в приложении.** * **Перейти в раздел "Профиль".** * **Нажать «Изменить фото»** * **Выбрать новую фотографию** * **Нажать «Сохранить изменения»** |
| Данные тестирования | **Данные профиля пользователя** |
| Ожидаемый результат | **Фотография изменяется правильно** |
| Фактический результат |  |
| Предпосылки | **Пользователь авторизован в системе** |
| Постусловия | **Пользователь видит свой профиль и пропуск** |
| Статус  (Pass/Fail) | **Pass** |
| Комментарии | **Тест пройден** |

|  |  |
| --- | --- |
| TestCase # | 9 |
| Приоритет теста | **Высокий** |
| Название тестирования/Имя | **Проверка отображения местоположения складов на карте** |
| Резюме испытания | **Проверка отображения местоположения складов на карте** |
| Шаги тестирования | * **Авторизоваться в приложении.** * **Перейти в раздел "Местоположение складов".** * **Посмотреть, отображаются ли метки на карте** |
| Данные тестирования | **Список складов с координатами для отображения в базе данных** |
| Ожидаемый результат | **Все метки отображаются корректно** |
| Фактический результат |  |
| Предпосылки | **Пользователь авторизован в системе** |
| Постусловия | **Пользователь видит метки на карте** |
| Статус  (Pass/Fail) | **Pass** |
| Комментарии | **Тест пройден** |

|  |  |
| --- | --- |
| TestCase # | 10 |
| Приоритет теста | **Высокий** |
| Название тестирования/Имя | **Проверка отображения справочного раздела приложения** |
| Резюме испытания | **Проверка отображения справочного раздела приложения** |
| Шаги тестирования | * **Авторизоваться в приложении.** * **Перейти в раздел "Справка".** * **Посмотреть, отображается ли справочная информация** |
| Данные тестирования | **Раздел "Справка" содержит информацию о работе приложения** |
| Ожидаемый результат | **Раздел "Справка" отображается корректно, информация доступна для чтения** |
| Фактический результат |  |
| Предпосылки | **Пользователь авторизован в системе** |
| Постусловия | **Пользователь видит раздел "Справка" с доступной информацией** |
| Статус  (Pass/Fail) | **Pass** |
| Комментарии | **Тест пройден** |

**Руководство оператора для мобильного приложения**

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 9.1).



Рисунок 9.1 – Окно авторизации

Для того что бы начать использовать приложение вам необходимо пройти авторизацию. Для этого необходимо ввести свои учетные данные и нажать на кнопку «Войти». После нажатия кнопки входа появится всплывающее окно со следующим этапом – двухфакторная аутентификация, где вам необходимо ввести ваш код 2FA. (Рисунок 9.2)

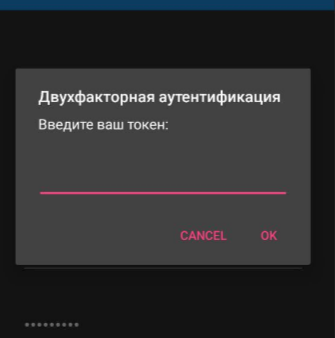


Рисунок 9.2 – Двухфакторная аутентификация

После ввода правильных данных, появится уведомление об успешной авторизации. (Рисунок 9.3).

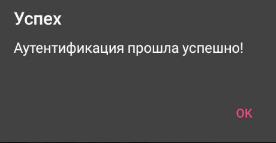


Рисунок 9.3 – Уведомление об успешной авторизации

Далее после успешной авторизации открывается меню навигации по разделам приложения. (Рисунок 9.4).



Рисунок 9.4 – Навигация

Первая страница в разделе навигации представляет список складов с подробной информацией (Рисунок 9.5).

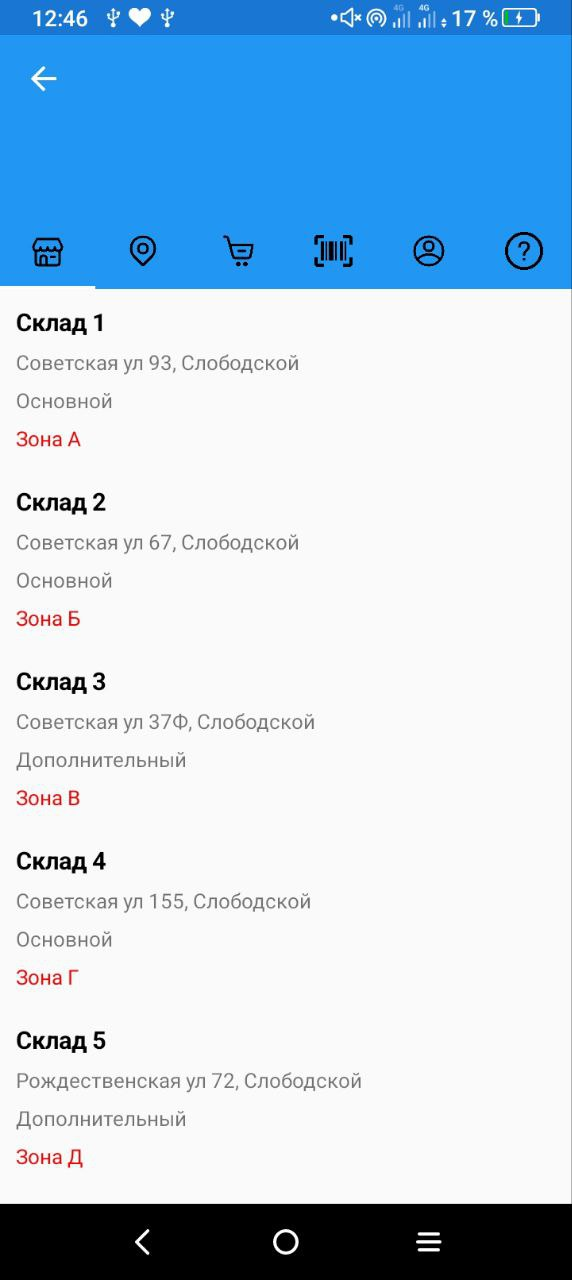


Рисунок 9.5 – Список складов

Вторая страница представляет собой карту с местоположением складов (Рисунок 9.6).

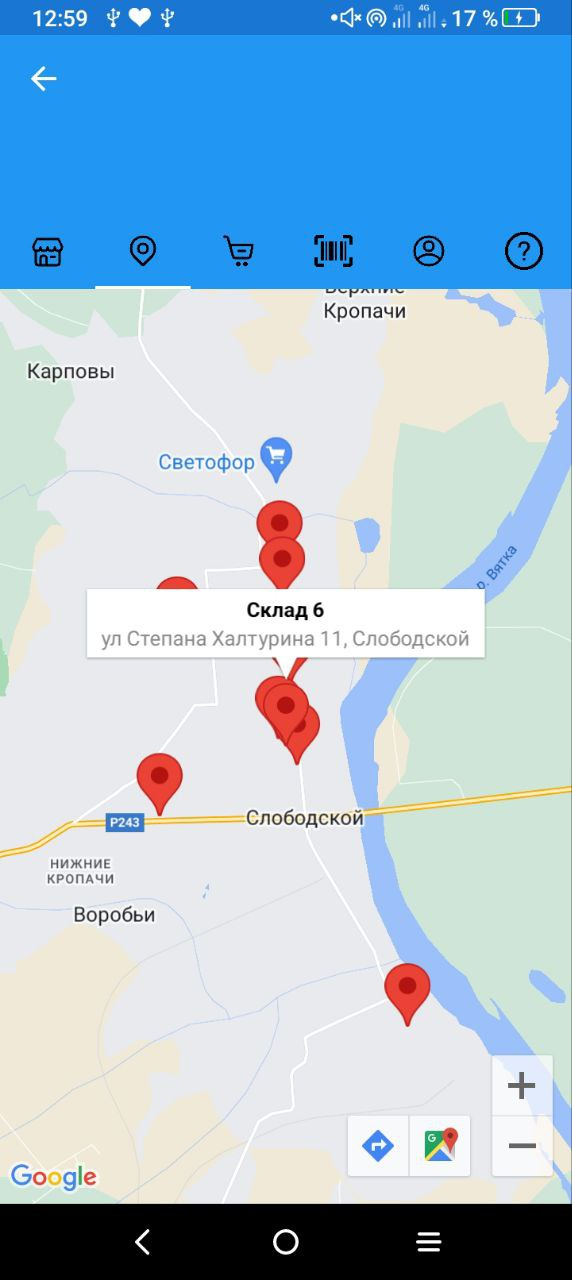


Рисунок 9.6 – Местоположение складов

Третья страница представляет возможность просмотра списка товаров по складам, сам список появляется в виде сплывающего окна. (Рисунок 9.7, Рисунок 9.8)

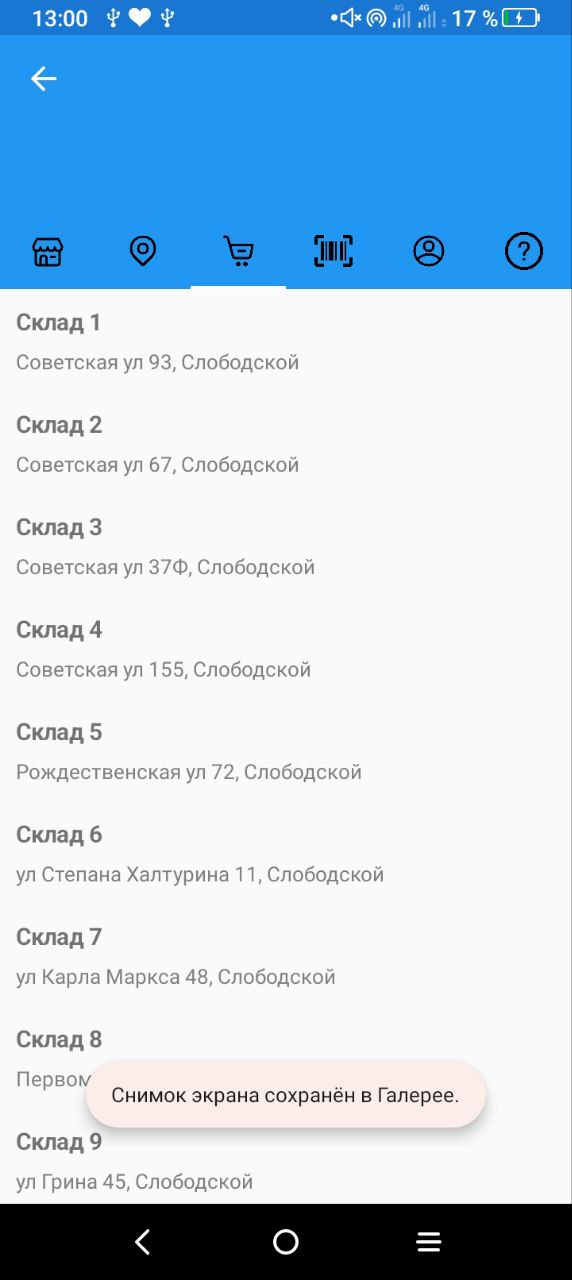


Рисунок 9.7 – Просмотр товаров на складах

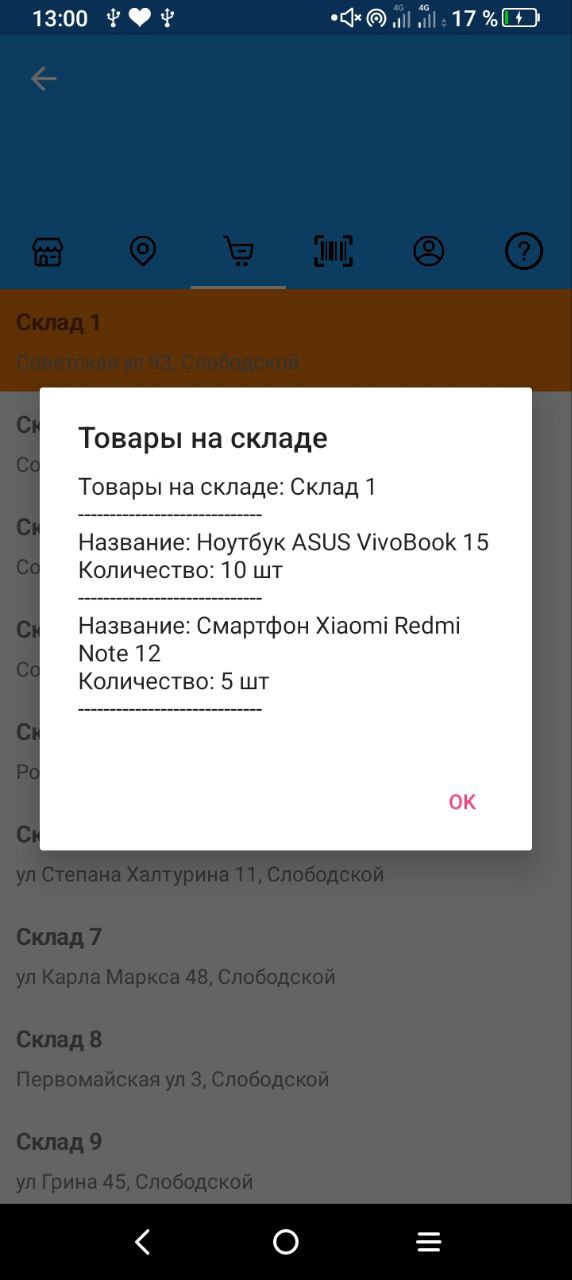


Рисунок 9.8 – Всплывающее окно с информацией

Четвертая страница представляет возможность сканирования штрих-кодов товаров: как через камеру, так и вручную. Информация о товаре по отсканированному штрих-коду появляется в виде сплывающего окна. (Рисунок 9.9)

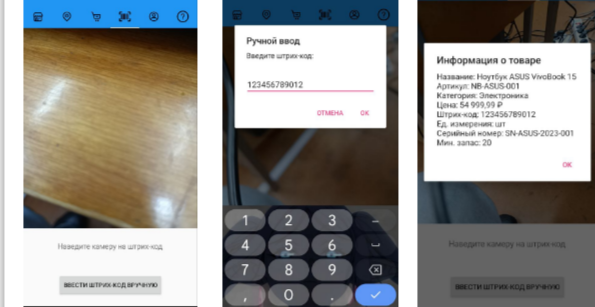


Рисунок 9.9 – Сканирование штрих-кодов

Пятая страница представляет собой профиль пользователя с возможностью редактирования данных и генерацией пропуска (QR-кода) (Рисунок 9.10)

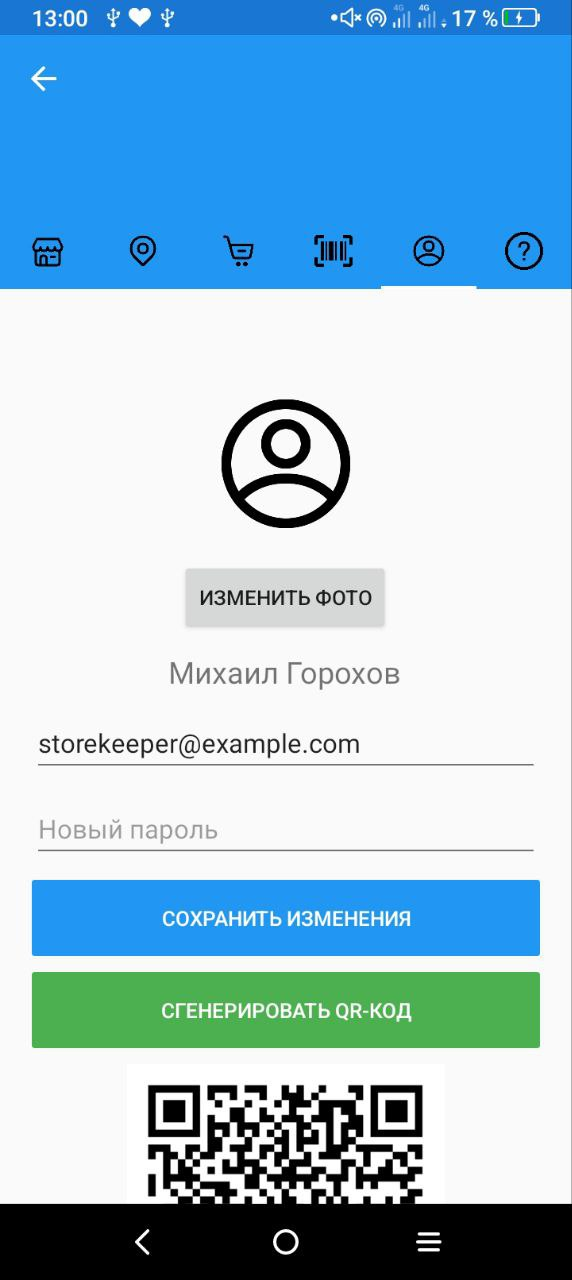


Рисунок 9.10 – Профиль пользователя

Последняя страница представляет из себя справку о программе (Рисунок 9.11)

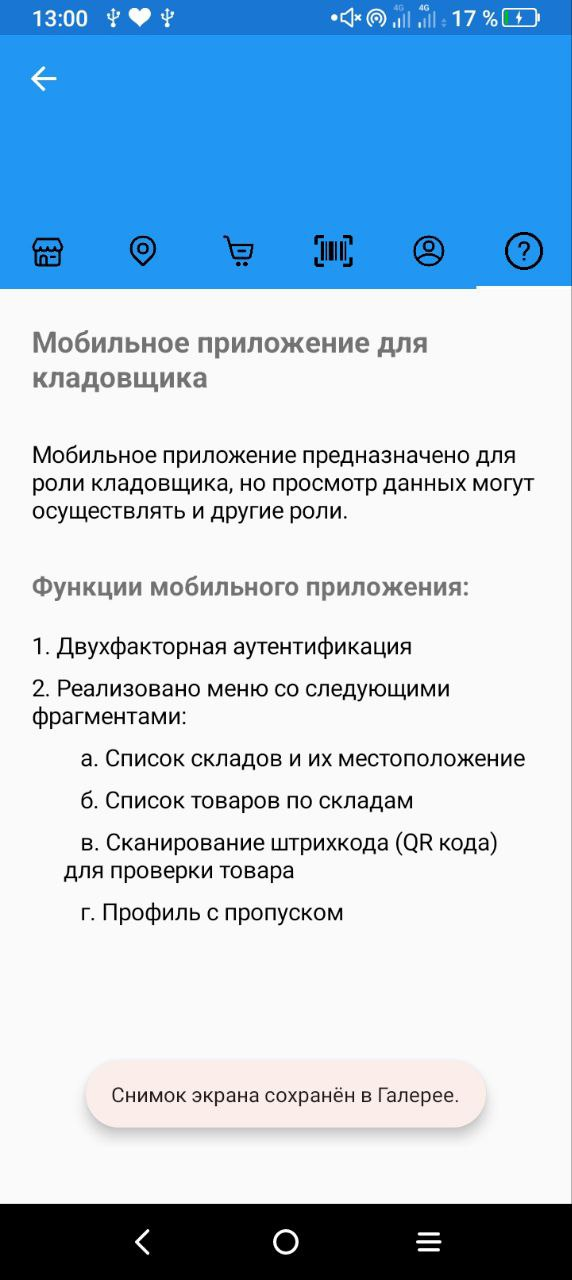


Рисунок 9.11 – Справка о программе

1. **ЗАКЛЮЧЕНИЕ.**

Учебная практика дала мне ценный опыт, который помог расширить мои профессиональные знания и навыки. В процессе работы над проектом я углубила свои знания в области разработки программного обеспечения, включая проектирование архитектуры системы, работу с базами данных, создание API и мобильных приложений.

Кроме того, я освоила современные инструменты и технологии, такие как C#, WPF, Xamarin, SQLite и ADO.NET, что значительно повысило мою техническую компетентность.

Кроме того, учебная практика позволила мне применить теоретические знания на практике, что помогло мне лучше понять их применение в реальных рабочих ситуациях. Благодаря работе над практическими проектами я приобрела ценный опыт в решении реальных проблем и разработке инновационных решений.

Например, я научилась анализировать требования технического задания, проектировать диаграммы (ER-диаграммы, диаграммы последовательности и вариантов использования), а также проводить тестирование и отладку программного обеспечения для обеспечения его стабильной работы.

Этот практический опыт дал мне преимущество перед другими кандидатами на рынке труда, поскольку я уже обладаю практическими навыками и опытом, необходимыми для выполнения рабочих задач. Я научилась работать в команде, эффективно распределять время и ресурсы, а также находить оптимальные решения для сложных задач.

Эти навыки, безусловно, будут полезны в моей будущей профессиональной деятельности.

Я уверена, 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 готова 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/HoromiUzumaki/Praktika_PM01>