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

Труфакин Сергей Васильевич

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.

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).

Изображение выглядит как диаграмма, снимок экрана, круг, текст

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 1 - Диаграмма вариантов использования

**Диаграмма последовательности** — для демонстрации последовательности действий и взаимодействия между компонентами системы (рисунок 2.).

Изображение выглядит как текст, снимок экрана, Шрифт, число

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 2. - Диаграмма последовательности (приемка товаров)

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

Изображение выглядит как текст, диаграмма, План, Параллельный

Контент, сгенерированный ИИ, может содержать ошибки.

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.

**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.

Изображение выглядит как текст, снимок экрана, Шрифт, число

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 3.5 – Окно входа

Изображение выглядит как текст, снимок экрана, число, Шрифт

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 3.6 – Настольное приложение (окно Администратора)

Изображение выглядит как текст, снимок экрана, Шрифт

Контент, сгенерированный ИИ, может содержать ошибки.

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).

Изображение выглядит как текст, снимок экрана, Шрифт, число

Контент, сгенерированный ИИ, может содержать ошибки.

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).

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.

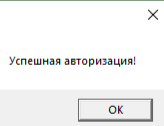


Рисунок 4.3 – Уведомление об успешной авторизации

**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).

Изображение выглядит как текст, снимок экрана, число, Шрифт

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 4.4 – Раздел администрирования складами

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

Изображение выглядит как текст, снимок экрана, Шрифт

Контент, сгенерированный ИИ, может содержать ошибки.

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).

Изображение выглядит как текст, снимок экрана, программное обеспечение, число

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 4.10 – Оформление приходной накладной

Изображение выглядит как текст, электроника, снимок экрана, дисплей

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 4.11 – Оформление расходной накладной

Изображение выглядит как текст, снимок экрана, дисплей, программное обеспечение

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 4.12 – Создание штрих-кода

Изображение выглядит как текст, электроника, снимок экрана, программное обеспечение

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 4.14 – 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. **ЗАКЛЮЧЕНИЕ.**

Учебная практика дала 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, создание API и мобильных приложений.

Кроме того, я освоил 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 приобрел ценный опыт в решении реальных проблем и разработке инновационных решений.

Например, я научился 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.

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/HoromiUzumaki/Praktika_PM01>