СОДЕРЖАНИЕ

ВВЕДЕНИЕ 3

АНАЛИТИЧЕСКАЯ ЧАСТЬ 5

Анализ предметной области 5

Постановка задачи 6

ПРОЕКТНАЯ ЧАСТЬ 8

Проектирование и реализация базы данных системы 8

Проектирование и реализация пользовательского интерфейса системы 9

Структура приложения 10

Реализация пользовательских форм 11

Разработка запросов 15

Разработка отчетов 18

Меню и справка 20

ЗАКЛЮЧЕНИЕ 21

Список литературы 22

Приложение А. Техническое задание на курсовой проект 23

# ВВЕДЕНИЕ

Информационные системы бухгалтерского и складского учёта брали начало ещё при зарождении самих информационных технологий. Из-за скорости и удобства передачи информации, формализации данных и организации результатов при помощи ЭВМ ещё с 80-х годов происходило повышение эффективности деятельности.

Информационная система - система, предназначенная для хранения, поиска и обработки информации, и соответствующие организационные ресурсы (человеческие, технические, финансовые и т. д.), которые обеспечивают и распространяют информацию (ISO/IEC 2382:2015).

Информационная система позволяет в кратчайшие сроки составить отчёт или документацию по отдельному товару, набору деталей, ассортименту магазина и даже всему складскому наполнению.

Информационные системы помимо снижения трудозатрат на составление документа, позволяют сократить сам документооборот вплоть до отмены от него. Некоторые документы могут быть заменены на вывод отчёта в самой программе или сам документ может быть более исчерпывающий.

Из вышеописанного можно сделать вывод: эффективность работы бухгалтерии или складских учётов напрямую зависит от степени развития информационных систем и их применения в деятельности отдельных организаций.

При учёте поставок крайне полезной была бы система автоматизированного учёта, способная сохранять и передавать в необходимые хранилища отчеты о поставке, количествах продукции и прочих деталях.

Целью курсового проекта является разработка системы учёта продукции на складе и нахождение оптимальных решений возможных проблем. Данная система будет способна создавать отчёт о текущем наполнении склада, дефицитной продукции, поставках как прошедших, так и готовящихся, а также учёте прибывшей и убывшей продукции.

Так как склад не занимается продажами, то в системе не будет учёта продаж, цены или прибылей, так как склад занимается только хранением и учётом продукции, которая была поставлена на хранение заинтересованными лицами.

В данном курсовом проекте будет рассмотрена автоматизация работы с учётом продукции на складе. Результатом данной работы будет информационная система учёта продукции в наличии на складе абстрактной компании, а также будут продемонстрированы преимущества использования информационных систем в данной сфере.

Так как каждая организация, занимающаяся розничной или оптовой продажей нецифровых товаров, имеет склад, нужно учесть, что возможно множество нюансов между организациями и самой продукцией.

Предметом исследования является процесс учета продукции в наличии и процесс приёма новых поставок, а также перемещение продукции со склада.

Для достижения поставленной цели необходимо выполнить следующие задачи:

* + 1. Провести анализ предметной области.
    2. Разработать техническое задание на информационную систему.
    3. Спроектировать БД системы.
    4. Разработать запросы в БД.
    5. Реализовать БД.
    6. Реализовать пользовательский интерфейс.
    7. Разработать пособие и инструкцию по работе с данной информационной системой.
    8. Провести тестирование разработанной информационной системы.

# АНАЛИТИЧЕСКАЯ ЧАСТЬ

## Анализ предметной области

Склад – это некоторая площадь, чаще всего в помещении, предназначенная для хранения любого рода материальных ценностей компаний или любого другого заинтересованного лица. Сам по себе склад используется исключительно для аккумулирования, приёма и передачи ресурсов. Иные операции производятся уже иными сущностями, магазинами для продажи, компаниями транспортировки для перевозки и т.д.

Материальные ценности – это товары, имущество или любые другие материальные предметы, являющиеся собственностью заинтересованного лица.

Функции склада:

* Снабжение – это основная функция склада. Данная функция состоит в том, что склад имеет главной целью обеспечение и снабжение какого-либо производства
* Связующая – данная функция заключается в том, что склад, помимо хранения материалов для изготовления, может хранить и готовую продукцию, таким образом склад будет связующим звеном между рынком сбыта и производством.
* Переработка – так как некоторая продукция в своём производственном цикле требует некоторого времени хранения для подготовки к последующему изменению или продаже.
* Перетарка – зачастую именно на складе происходит изменение упаковки продукции, если это необходимо владельцам продукции.
* Спекулятивная – данная функция может быть названа защищающей, сохранение товара при падении ценности товара или требование роста поставок при их росте.

Система, разработанная в данной курсовой работе, будет предназначена для отслеживания, текущего количество отдельной продукции или продукции отдельных клиентов склада: добавление новых товаров и исключение товаров убывших. Данная информационная система не будет отслеживать затраты поставку товара или учитывать прибыли от его продажи.

Поставка – совокупность продукции или любых других материальных ценностей, прибывших на хранение от одного заказчика в определённый момент.

Хоть отслеживанием поставок или продажей склад не занимается, так как это не является частью его деятельности, данная система должна отслеживать поступление и отчисление продукции со склада.

Существуют следующие способы поступления-отчисления продукции со склада:

* FIFO (first-in-first-out отгружается продукция, которая прибыла первой)
* LIFO (last-in-first-out отгружается продукция, которая прибыла последней)
* FEFO (first-expire-first-out отгружается продукция с ближайшим к истечению сроком годности)
* HIFO (highest-in-first-out отгружается продукция себестоимость которой, была завышена)
* LOFO (lowest-in-first-out отгружается продукция себестоимость которой, была занижена)

Так как в системе не будет предусмотрен учёт продаж и цены отдельной продукции, не имеет смыла реализация последних двух методов зависящий от цены товаров.

В системе будет возможность выбор расположения товара на складе с учётом его особенностей:

* Хрупкий – данная продукция должна храниться в помещениях с ручной погрузкой
* Хранение в темноте – данная продукция (чаще всего это бытовая химия) негативно реагирует на долгое воздействие света, в следствии чего приходит в негодность раньше
* Влагобоязнь – данная продукция должна храниться в помещениях с определённым уровнем влажности
* Боязнь излучение – любое излучение портит продукцию (к примеру фотоплёнка)
* Запрет хранения штабелями – данное свойство указывает, что продукцию нельзя хранить штабелями

Также будет предусмотрена возможность добавления новых критериев при необходимости.



Рисунок 1 – контекстная функциональная диаграмма по учету продукции на складе

IDEF0 – функциональная модель, отображающая основной или набор второстепенных бизнес-процессов, а также входные данные, которые должны быть предоставлены для выполнения определённого бизнес-процесса, механизмы, при помощи которых он выполняется (снизу) и контролирующие выполнение (сверху), а также результаты данного процесса.



Рисунок 2 – диаграмма декомпозиции учёта продукции

Диаграмма декомпозиции – это функциональная модель, описывающая подпроцессы основного бизнес-процесса системы.

## Постановка задачи

Автоматизация данных процессов позволит сохранить информацию в базе, в которую вводится данная информация с помощью удобного интерфейса программы.

В данной информационной системе необходимо разработать:

* Базу данных в MsSQL.
* Приложение в среде программирования.
* Разработать интуитивно понятный и удобный интерфейс.

Так же нужно разработать функциональные возможности системы:

* Редактирование и просмотр данных о продукции.
* Поиск и фильтр данных о продукции.
* Редактирование и просмотр данных клиентов.
* Редактирование и просмотр данных поставок на склад и отправлений со склада.
* Выбор места для товаров на складе

Так же нужно разработать отчеты в системе:

* отчет о количестве продукции на складе по определённым критериям.

Более подробный функционал системы смотри в приложение А (см. Техническое задание на ИС).

# ПРОЕКТНАЯ ЧАСТЬ

## Проектирование и реализация базы данных системы

Для корректного построения базы данных, которая будет использоваться в данной системе, будет логично начать с построения модели данных или ER-модели.

ER-модель – модель данных или схема вида «сущность-связь». Данная модель предназначена для отображения сущностей, присутствующих в системе, взятой для моделирования, а также связей между ними.

Так как мы будем использовать реляционную базу данных в нашей системе, все связи будут в виде «один-ко-многим», это значит, что на объект в первой таблице может ссылаться множество объектов из второй, но объект из второй таблицы может ссылаться лишь на один объект из первой.

Реляционная база данных – база данных, объекты которой могут иметь связи с объектами в других таблицах данной базы.



Рисунок 2 – ER-модель БД

Разработанная ER-модель состоит из 9-и сущностей: транзакция, клиент, список продукции, продукция/условие, условие хранения, место/условие и место хранения.

Сущности “Транзакция”, “Клиент”, “Список продукции”, “Тип транзакции” и “Сотрудник” предназначены для выполнения следующих задач базы данных:

* Хранение контактной информации о клиенте
* Отслеживание текущей продукции на складе
* Отслеживание транзакций склада, то есть не только получение продукции, но и её отправка
* Отслеживание даты поступления продукции, так как предполагается использовать разные способы поступления-отчисления, в том числе зависящие от даты поставки
* Также присутствует необязательное поле срока годности, необязательное оно так как предполагается, что склад имеет возможность хранить совершенно разные виды продукции

Сущности “Условие хранения”, “Место хранения”, “Место/условие” “Продукция/условие”, предназначены для реализации возможности учёта расположения продукции на складе в зависимости от особенностей хранения продукции.

После проектирования ER-модели была создана база данных в MS SQL - server. Microsoft SQL Server - система управления реляционными базами данных (РСУБД), разработанная корпорацией Microsoft. Основной используемый язык запросов - Transact-SQL, создан совместно Microsoft и Sybase. Transact - SQL является реализацией стандарта ANSI/ISO по структурированному языку запросов (SQL) с расширениями. В MS SQL - server были установлены типы данных для каждой таблицы.

На рисунке 3 представлена физическая схема БД, реализованная на платформе MS SQL Server 2017. На модели отражены названия атрибутов и их типы данных на сервере. Так же на модели отображены связи между сущностями для обеспечения целостности данных.

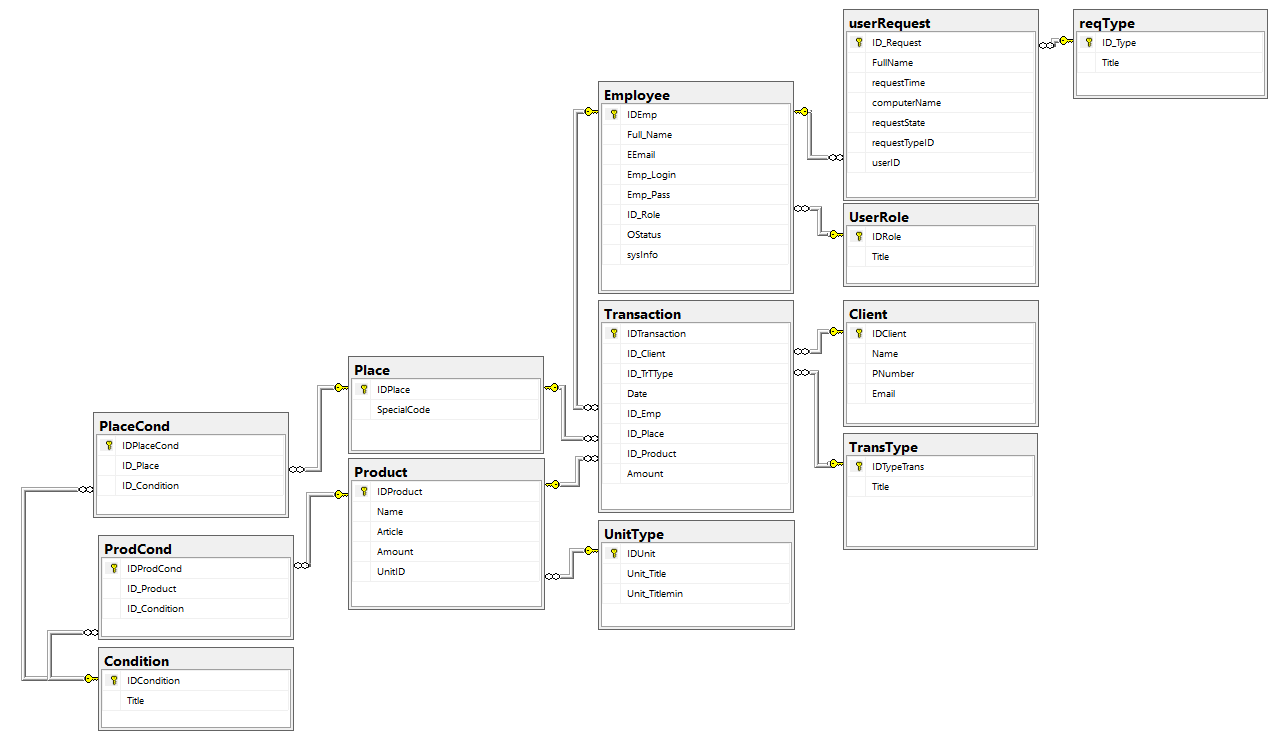


Рисунок 3 – Физическая модель БД

## Проектирование и реализация пользовательского интерфейса системы

Для реализации пользовательского интерфейса была выбрана интегрированная среда разработки Visual Studio 2019.

Visual Studio - линейка продуктов компании Microsoft, включающих интегрированную среду разработки программного обеспечения и ряд других инструментальных средств. Данные продукты позволяют, разрабатывать как консольные приложения, так и приложения с графическим интерфейсом. В данном проекте будет использована технология WPF.

WPF – система для построения пользовательского интерфейса в основе которой находится векторная система визуализации, что даёт возможность не зависеть от разрешения устройства на котором используется данный интерфейс.

Преимущества Visual Studio по сравнению с аналогичными программными продуктами это:

* высокая популярность позволяет находить решение проблем быстрее;
* низкие требования разработанного приложения к ресурсам компьютера;
* удачная проработка иерархии объектов;
* частично бесплатная;
* удобна в использовании.

Так же был использован язык программирования C# так как:

* также имеет высокую популярность и поддержку Майкрософт;
* в последнее время много совершенствуется;
* высокое удобство написания кода, что уменьшает возможное количество ошибок;
* прост, но имеет высокую эффективность и перспективы в развитии;

## Структура приложения

Для реализации поставленных задач была выбрана клиент-серверная архитектура приложения.

Клиент-серверная архитектура — это вычислительная или сетевая архитектура, в которой задания или сетевая нагрузка распределены между поставщиками услуг, называемыми серверами, и заказчиками услуг, называемыми клиентами. Фактически клиент-сервер — это программное обеспечение. Обычно эти программы расположены на разных вычислительных машинах и взаимодействуют между собой через вычислительную сеть посредством сетевых протоколов, но они могут быть расположены также и на одной машине.

Данная архитектура обладает рядом преимуществ:

* Отсутствие дублирования кода программы-сервера программами-клиентами.
* Так как поиск данных выполняется на сервере, путём отправки запроса к БД, то требования к компьютерам, на которых установлен клиент, снижаются.
* Все данные хранятся на сервере, который, как правило, защищён гораздо лучше большинства клиентов.
* На сервере проще организовать контроль полномочий, чтобы разрешать доступ к данным только клиентам с соответствующими правами доступа.

Сервер БД

Клиент

Рисунок 4 – Схема клиент-серверного приложения

Разработанное клиентское приложение выполняет все поставленные задачи, и имеет сложную архитектуру, представленную на Рисунке 5.



Рисунок 5 – Схема приложения

На схеме разными цветами выделены разграничения прав доступа. Зелёным цветом выделены функции, доступные администратору, красным и синим цветом отмечены функции доступные менеджеру и сотруднику соответственно, черным выделены функции доступные для всех.

## Реализация пользовательских форм

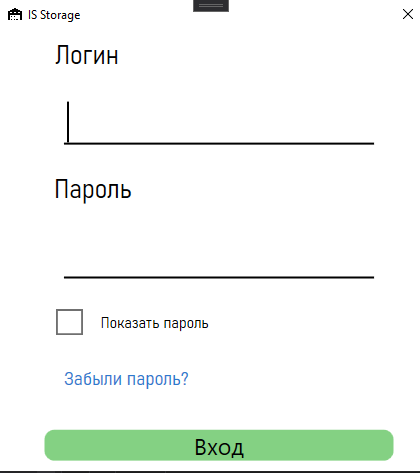
 В процессе выполнения курсового проекта было создано приложение в среде разработки Microsoft Visual Studio и база данных в Microsoft SQL Server manager.

Рисунок 6 – Окно авторизации

Авторизоваться можно с 2 сотрудников, 1 менеджера и 1 админа. Так же присутствует возможность восстановления пароля при помощи почты или подтверждения администратора.

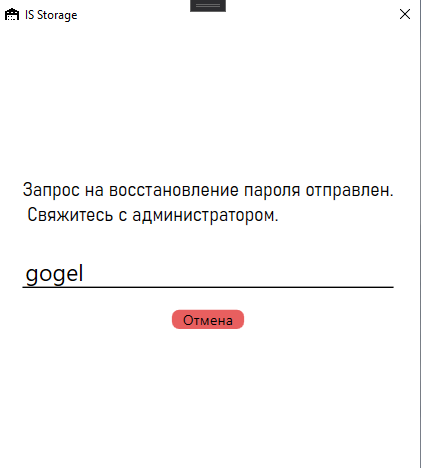


Рисунок 7 – Запрос восстановления пароля без почты

Код авторизации:

public static int passwCheck(string passw, string login)

{

try

{

var passCheck = stockEntities.GetStockEntityD().Employee.Where(p => p.Emp\_Login == login).FirstOrDefault();

if (stockEntities.GetStockEntity().userRequest.Where(p => p.userID == passCheck.IDEmp && p.requestState == 0 && p.requestTypeID == 1).FirstOrDefault() != null) return 2;

if (passCheck.Emp\_Pass == "-") return 1;

if (passCheck.Emp\_Login.Contains("\_\_\_")) return 3;

if (passCheck.OStatus) return 4;

passw = Sha256password(passw);/\*входящий пароль кодируется\*/

if (passw == passCheck.Emp\_Pass /\*пароль в базе\*/ )

return 0;

else return -1;

}

catch { MessageBox.Show("Пользователя с таким логином не существует!"); return -1; }

}

После успешной авторизации открывается окно, содержание которого зависит от того, какой вид пользователя произвёл вход.

Пользователю любого вида доступно редактирование своей учетной записи в системе. Все изменения связанные с базой данных записываются в список запросов.

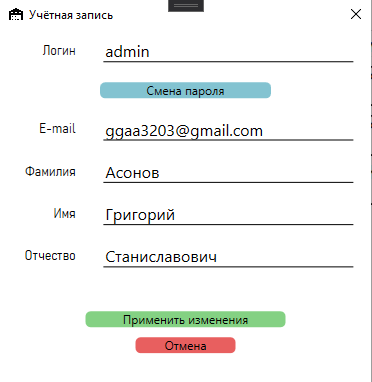


Рисунок 8 – Просмотр учётной записи

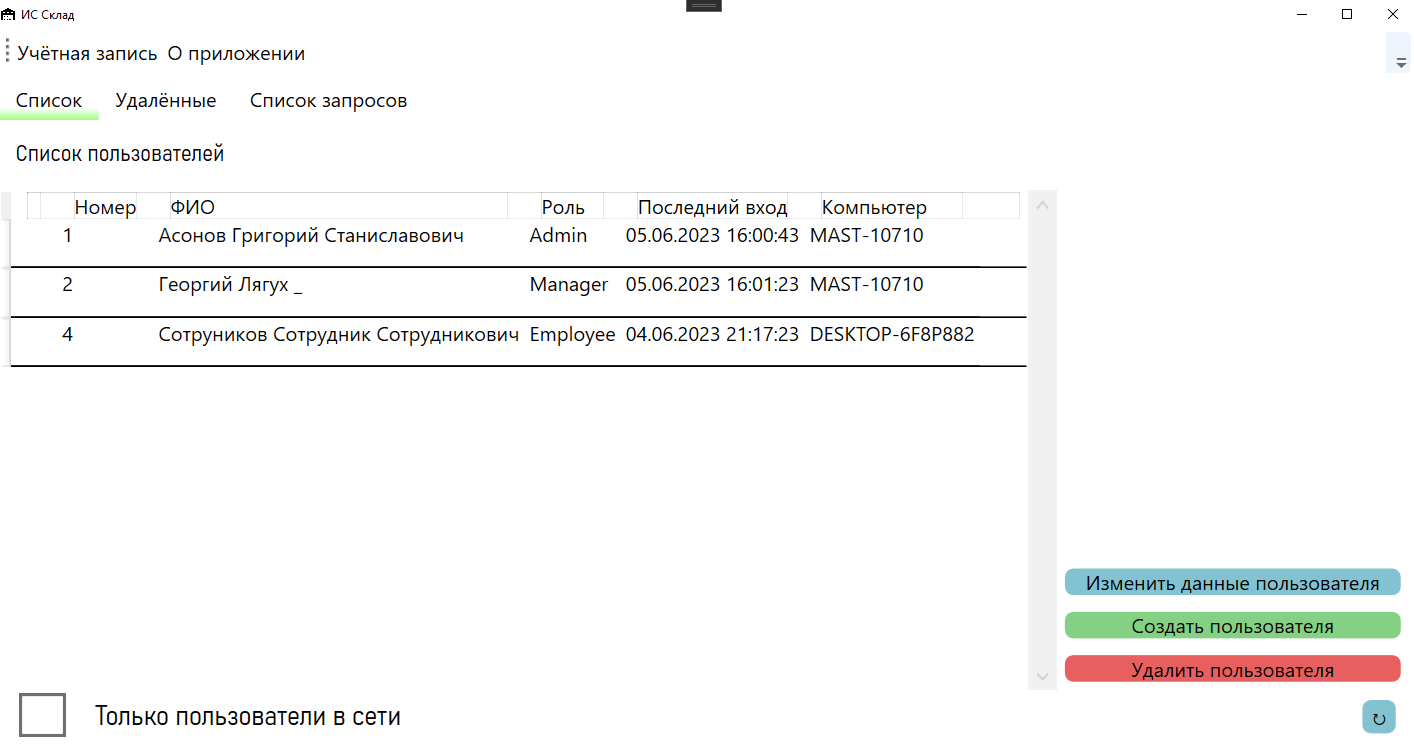


Рисунок 9 – Список пользователей

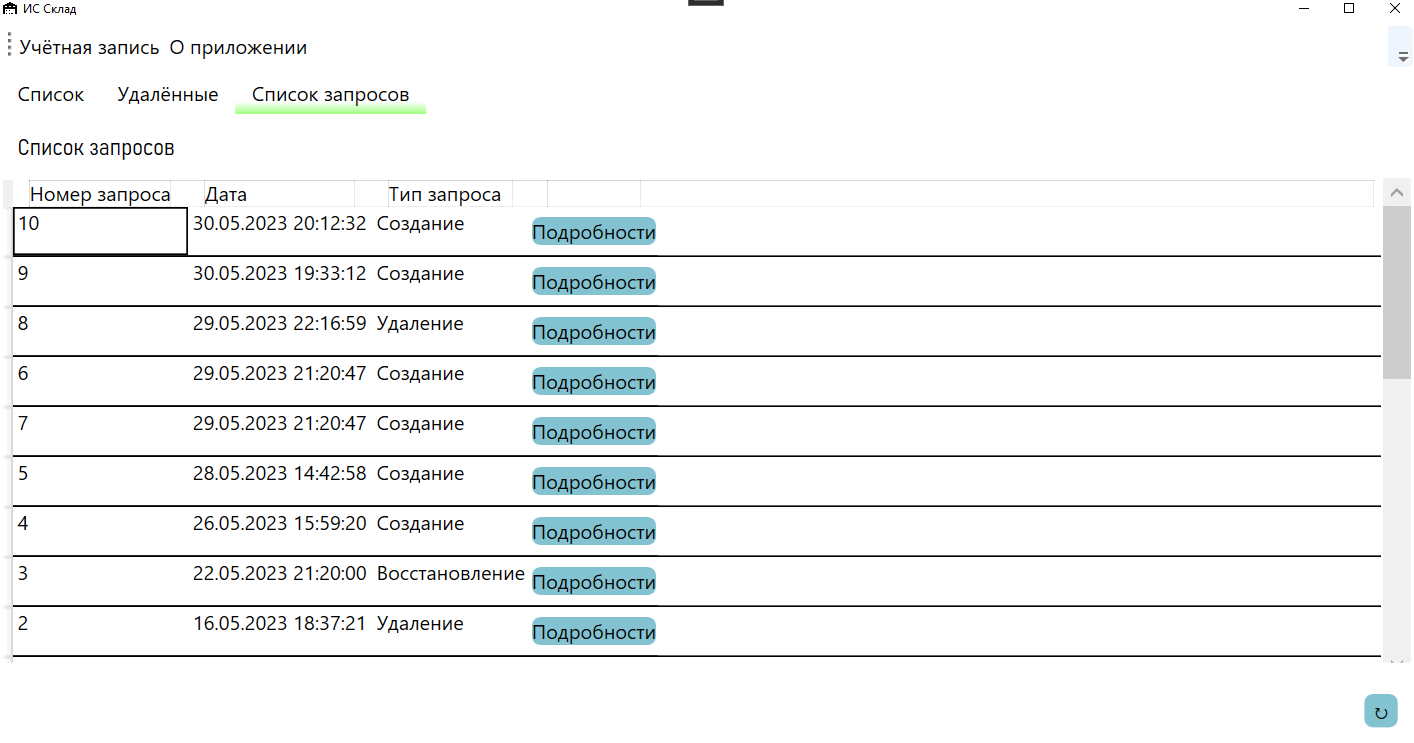


Рисунок 10 – Список запросов

Администратору доступно редактирование учетных записей других пользователей, просмотр пользователей, работающих в данный момент в системе, удалять и добавлять пользователей, а также просматривать список запросов и одобрять запросы на восстановление пароля.

Пользователю с ролью менеджера доступно редактирование, добавление и изменение клиентов, просмотр движений, происходивших на складах, импорт/экспорт клиентов, просмотр наполнения мест хранения и отправление отчётов клиентам по почте, эти отчёты в виде файлов word можно сохранять на компьютере менеджера.



Рисунок 11 – Список клиентов

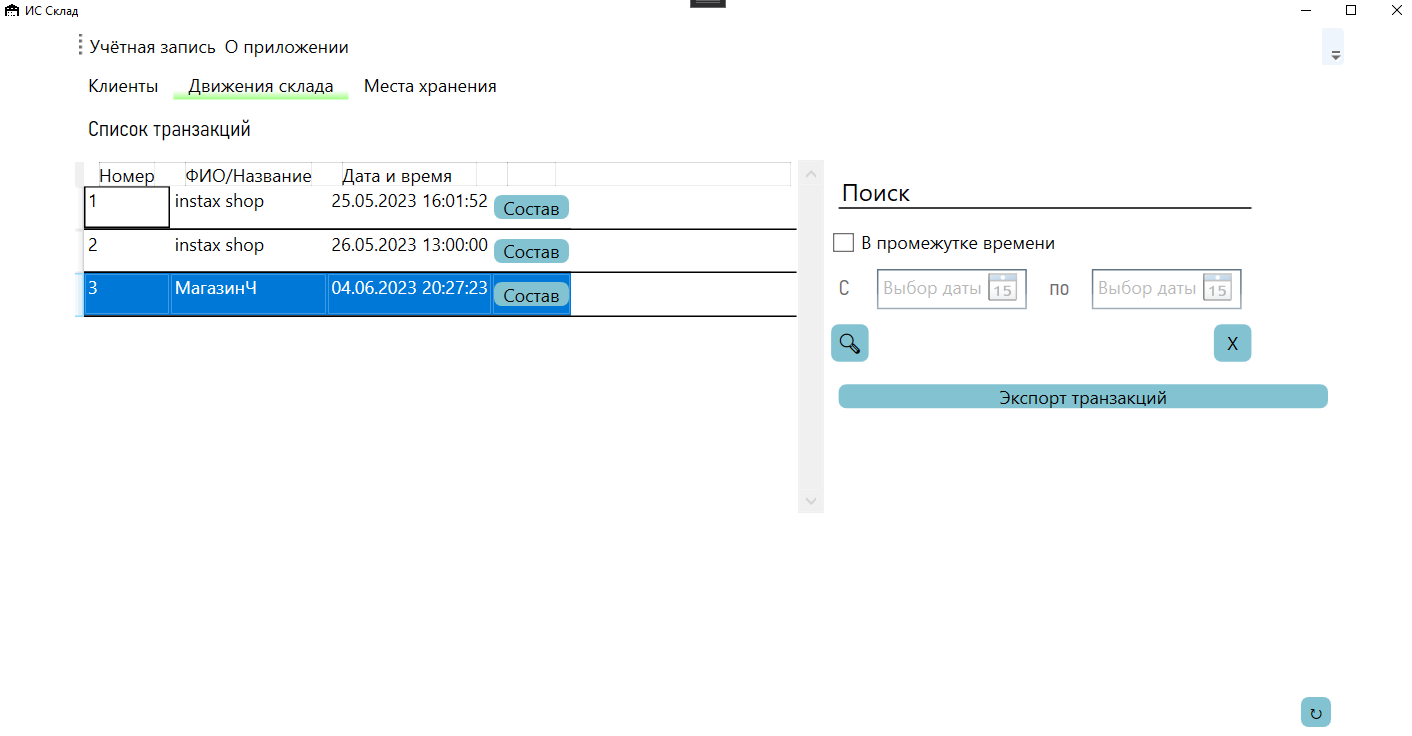


Рисунок 12 – Список транзакций

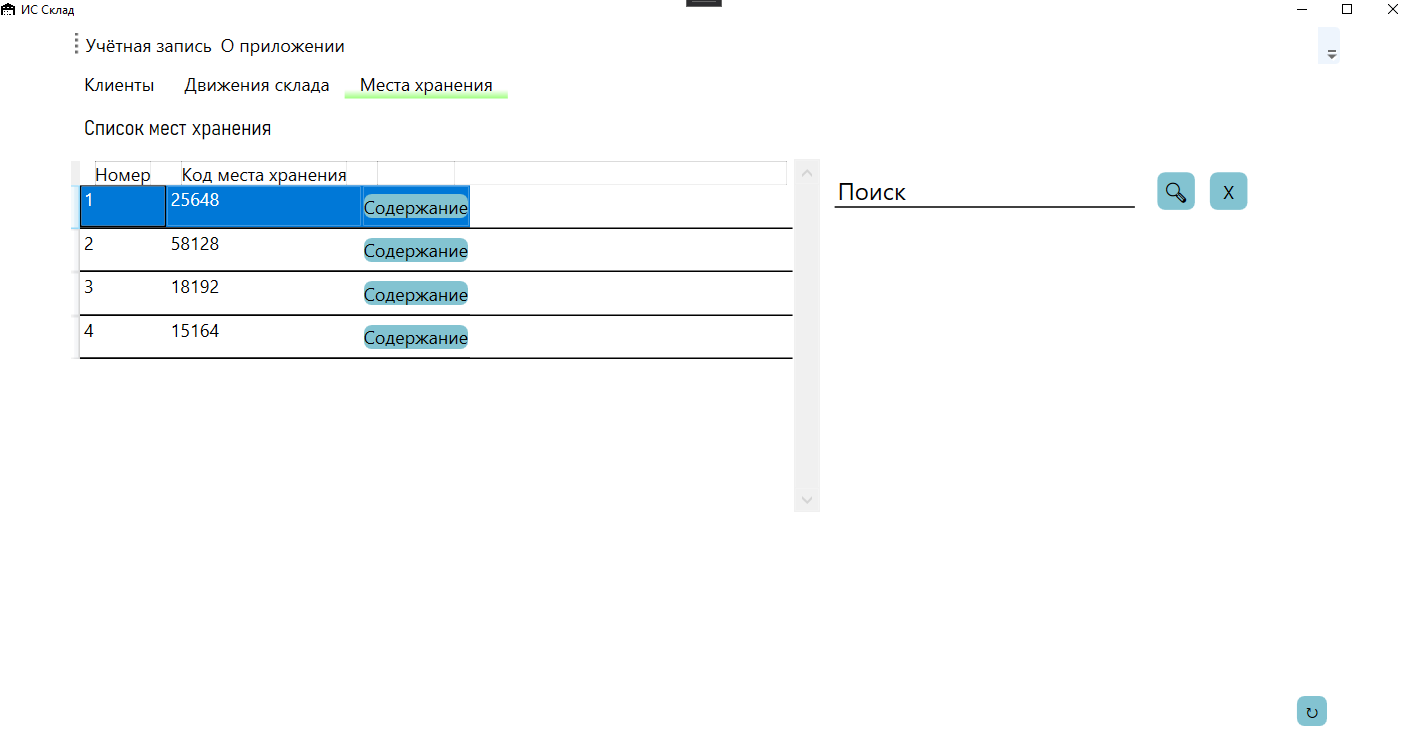


Рисунок 13 – Список мест хранения

Поиск и запись хранимого клиентом, поступившего в транзакции или на конкретном месте производится в одном методе:

public static List<Product> ProductsSearch(transactionControll transaction = null, Client cl = null, Place place = null, int type = 0)

{

stockEntities localCont = stockEntities.GetStockEntityD();

List<Transaction> tr = localCont.Transaction.ToList();

if (transaction != null) tr = transaction.actualList;

if (cl != null && cl != new Client()) tr = tr.Where(p => p.ID\_Client == cl.IDClient).ToList();

if (place != null) tr = tr.Where(p => p.ID\_Place == place.IDPlace).ToList();

List<Product> products = new List<Product>();

foreach (Transaction t in tr.Where(p => p.ID\_TrTType == 1).ToList())

{

if (products.Where(p => p.IDProduct == t.ID\_Product).Count() > 0)

{

products.Find(p => p.IDProduct == t.ID\_Product).Amount += t.Amount;

}

else

{

products.Add(localCont.Product.Where(p => p.IDProduct == t.ID\_Product).First()); products.Last().Amount = t.Amount;

}

}

foreach (Transaction t in tr.Where(p => p.ID\_TrTType == 2).ToList())

{

if (products.Where(p => p.IDProduct == t.ID\_Product).Count() > 0) products.Find(p => p.IDProduct == t.ID\_Product).Amount -= t.Amount;

}

if (type == 1)

foreach (Product pr in localCont.Product) if (products.Where(p => p.IDProduct == pr.IDProduct).Count() == 0) { products.Add(localCont.Product.Where(p => p.IDProduct == pr.IDProduct).First()); products.Last().Amount = 0; }

return products.Where(p => !p.Name.Contains("\_\_\_")).ToList();

}

Сотруднику доступно добавление или отмена транзакций на складе, добавление и удаление мест и условий хранения, а также добавление, изменение и удаление продукции.

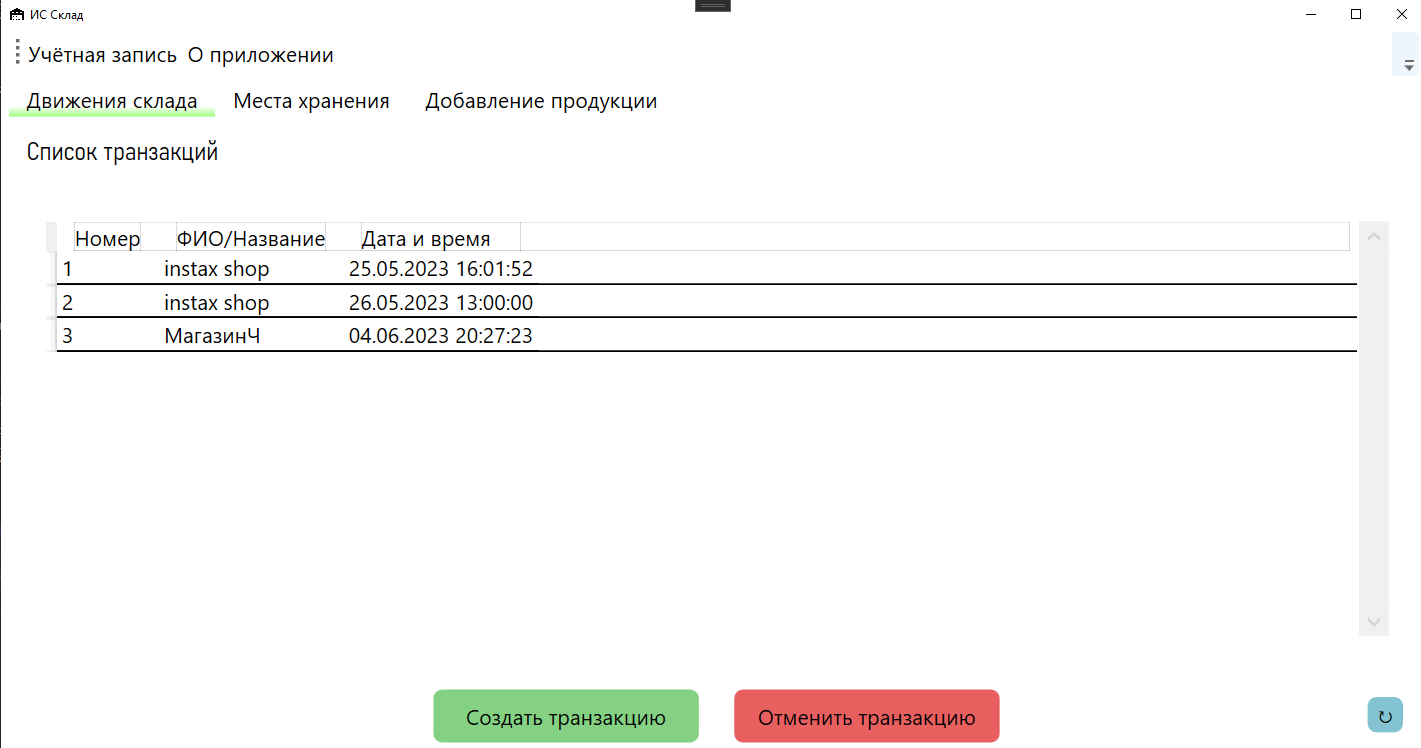


Рисунок 14 – Список транзакций и кнопки для работы с транзакциями

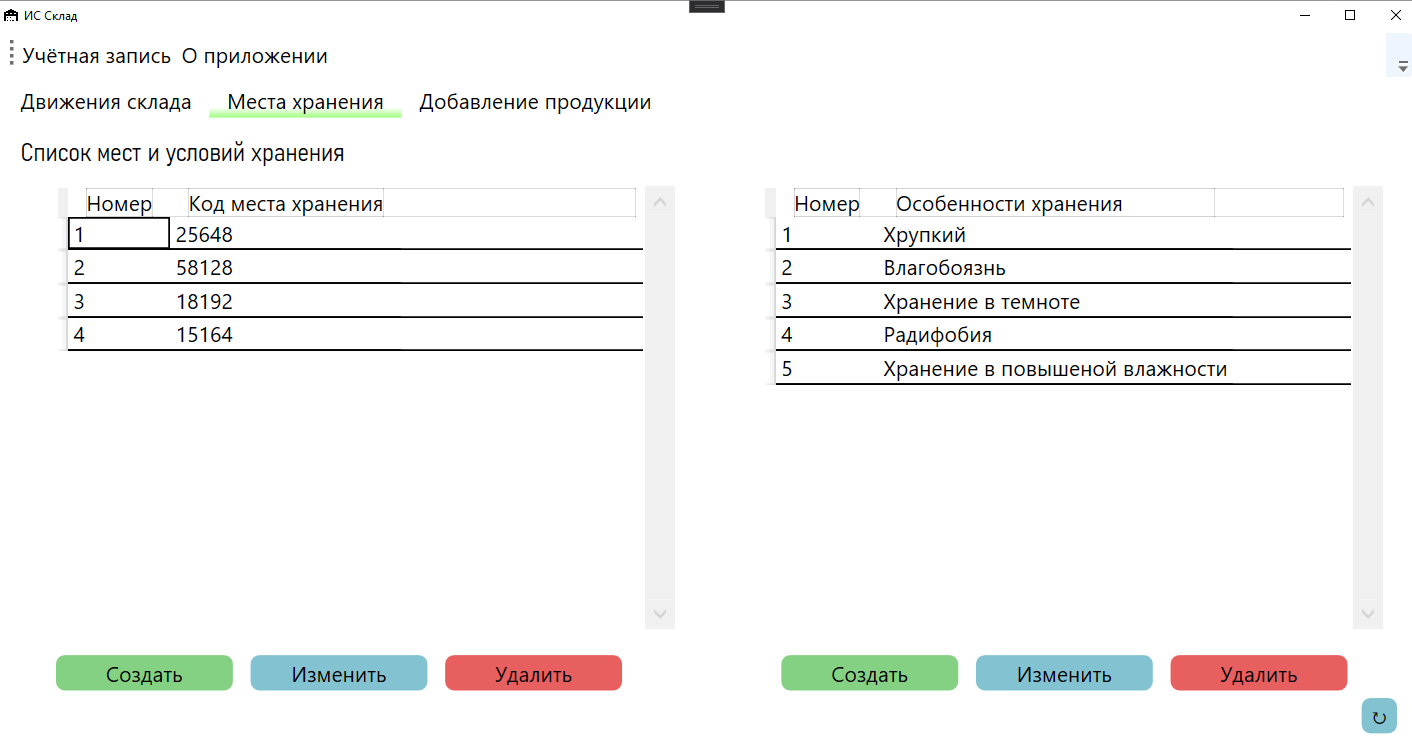


Рисунок 15 – Список мест хранения и условий

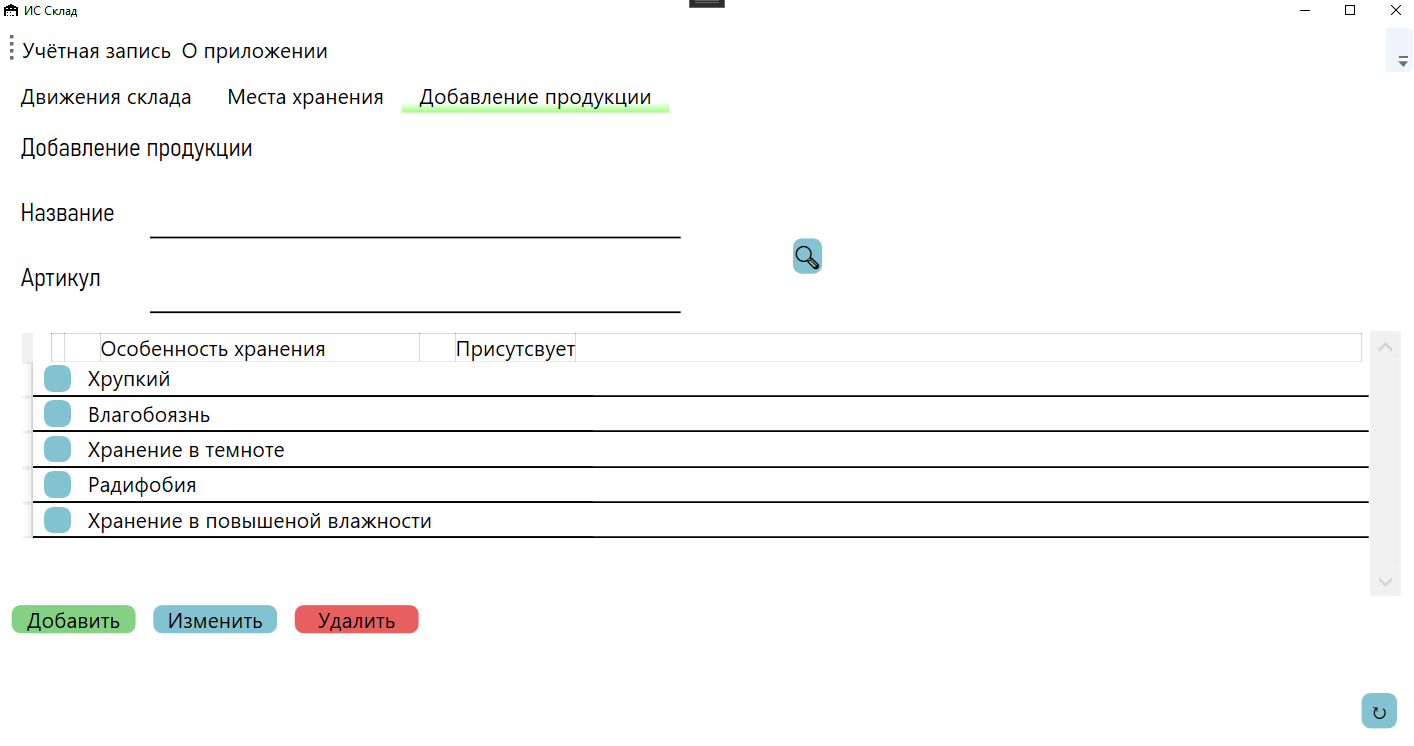


Рисунок 16 – Форма работы с продукцией

## Логирование

Каждое действие пользователя записывается в базу данных для дальнейшего просмотра администратором. Данная функция называется логированием.

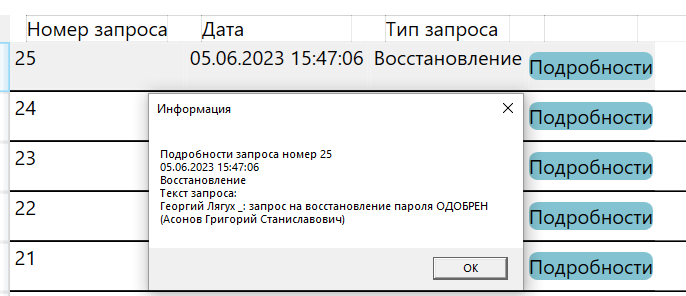


Рисунок 17 - Подробности запроса

Запрос генерируется в каждой процедуре отдельно, но сама генерация имеет следующий вид:

stockEntities.GetStockEntity().userRequest.Add(new userRequest

{

requestTypeID = 3,

FullName = employee.Full\_Name + a.actions,

requestState = 1,

requestTime = DateTime.Now.ToString("G"),

computerName = Environment.MachineName + " " + Environment.UserName,

userID = employee.IDEmp

});

Создание и изменение учётной записи пользователей происходит на одинаковых формах, но отличаются функционалом, так при изменении форма заполняется начальными данными и не позволяет изменять логин пользователя.

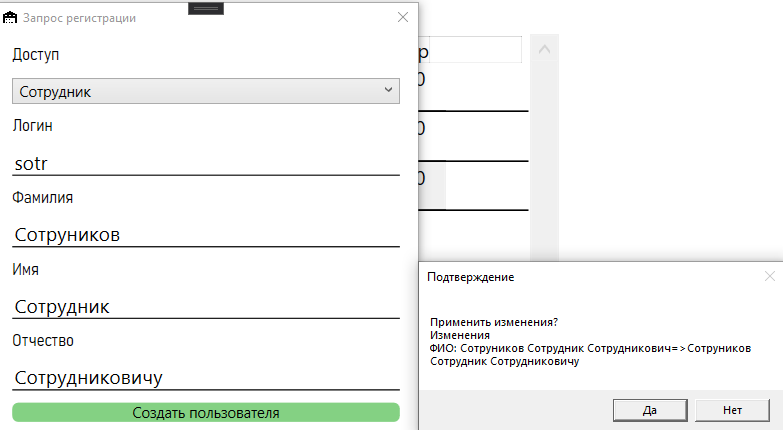


Рисунок 18 – Изменение данных пользователя

Логированию подвергаются следующие операции:

* Создание, удаление и изменение учётных записей администратором
* Запросы на восстановление паролей
* Создание, удаление и изменение клиентов менеджером
* Создание и отправка клиентам документов менеджером
* Создание, удаление и изменение данных мест хранения сотрудником
* Создание, удаление и изменение условий хранения сотрудником
* Создание, удаление и изменение продукции сотрудником
* Создание, отмена и изменение транзакций сотрудником

## Формирование документов

Как уже было сказано, менеджер имеет возможность создания документов наполнения склада по месту хранения, клиенту или отдельной транзакции. Данный документ сохраняется в формате .docx и может быть отправлен миную без сохранения на локальный компьютер непосредственно на почту клиента, если документ связан с клиентом или отдельной транзакцией.

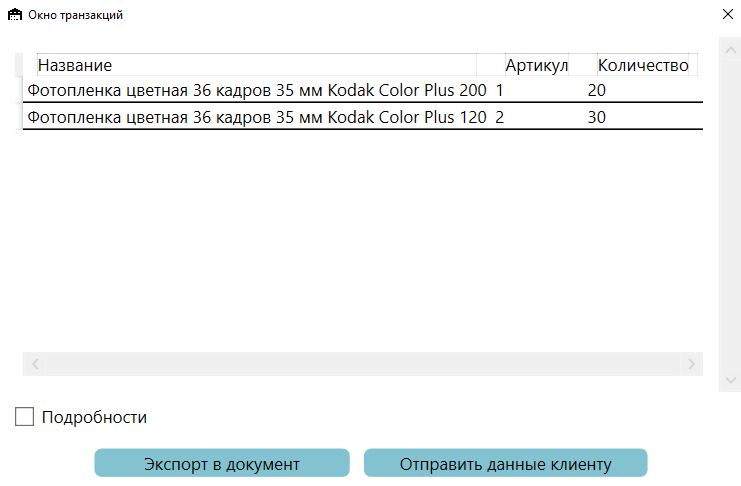


Рисунок 19 – Форма работы с транзакцией

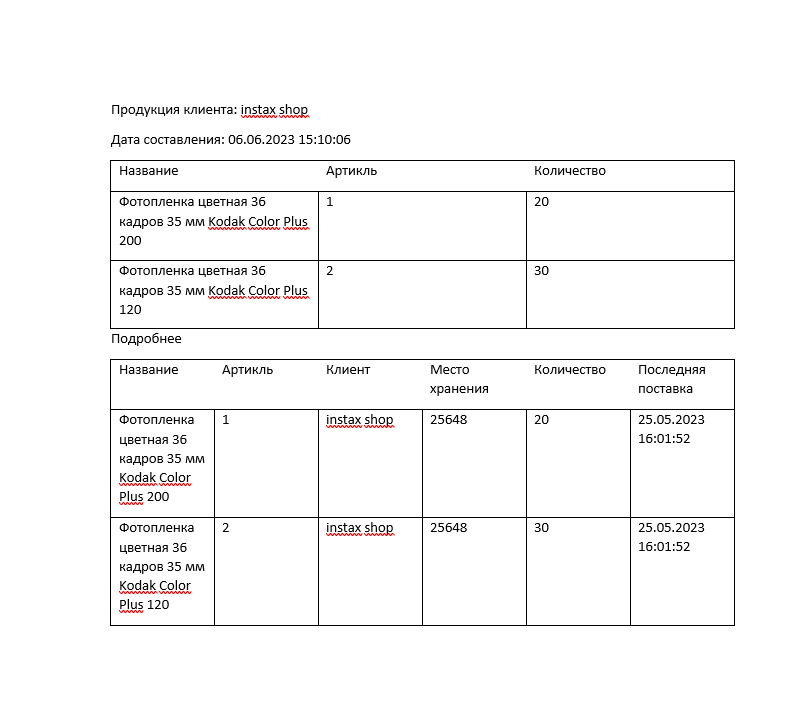


Рисунок 20 – Форма работы с транзакцией

## Работа с транзакциями

Работа с транзакцией разделена на два окна: добавление движений отдельной продукции и непосредственно список всех движений в одной транзакции

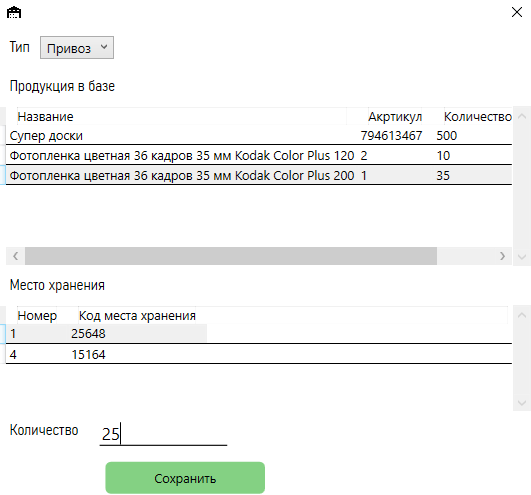


Рисунок 21 – Добавление движения на складе

Код добавления движения в базу данных:

private void addButton\_Click(object sender, RoutedEventArgs e)

{

if (productsClientGrid.SelectedItem!=null)

{

if (placeGrid.SelectedItem != null)

{

Place cp = (Place)placeGrid.SelectedItem;

Product p = (Product)productsClientGrid.SelectedItem;

if (amountTxt.Text != "0" || amountTxt.Text != "")

{

if (p.Amount<Convert.ToDouble(amountTxt.Text)&&typeTr.SelectedIndex == 1)

{

if (MessageBox.Show("У клиента не хватает выбранной продукции, добавить максимальное количество?", "Подтверждение", MessageBoxButton.YesNo) == MessageBoxResult.Yes)

amountTxt.Text = p.Amount.ToString();

}

if (MessageBox.Show("Сохранить?", "Подтверждение", MessageBoxButton.YesNo) == MessageBoxResult.Yes)

{

controll = new Transaction()

{

ID\_Client = cl.IDClient,

Amount = Convert.ToDouble(amountTxt.Text),

Date = controll.Date != null ? controll.Date : DateTime.Now.ToString("G"),

IDTransaction = controll.IDTransaction != 0 ? controll.IDTransaction : 0,

ID\_Emp = cEmp.IDEmp,

ID\_Place = cp.IDPlace,

ID\_Product = p.IDProduct,

ID\_TrTType = typeTr.SelectedIndex == 0 ? 1 : 2,

Product = p,

Place = cp

};

DialogResult = true;

}

}

else MessageBox.Show("Введите корректное количество!");

}

else MessageBox.Show("Выберите место хранения!");

}

else MessageBox.Show("Выберите продукцию!");

}

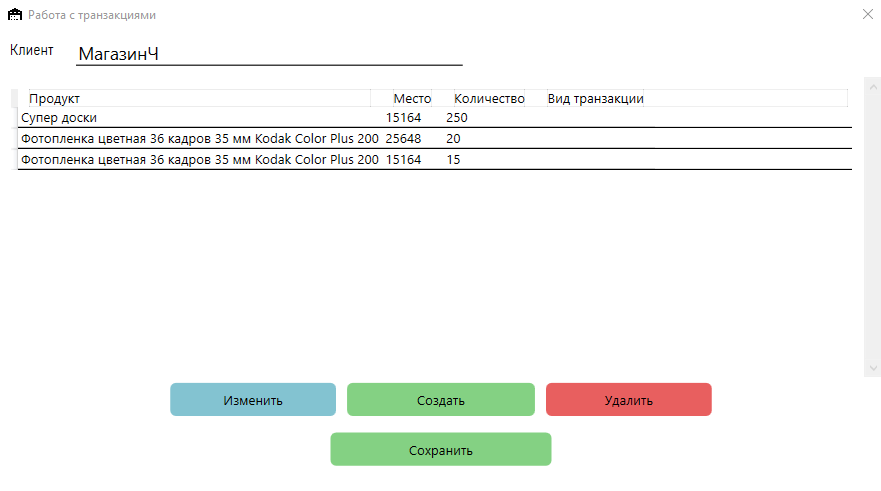


Рисунок 22 – Общий вид транзакции

## Меню и справка

В самом верху рабочего пространства программы, независимо от роли пользователя находиться ToolBar – элемент предоставляющий возможность хранить кнопки и надписи для общего пользования. В нём можно найти клавишу “Учётная запись”, представленную на рисунке 8 и информацию о программе, которая выводится как окно уведомления, представленное на рисунке 22.

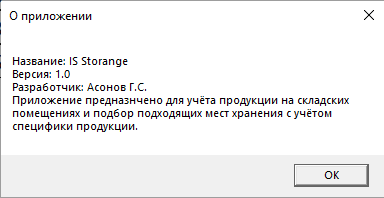


Рисунок 23 – О программе

# ЗАКЛЮЧЕНИЕ

В заключении курсового проекта большинство поставленных целей было выполнено. Дизайн самого приложения минималистичен и после недолгого изучения становится крайне удобным. Функционал приложения требует дополнения и тестирования, но основные задачи приложение выполняет.

Приложение, разработанное в рамках данного курсового проекта, может быть использовано для ведения учётов на складах и иных местах хранения.

Главной целью данного приложения является хранение и удобное отображение данных хранимого на местах хранения, составление отчётов и получение информации о происходящих движениях на местах хранения.

Для реализации приложения были использованы следующее программное обеспечение:

* Visual Studio 2022
* MS SQL Server Management Studio
* MS SQL Server

Приложение способно на выполнение следующих задач:

* Ведение учёта продукции на складе
* Вывод данных о продукции в разных контекстах
* Создание документов и отправление их по электронной почте

В ходе выполнения курсового проекта была более углублённо изучена работа c Visual Studio 2022, MS SQL Server, Entity Framework и запросами LINQ.

# Список литературы

1. Microsoft .NET информация о smtp клиенте, предназначенного для отправления электронных писем приложением - <https://learn.microsoft.com/ru-ru/dotnet/api/system.net.mail.smtpclient?view=net-8.0>
2. Microsoft .NET информация о backgrounworker - класс, позволяющий вывести операции в дополнительный поток - <https://learn.microsoft.com/ru-ru/dotnet/api/system.componentmodel.backgroundworker?view=net-7.0>
3. Википедия функции, предназначения и принципы хранения на складе - https://ru.wikipedia.org/wiki/%D0%A1%D0%BA%D0%BB%D0%B0%D0%B4