Федеральное государственное автономное
образовательное учреждение
высшего образования
«СИБИРСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»
Институт космических и информационных технологий
институт
Информационных систем
кафедра

УТВЕРЖДАЮ
Заведующий кафедрой ИС
_____ Дьячук П.П.
подпись фамилия, инициалы
« 21 » июня 2019 г.

БАКАЛАВРСКАЯ РАБОТА

09.03.02 Информационные системы и технологии

Автоматизация процесса учета материалов на складе

Руководитель _____ канд. техн. наук, доцент И.А. Легалов
подпись, дата ученая степень, должность инициалы, фамилия

Студент КИ15-13Б 031509052 _____ И.В. Дышлюк
номер группы номер зачетной книжки подпись, дата инициалы, фамилия

Нормоконтролер _____ Ю.В. Шмагрис
подпись, дата инициалы, фамилия

Красноярск 2019

Федеральное государственное автономное
образовательное учреждение
высшего образования
«СИБИРСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»
Институт космических и информационных технологий
институт
Информационных систем
кафедра

УТВЕРЖДАЮ
Заведующий кафедрой ИС
_____ Дьячук П.П.
подпись фамилия, инициалы
« » 2019 г.

ЗАДАНИЕ
НА ВЫПУСКНУЮ КВАЛИФИКАЦИОННУЮ РАБОТУ
в форме бакалаврской работы

Студенту: Дышлюку Ивану Владиславовичу

Группа: КИ15-13Б Направление: 09.03.02 «Информационные системы и технологии»

Тема выпускной квалификационной работы: Автоматизация процесса учета материалов на складе

Утверждена приказом по университету № 7237/с от 24.05.2019 г.

Руководитель ВКР: И. А. Легалов, канд. техн. наук, доцент кафедры «Информационные системы» ИКИТ СФУ

Исходные данные для ВКР: Требования к разрабатываемому приложению, учебные материалы, рекомендации руководителя

Перечень разделов ВКР: Введение, анализ предметной области, проектирование приложения, реализация приложения, инструкция работы с приложением, заключение, список сокращений, список использованных источников.

Перечень графического материала: презентация, выполненная в Microsoft Office PowerPoint 2013.

Руководитель ВКР _____ И. А. Легалов

подпись

Задание принял к исполнению _____ И.В. Дышлюк

подпись

« » _____ 2019 г.

РЕФЕРАТ

Выпускная квалификационная работа «Автоматизация процесса учета материалов на складе» содержит 42 страницы текстового документа, 21 использованных источников, 25 листов графического материала.

DESKTOP – ПРИЛОЖЕНИЕ, Microsoft Visual Studio, C#, MVP, MS SQL, трехуровневая архитектура, Use Case, NET Framework.

Объектом исследования является склад.

Предметом исследования является процесс учета материалов на складе.

Цель работы – автоматизация процесса учета материалов на складе.

Для достижения поставленной цели были выдвинуты следующие задачи:

1. провести анализ предметной области;

2. рассмотреть аналоги приложения;

3. определить инструментальные средства, которые будут использоваться в работе;

4. разработать бизнес-логику и интерфейс;

5. описать работу полученной в результате системы.

В результате выполнения выпускной квалификационной работы было разработано desktop-приложение «Склад».

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Использования складов берет свое начало во времена развития земледелия и продажи различных благ. Возникла необходимость где-то хранить запасы, при этом соблюдать условия, при которых эти запасы смогут сохранить свое изначальное состояние. В Древнем Египте сельскохозяйственные товары хранились в зернохранилищах и являлись единственной разновидностью складов в то время. Впоследствии были внедрены складские операции, которые требовали новых исполнителей.

Сам принцип хранения заключается в правильном расположении и укладке материалов на складе в отдельных зонах склада. Рациональным размещением материала считается правильное использования пространства в секторах хранения, но при этом стоит учесть нормальные условия работы специальной техники и механизмов. Рациональное использование складского пространства позволяет максимизировать прибыль и уменьшить расходы на хранение.

Эффективность предприятия на сегодняшний день напрямую зависит от правильной организации хранения материальных ценностей на складе. Для учета материалов на складе возможно использование бумажной отчетности. Данный вариант позволяет сэкономить время и деньги на разработку автоматизированной системы. В свою очередь, автоматизация позволяет повысить эффективность работы, снизить потери материалов и трудозатраты на выполнение операций на складе.

Процесс хранения материалов на складе начинается после приемки и перемещения товара на склад. Основным требованием хранения является обеспечение условий для сохранения целостности материала. Материалы, размещенные на складе, сортируются по размеру, весу, времени и условию хранения, а также особенностям, которые не позволяют их содержать с другими объектами.

Цель настоящей выпускной квалификационной работы – автоматизация процесса учета материалов на складе.

Для достижение данной цели необходимо решить следующие задачи:

- изучить область решаемой задачи;
- определить инструментальные средства для разработки;
- спроектировать приложение;
- реализовать приложение;
- описать работу разработанной системы.

1 Анализ предметной области

1.1 Описание предметной области

Склад – это специальное помещение для хранения материальных ценностей и организации складских услуг. На сегодняшний день функционирование любого склада можно разделить на три больших процесса, такие как:

1. поступление товара на склад;
2. хранение и учет;
3. выдача товара.

Перед поступлением товара на склад, поставщику создается заказ на необходимое количество товара. После сделки товар попадает на склад, где необходимо его принять, занести информацию о товаре в базу данных и расположить товар на складе. Перед тем, как расположить товар на склад, на него наклеивается этикетка с характеристиками товара, а также с местом хранения на складе.

Хранение и учет материалов на складе включает в себя транспортировку и инвентаризацию. В ходе процесса инвентаризации происходит проверка соответствия между фактическими характеристиками товара с характеристиками, заявленными в документации. Товар может быть перемещен как внутри помещения, так и на другой склад.

Выдача товара включает различные операции расходования товаров, например, отгрузка клиенту, списание брака, списание на внутренние нужды. Выдача товара происходит только после формирования документа об выдачи. В нем хранятся такие сведения о товаре, как:

- наименование товара;
- номер товара в системе;
- количество;

- цена продажи;
- дата продажи;
- наименование получателя.

1.2 Описание аналогов

Сервисы автоматизированного учета материалов на складе можно разделить на 2 большие группы:

1. программы для покупки и скачивания;
2. облачные сервисы.

У первой группы можно выделить программу – «СуперСклад». С помощью нее можно обеспечить полноценный учет товаров и денег от киоска до крупного предприятия. История создания данной программы начитается 1993. В 2016 году был запущен облачный аналог «СуперСклад.Облако». Основное меню «СуперСклад» показано на рисунке 1.

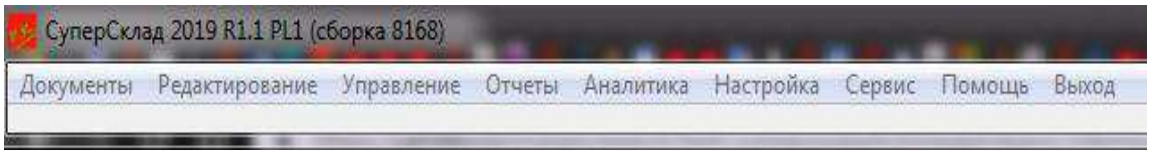

Рисунок 1 – Основное меню «СуперСклад»

«СуперСклад» имеет большую функциональность, в том числе анализ бизнес-процессов, а также простой интерфейс, который будет понятен любому пользователю. Приложение фиксирует все произведенные изменения товарного или денежного документа. При необходимости возможен экспорт отчетов в «MS Excel» для дальнейшего анализа. Данное приложение может быть установлена на жесткий диск компьютера или на съемный носитель. Стоимость приложения зависит от функционала и необходимости помощи техподдержки. Интерфейс составления приходных накладных показан на рисунке 2.

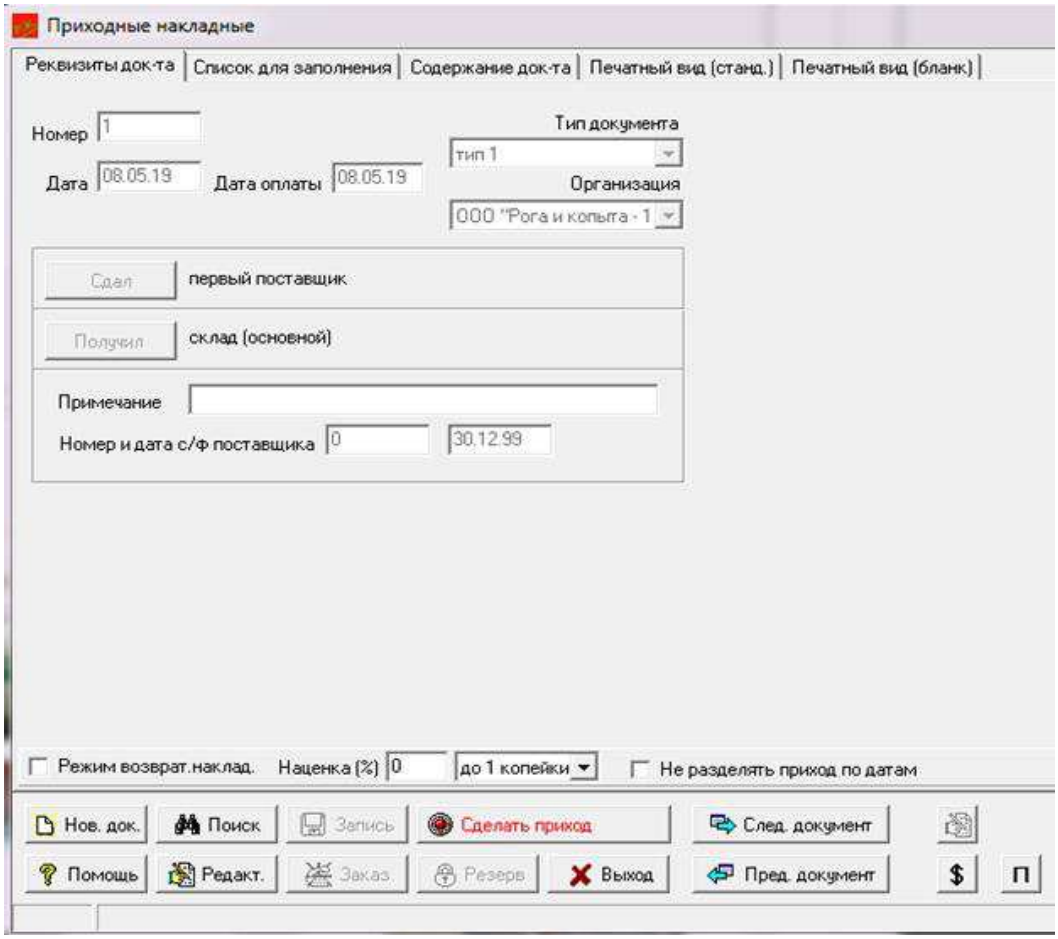

Рисунок 2 – Составление приходных накладных «СуперСклад»

Ко второй группе относится популярный облачный сервис «МойСклад», который берет свое начало в 2008 году и на сегодняшний день насчитывает более 400000 компаний.

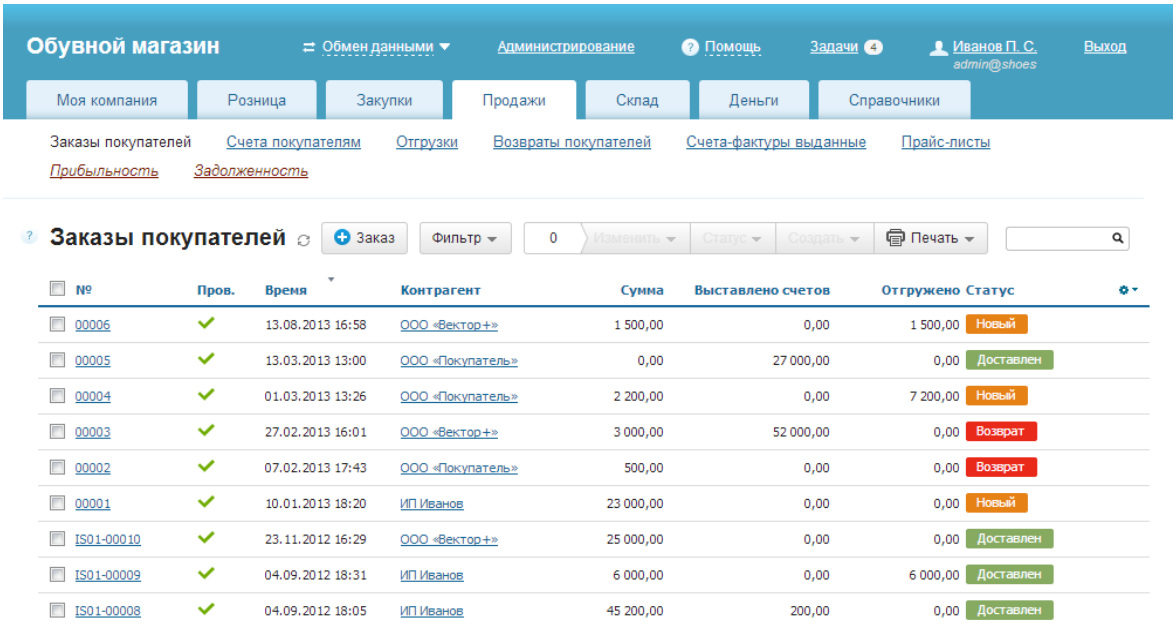

Рисунок 3 – Облачный сервис «МойСклад»

Данный сервис позволяет, без установки программ на ПК, автоматизировать малый и средний бизнес. «МойСклад» имеет большой функционал, в том числе работу с базой покупателей(CRM). Возможно объединение в одной системе несколько графически удаленных офисов, магазинов или складов. Изначально дается пробный период для ознакомления, далее цена зависит от размера бизнеса. Одна из CRM возможностей изображена на рисунке 4.

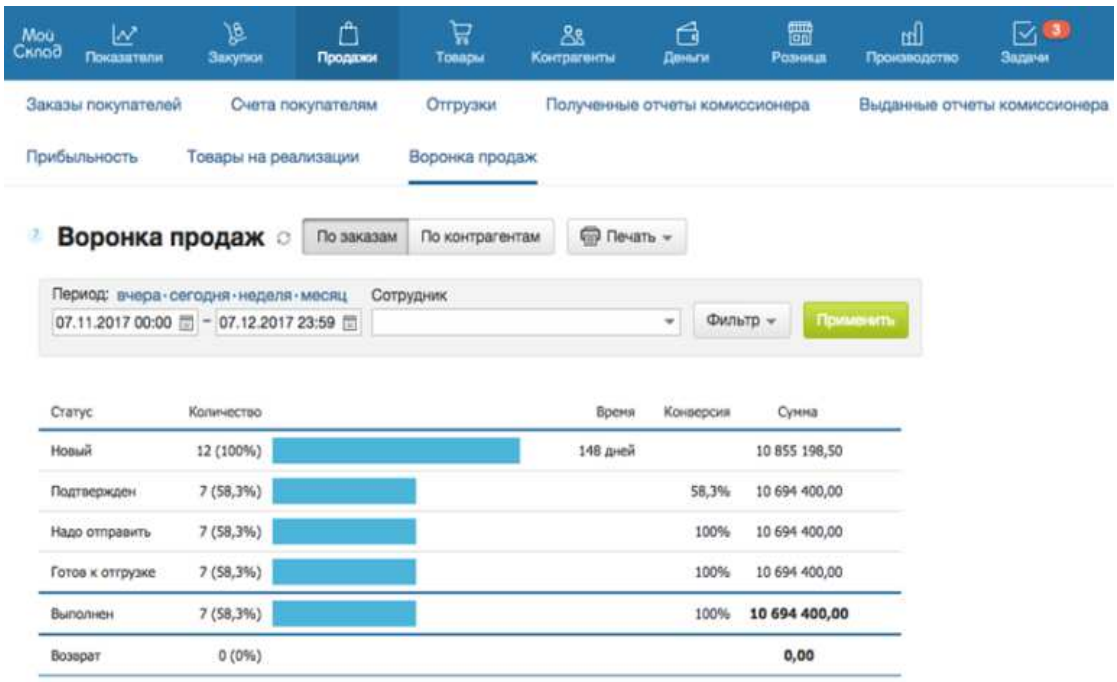

Рисунок 4 – Одна из CRM возможностей сервиса «МойСклад»

Стоит упомянуть довольно популярный в России «1С: Торговля и склад», который является составной частью «1С: Предприятие». Основной интерфейс представлен на рисунке 5.

Рисунок 5 – Основное меню в «1С: Торговля и склад»

Данный сервис имеет свои достоинства, такие как:

- автоматизация оформления первичных документов торгового и складского учета

- возможность адаптации функционала путем включения/отключения различных функциональных опций

- возможность работы с торговым оборудованием, например, с чековыми принтерами

Из минусов стоит отметить сложность в освоении.

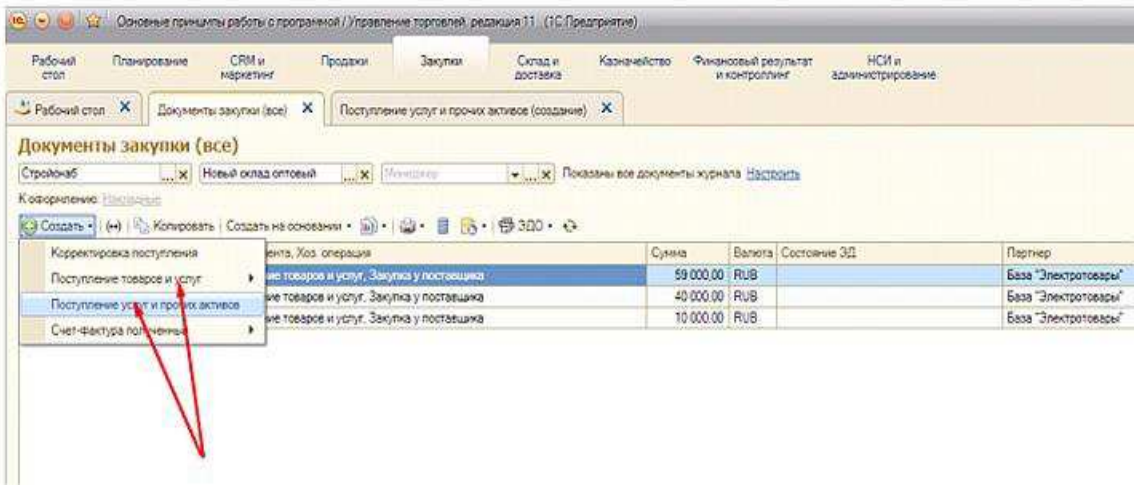

Рисунок 6 – Меню создания приходного документа в «1С: Торговля и склад»

1.3 Требования к функционалу приложения

Использование автоматизированной системы учета материалов должно повысить эффективность работы, снизить потери материалов и трудозатраты на выполнение операций на складе.

При автоматизации необходимо обеспечить:

- авторизация в приложении;
- быстрый и удобный интерфейс для работы пользователя;
- организованное хранение материалов;
- всевозможные операции с материалами, которые находятся на складе;
- возможность автоматического составления отчета.

На данном этапе разработки приложения существует две роли:

- администратор;
- кладовщик.

Цели, которые необходимо достичь при автоматизации, направлены на уменьшение времени, затрачиваемого персоналам на составление различных документов, а также уменьшение ошибок из-за человеческого фактора.

2 Проектирование приложения

2.1 Используемые технологии

Для разработки информационной системы автоматизации процесса учета материалов на складе использовался объектно-ориентированный язык программирования C#. В качестве интегрированной среды разработки использовалась Microsoft Visual Studio. Для создание БД, а также для различных операций с ней была выбрана СУБД MS SQL Server.

Выбор СУБД MS SQL Server обоснован удобством работы с ней в Microsoft Visual Studio (средства для подключения, генерации и выполнения запросов и т.д.).

2.1.1 Язык программирования C#

В 2000 г., благодаря усилиям Андерса Хейлсберга появился язык C#, который был основан на языке программирования Delphi и среде разработки Turbo Pascal. В настоящий момент — это мощный язык программирования, с помощью которого можно написать практически любое приложение. Также этот язык подойдет для людей, которые только осваивают программирование.

C# является объектно-ориентированным. Объектно-ориентированный подход позволяет решить задачи по построению крупных, но в тоже время гибких, масштабируемых и расширяемых приложений. C# поддерживает наследование, полиморфизм, перегрузку операторов, статическую типизацию.

Особенности C#:

- полная поддержка классов и объектно-ориентированного программирования, включая наследование интерфейсов и реализаций, виртуальных функций и перегрузки операторов;
- полный и хорошо определенный набор основных типов;
- встроенная поддержка автоматической генерации XML-документации;
- автоматическое освобождение динамически распределенной памяти;

- возможность отметки классов и методов атрибутами, определяемыми пользователем;

- полный доступ к библиотеке базовых классов .NET, а также легкий доступ к Windows API;

- указатели и прямой доступ к памяти, если они необходимы. Однако язык разработан таким образом, что практически во всех случаях можно обойтись и без этого;

- простое изменение ключей компиляции. Позволяет получать исполняемые файлы или библиотеки компонентов .NET, которые могут быть вызваны другим кодом.

2.1.2 Microsoft Visual Studio

Microsoft Visual Studio — линейка продуктов компании Майкрософт, включающих интегрированную среду разработки программного обеспечения и ряд других инструментальных средств. Первая версия была выпущена в 1997 году.

Основные компоненты Visual Studio – редактор исходного кода, встроенный отладчик, дизайнеры классов, редактор форм для создания графического интерфейса и так далее.

Microsoft Visual Studio позволяет разрабатывать как консольные приложения, так и приложения с графическим интерфейсом, в том числе с поддержкой технологии Windows Forms. Возможности Visual Studio могут быть расширены за счёт использования сторонних плагинов.

Visual Studio включает в себя редактор исходного кода с поддержкой технологии IntelliSense и возможностью простейшего перепроектирование кода. Встроенный отладчик может работать как отладчик уровня исходного кода, так и отладчик машинного уровня. Остальные встраиваемые инструменты включают в себя редактор форм для упрощения создания графического интерфейса приложения, дизайнер классов и дизайнер схемы базы данных, веб-редактор.

2.1.3 .NET Framework

.NET Framework — программная платформа, выпущенная компанией Microsoft в 2002 году.

Основной идеей при разработке .NET Framework являлось обеспечение свободы разработчика за счёт возможности создавать приложения различных типов, способные выполняться на различных типах устройств и в различных средах. Второй идеей стала ориентация на системы, работающие под управлением семейства операционных систем Microsoft Windows.

Хотя .NET является патентованной технологией корпорации Microsoft и официально рассчитана на работу под операционными системами семейства Microsoft Windows, существуют проекты (например, это Mono и Portable.NET), позволяющие запускать программы .NET на других операционных системах.

На рисунке 7 изображены технологии, которые входят в .NET Framework.

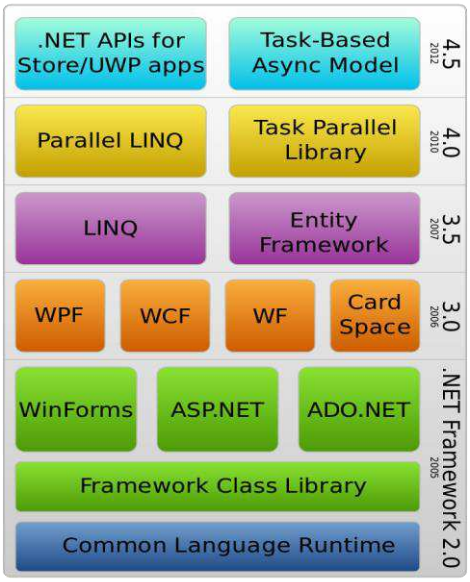

Рисунок 7 – Стек технологий .NET Framework

2.1.4 Microsoft SQL Server

MS SQL Server — система управления базами данных (СУБД), которая использует язык Transact, разработанная компанией Microsoft. Первая версия вышла в 1987 году.

Основным аспектом в MS SQL Server, как и в любой СУБД, является база данных. База данных – это некоторое хранилище данных, организованных определенным способом. Нередко физически база данных представляет из себя файл на жестком диске, хотя такое соответствие необязательно.

MS SQL Server использует реляционную модель для организации баз данных. Данная модель базы данных была разработана в 1970 году Эдгаром Коддом и Кристофером Дейтом. Сущность реляционной модели данных заключается в том, что хранение данных происходит в виде таблиц, которая состоит из строк и столбцов. Каждая строка хранит отдельный объект, например, карандаш, а в столбцах размещаются атрибуты этого объекта, например, цвет, размер. На сегодняшний день MS SQL Server фактически является стандартом для организации баз данных. Однако создатели реляционной модели данных Эдгар Кодд, Кристофер Дейт и их сторонники говорят о том, что SQL не является таковым по ряду причин:

- повторяющиеся строки;
- неопределённые значения (NULL);
- использование указателей;
- явное указание порядка колонок слева направо и т.д.

Для идентификации каждой строки в таблице применяется первичный ключ (primary key). В качестве первичного ключа может выступать один или несколько столбцов. Используя первичный ключ, мы можем ссылаться на определенную строку в таблице. Соответственно две строки не могут иметь один и тот же первичный ключ.

Через ключи одна таблица может быть связана с другой, то есть между двумя таблицами могут быть организованы связи. А сама таблица может быть представлена в виде отношения (relation).

Для взаимодействия с базой данных применяется язык SQL (Structured Query Language). Клиент (например, внешняя программа) отправляет запрос на языке SQL посредством специального API. СУБД должным образом

интерпретирует и выполняет запрос, а затем посылает клиенту результат выполнения.

MS SQL Server в качестве языка запросов использует версию SQL, получившую название Transact-SQL. T-SQL позволяет использовать дополнительный синтаксис для хранимых процедур и обеспечивает поддержку транзакций (взаимодействие базы данных с управляющим приложением).

2.2 Архитектура приложения

Архитектура информационной системы – концепция, согласно которой взаимодействуют компоненты информационной системы.

Model-View-Controller – паттерн, при котором происходит разделение данных приложения, интерфейса и логики на три отдельных компонента таким образом, что каждый компонент может быть реализован независимо от других. Была описана Трюгве Реенскаугом в 1978 году. В последствии, на основе MVC были разработаны и другие паттерны, например, MVVM, MVP и др.

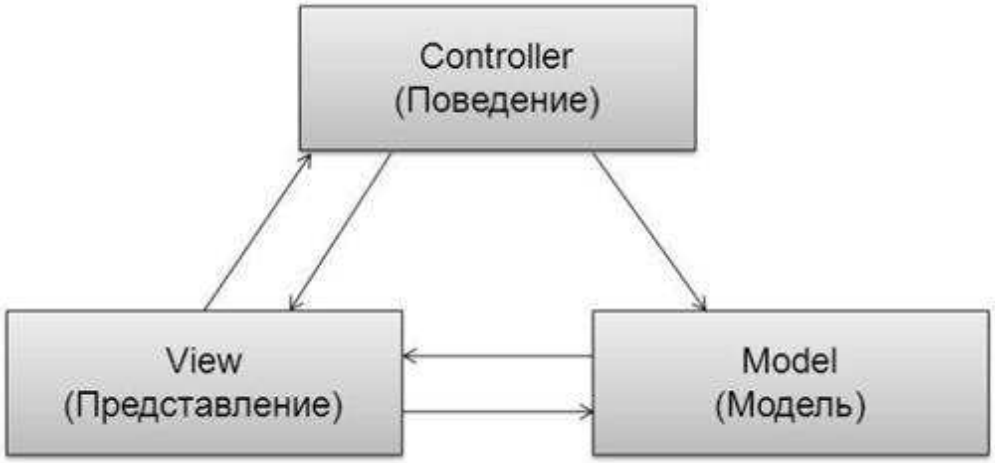

Рисунок 8 – Схема паттерна MVC

Model-View-Presenter – паттерн, производный от Model-View-Controller, который используется в основном для построения пользовательского

интерфейса. Основная идея этого паттерна в том, что и контроллер, и представление зависят от модели, но модель никак не зависит от этих двух компонент.

Элемент Model - это бизнес-логика приложения, т.е. это набор данных и методов их обработки. Модель ничего не должна знать о существовании Presenter и View. Model полностью независима.

Элемент View обеспечивает разные способы представления данных. Он может быть шаблоном, который заполняемым данными. View может быть несколько различных видов, и он обращается к Presenter за обновлением данных.

Элемент Presenter в данном шаблоне берёт на себя функциональность посредника (аналогично контроллеру в MVC) и отвечает за управление событиями пользовательского интерфейса (например, использование мыши) так же, как в других шаблонах обычно отвечает представление.

Данный подход позволяет создавать абстракцию представления. Реализован данный паттерн обычно при помощи создания интерфейсов представления. У каждого представления будут интерфейсы с определенными наборами методов и свойств, необходимых Presenter. Presenter же, в свою очередь, инициализируется с данным интерфейсом, подписывается на события представления и по срабатывании событий обновляет данные.

Рисунок 9 – Схема паттерна MVP

Признаки контроллера:

- Контроллер определяет, какие представление должны быть отображены в данный момент;

- События представления могут повлиять только на контроллер; контроллер может повлиять на модель и определить другие представления;

- Возможны создание нескольких представлений только для одного контроллера.

Представление жестко отделяется от модели, при этом связь организуется через Presenter. В отличие от MVC, MVP допускает повторное использование бизнес-логики за счет использования интерфейса IView, без непосредственного видоизменения модели.

2.3 Диаграмма вариантов использования

Диаграмма вариантов использования - это форма представления концептуальной модели проектируемой системы, которая описывает ее функциональное назначение и в дальнейшем будет детализирована, и преобразована в модели логического и физического уровней.

Разработка диаграммы вариантов использования предназначена для решения следующих задач:

- на начальных этапах проектирования системы определить общие границы и контекст предметной области;
- выявить общие требования к функциональному поведению разрабатываемой системы;
- разработать исходную концептуальную модель разрабатываемой системы для ее последующей детализации в форме модели логического и физического уровня;
- подготовить исходную документацию для взаимодействия разработчиков, заказчиков и пользователей.

Диаграмма строится базе трех основных компонентов: вариант использования (use case), действующее лицо или актер (actor) и связь или отношение (relationship).

Актеры" — это тип пользователей, которые взаимодействуют с системой.

Вариант использования (прецеденты) - набор некоторых действий, которые система предоставляет актерам, приводящие к некоторому результату.

Отношение – это семантическая связь между отдельными элементами модели. Существуют различные отношения между элементами диаграммы, которые показывают взаимосвязь между актерами и вариантами использования.

В языке UML имеется несколько стандартных видов отношений между актерами и вариантами использования:

1. ассоциации (Association) - служит для обозначения специфической роли актера при его взаимодействии с отдельным вариантом использования;

2. включения (Include) - отношением зависимости является такое отношение между двумя элементами модели, при котором изменение одного элемента (независимого) приводит к изменению другого элемента (зависимого);

3. расширения (Extend) - определяет взаимосвязь базового варианта использования с другим вариантом использования, функциональное поведение которого задействуется базовым не всегда, а только при выполнении дополнительных условий;

4. обобщения (Generalization) - Два и более актера имеют общие свойства, т. е. взаимодействуют с одним и тем же множеством вариантов использования одинаковым образом.

На рисунке 10 представлена UseCase диаграмма.

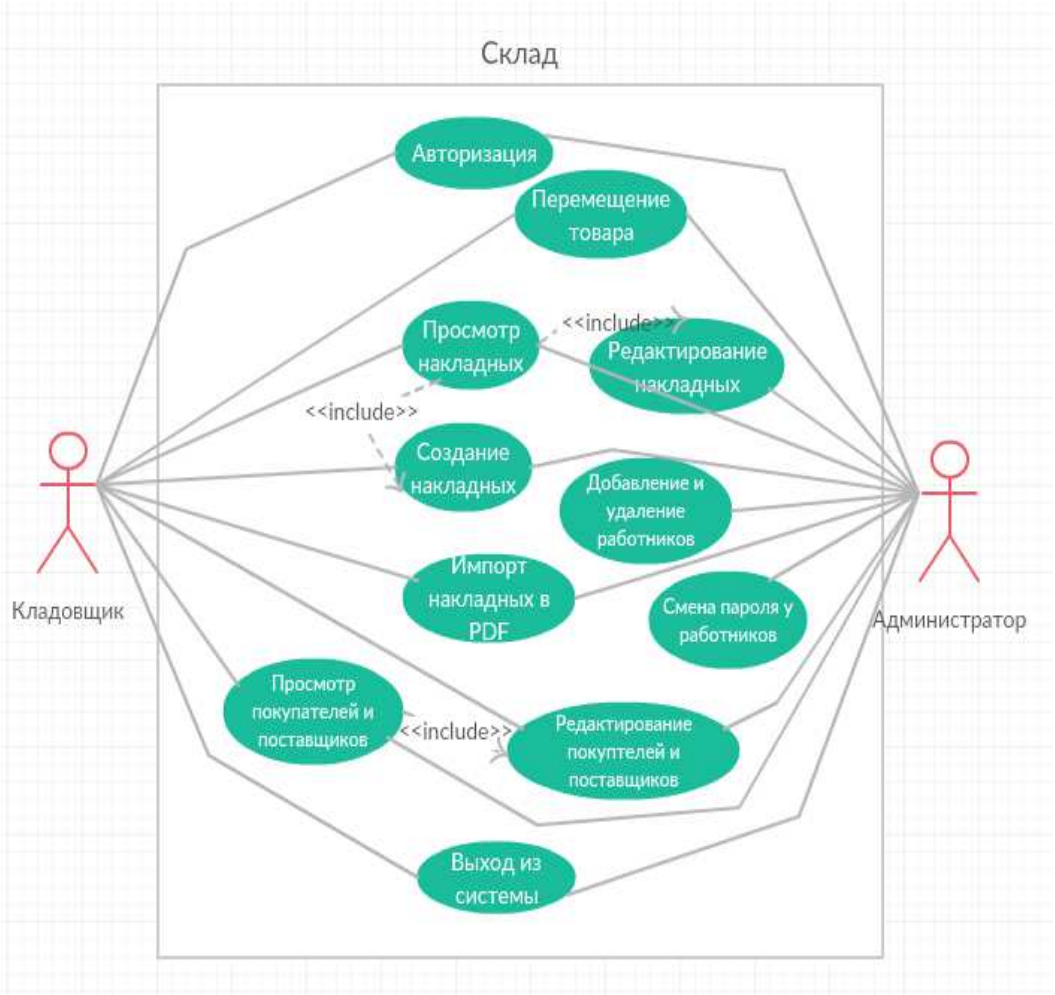

Рисунок 10 – Диаграмма UseCase Stock

На рисунке изображены 2 актера: Администратор и Кладовщик.

Основные функции, доступные Администратору и Кладовщику:

1. авторизация;
2. перемещение материала;
3. просмотр накладных;
4. создание накладных;
5. импорт накладных в PDF;
6. редактирование покупателей и поставщиков;
7. просмотр покупателей и поставщиков;
8. выход из системы.

Помимо основных функций, Администратору доступно:

1. редактирование накладных;

2. добавление, удаление работников;

3. смена пароля у работников.

2.4 Проектирование базы данных

Почти в каждой информационной системе присутствует база данных. База данных позволяет хранить данные в структурированном виде. БД состоит из таблиц – основных объектов базы данных, а также связей между ними. Таблицы состоят из полей – столбцы таблицы, и записей – строки таблицы.

Базы данных позволяет создавать запросы – это специальное средство, с помощью которого можно получить информацию с БД. Запросы могут быть отфильтрованы по определенным критериям.

Каждая таблица должна содержать в себе столбец или набор столбцов, которые однозначно определят каждую строку таблицы. Такой уникальной записью может стать номер паспорта, или код сотрудника.

Таблицы в базе данных могут быть связаны между собой, т.е. быть зависимыми. В реляционных базах данных существует три вида связей между таблицами:

- связь один к одному;
- связь один ко многим;
- связь многие ко многим.

Связь один ко многим реализуется тогда, когда одному объекту может принадлежать несколько объектов, но второму объекту может соответствовать только один объект. Например, у человека может быть несколько телефонов, но конкретный телефон может быть только у одного человека.

Связь многим ко многим реализуется тогда, когда нескольким объектам из таблицы может соответствовать несколько объектов из другой таблицы. Например, одна книга могла быть написана несколькими авторами, а эти авторы написали еще несколько.

Одна из самых редких связей – это связь один к одному. Она реализуется тогда, когда одному объекту таблицы соответствует один объект другой таблицы. Например, человек и паспорт.

Для корректной работы информационно системы необходимо предусмотреть авторизацию работников склада. Также правильным решением будет отслеживание физическое местонахождения материалов на складе для быстрого доступа к ним. При приеме и выдаче товара необходимо сохранить важную информацию, такую как дата выдачи, количество, стоимость, имя работника, который выдал или принял товар, а также оплачена ли данная операция. Чтобы реализовать это, необходимо хранить информацию в базе данных.

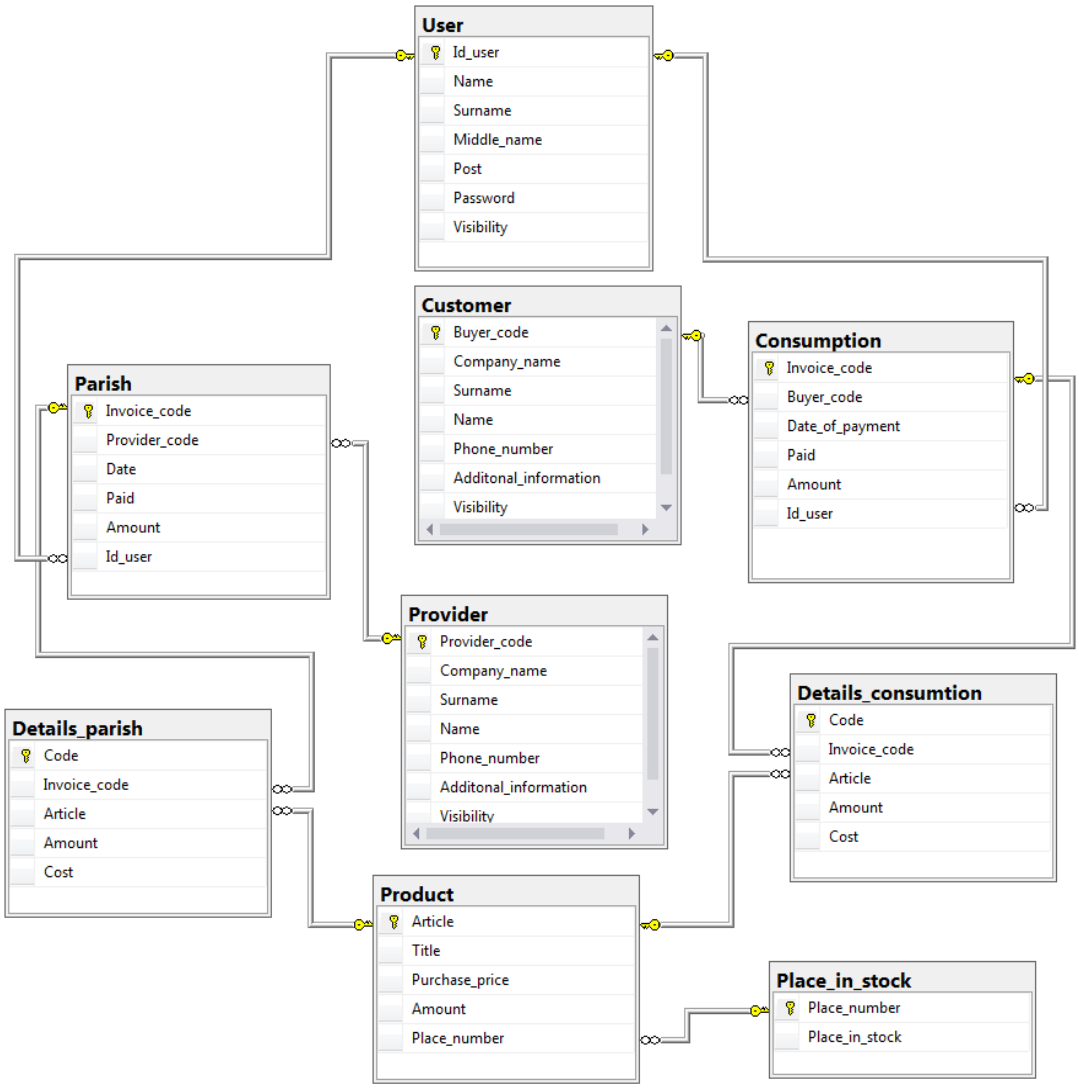

Рисунок 11 – Диаграмма БД

На рисунке 11 изображены 9 сущностей и связи между ними:

- User – информация о кладовщиках и администраторах;
- Consumption – информация о документе расхода;
- Details_consumtion – информация о расходе конкретного продукта;
- Product – информация о продукте на складе;
- Parish – информация о документе прихода;
- Details_parish – информация о приходе конкретного продукта;
- Provider – данные о поставщике;
- Customer - данные о покупателе;
- Place_in_stock – место хранения продукта.

3 Реализация приложения

3.1 Реализация Model

Model содержит бизнес-логику приложения, через Presenter получает или передает данные на View. Для соблюдения архитектурного паттерна MVP необходимо создать интерфейсы для работы с репозиторием, а также реализовать их. Фрагмент интерфейсов представлен на рисунке 12.

```
namespace ServiceLayer.Services
{
    public interface IUserRepository
    {
        void Add(UserModel userModel);
        void Delete(UserModel userModel);
        void Updete(UserModel userModel);
        UserModel GetById(int id);
        UserModel GetByLastName(int id);
        IEnumerable<UserModel> getAll();
    }
```

Рисунок 12 – Интерфейс «IUserRepository»

Model является своеобразным фасадом к некоторому источнику данных или к слою доступа к данным. Ее задача - загружать и сохранять данные согласно задачам конкретного экрана. На рисунке 13 изображен фрагмент кода чтения с БД таблицы «User».

```
List<LoginModel> getAll()
{
    string connStr = @"Data Source=(local)\SQLEXPRESS;
                    Initial Catalog=Stock;
                    Integrated Security=True";
    List<LoginModel>name = new List<LoginModel>();
    using (SqlConnection connection = new SqlConnection(connStr))
    {
        connection.Open();
        SqlCommand command = new SqlCommand("SELECT * FROM [User]", connection);
        SqlDataReader reader = command.ExecuteReader();

        while (reader.Read()) // построчно считываем данные
        {
            object family = reader.GetValue(3);
            object pass = reader.GetValue(6);
            name.Add(new LoginModel(family, pass));
        }
    }
    return name;
}
```

Рисунок 13 – Чтение данных о работниках с БД

Вначале создается строка подключение к БД. Первой строкой является имя сервера, второй название базы данных, третьей некоторые параметры безопасности. Далее создается список класса «LoginModel», в котором хранятся только фамилия и пароль пользователей. Дальше происходит чтение с БД с помощью класса «SqlConnection». Данный класс позволяет инкапсулировать sql-выражение, которое должно быть выполнено. В конце происходит сохранение данных в список «name». Данная конструкция «using» закрывает соединение с базой данных самостоятельно.

На рисунке 14 изображен метод, который возвращает «true» в случае совпадение данных, которые ввел пользователь, с данными, хранящихся в базе данных.

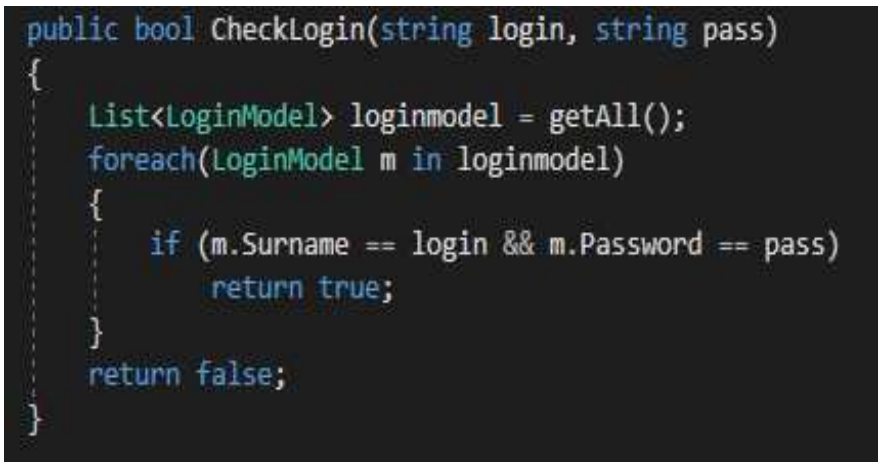

Рисунок 14 – Метод проверки

Данный метод принимает два параметра – логин и пароль, введенными пользователем. Метод «getAll» передает списку все записи, которые хранятся в таблице «User». Далее происходит проверка на совпадение логина и пароля.

3.2 Реализация View

View отвечает за визуализацию данных, которые были получены от Model. View создается с помощью интерфейса программирования приложений Windows Forms. Во View реализованы интерфейсы, с помощью которых Presenter общается с представлением. В проекте View реализовано 7 представлений, а соответственно 7 Presenter и Model.

Интерфейс входа в учетную запись представлен на рисунке 15.

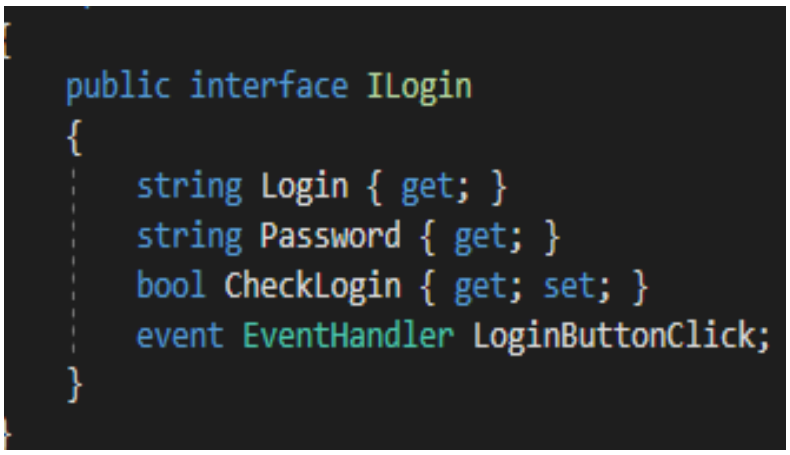

Рисунок 15 – Интерфейс класса «ILogin»

На рисунке 16 изображена часть кода реализации интерфейса.

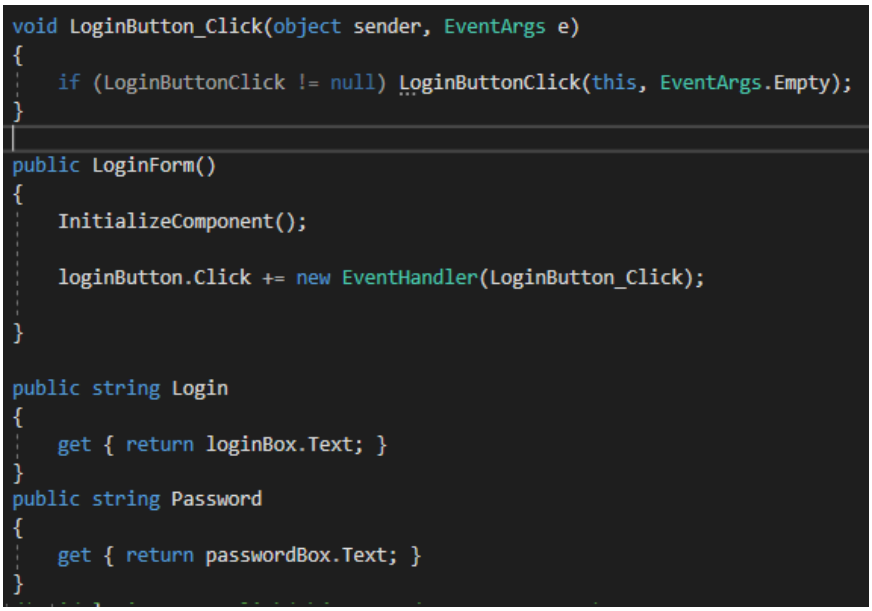

Рисунок 16 – Часть реализации интерфейса «ILogin»

Первый метод представляет из себя привязку события. Если пользователь нажал на кнопку «Войти», то вызывается данный метод и передает информацию о нажатии Presenter. Следующий метод – конструктор класса. Первая строчка инициализирует компоненты, а следующая привязывает событие к нажатию кнопки. Последние две реализации позволяют Presenter считывает логин и пароль, введенный пользователем.

3.3 Реализация Presenter

Presenter определяет поведение и структуру приложения, не зависит от конкретной реализации интерфейса. Presenter главным образом обеспечивает реализацию вариантов использования приложения, а также координирует взаимодействие пользователя с бизнес-логикой. На рисунке 17 изображена реализация Presenter для входа в систему.

```
public class PresenterLogin
{
    private readonly ILoginView _view;
    private readonly ILoginModel _model;

    public PresenterLogin(ILoginView view, ILoginModel model)
    {
        _view = view;
        _model = model;

        _view.LoginButtonClick += _view_LoginButtonClick;
    }

    private void _view_LoginButtonClick(object sender, EventArgs e)
    {
        _view.LoginCheck(_model.CheckLogin(_view.Login, _view.Password));
    }
}
```

Рисунок 17 – Реализация Presenter для входа в систему

В первых строчках объявлены «_view» и «_model». Они необходимы, чтобы общаться с View и Model. Далее реализуется конструктор класса «Presenterlogin». Так же, как и во View происходит привязка к методу того самого события, которое происходит при нажатии кнопки «Войти». Метод, который срабатывает при попытки пользователя войти в систему вызывает метод «LoginCheck» от View, который в свою очередь принимает bool значение, получаемое с метода «CheckLogin» от Model. Данный метод принимает значения строк, которые ввел пользователь.

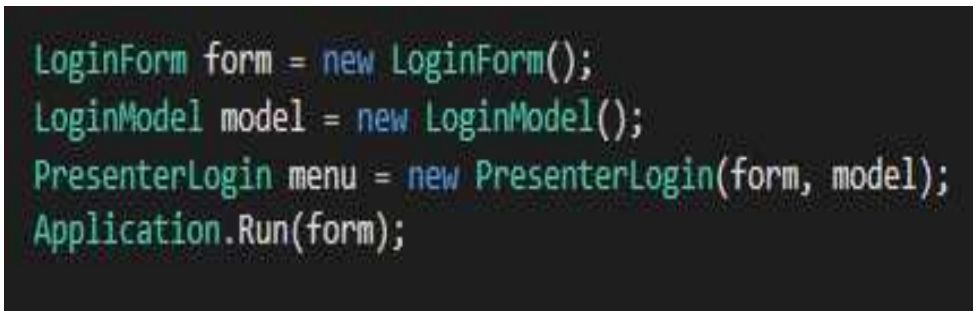

Рисунок 18 – Вход в программу

На рисунке 18 показан основной вход в программу. Создаются экземпляры классов «LoginForm» и «LoginModel» и передаются в конструктор экземпляра класса «PresenterLogin». Это необходимо для привязки Presenter с View и Model. Далее запускается форма.

4 Инструкция работы с приложением

На данном этапе разработки приложения существует две должности:

- кладовщик;
- администратор.

Кладовщик обязан регистрировать всевозможные перемещения товара на складе, а также лиц, которые относятся к этому товару.

Администратор, помимо обязанностей кладовщика, имеет возможность изменять некоторые данные в системе.

4.1 Инструкция для кладовщика

4.1.1 Вход в систему

Для работы с системой кладовщику необходимо сначала авторизоваться. При открытии программы появляется окно входа. Логином является фамилия кладовщика, пароль выдает администратор. Окно входа в программу изображено на рисунке 19.

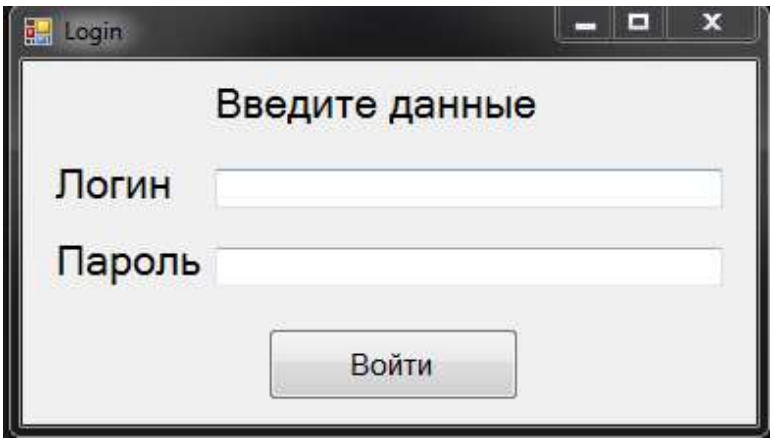

Рисунок 19 – Окно входа в программу

После правильного ввода логина и пароля открывается главное меню программы. Главное меню представлено на рисунке 20. В центре окна находится список всех товаров, расположенных на складе.

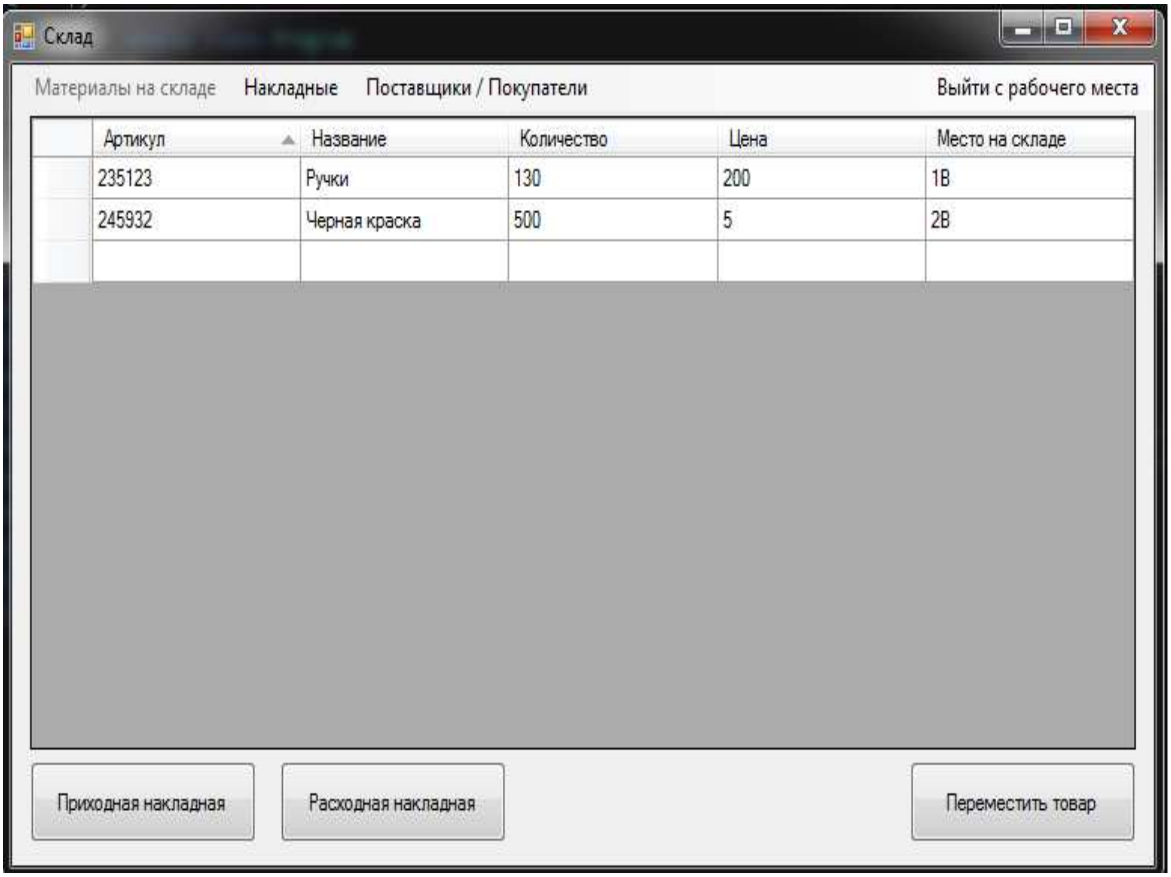

Рисунок 20 – Окно основного меню программы

Если необходимо переместить товар на новое место на складе, нужно выделить необходимый товар в списке и нажать кнопку «Переместить товар». Появится окно с возможностью выбора места хранения. После данного изменения необходимо дать указание переместить товар грузчикам склада с указанием нового места хранения.

При завершении рабочей смены или необходимости выйти с рабочего места необходимо выйти из системы для предотвращения доступа к данным. При нажатии кнопки «Выйти с рабочего места» главное меню программы закрывается и появляется окно входа в систему.

4.1.2 Просмотр накладных

При необходимости просмотра накладных необходимо нажать на кнопку «Накладные». Меню просмотра накладных изображено на рисунке 21. При нажатии кнопки «Импорт в PDF» создается PDF файл с выбранным отчетом.

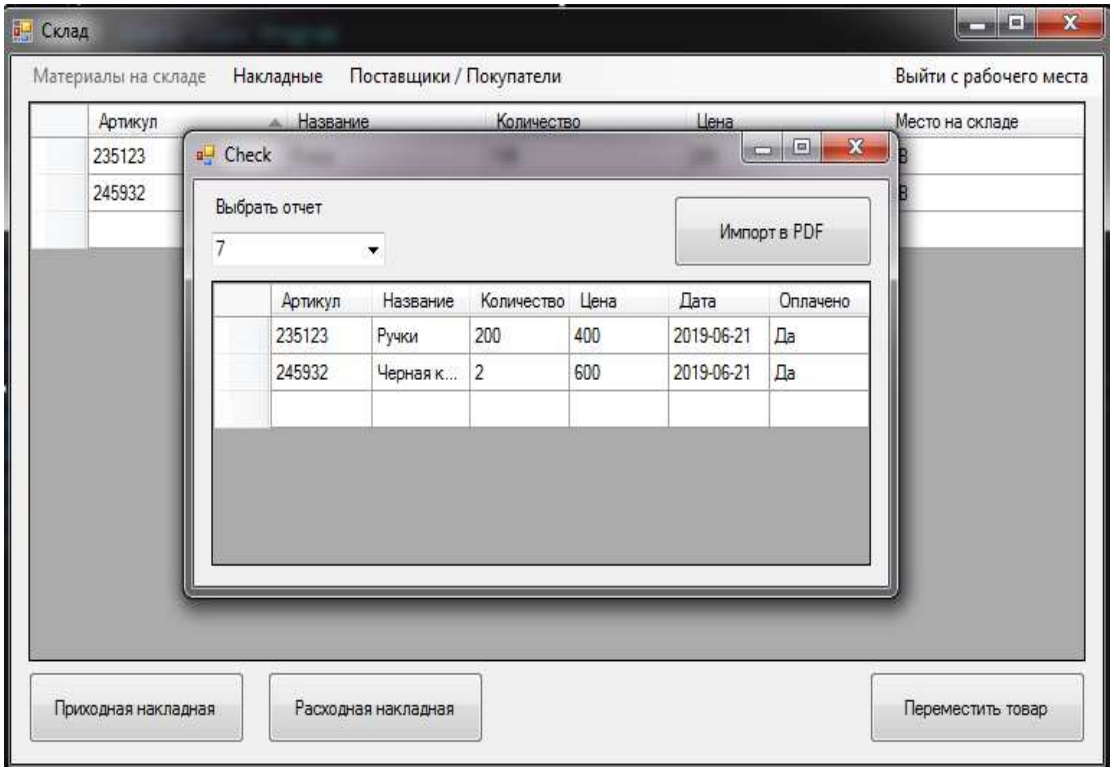

Рисунок 21 – Меню просмотра накладных

4.1.3 Поставщики и покупатели

При нажатии кнопки «Поставщики / Покупатели» открывается окно с возможностью просмотра и добавления поставщиков и покупателей (в зависимости от выбора в выпадающем списке. Для того, чтобы добавить данные сущности, необходимо заполнить поля:

- название компании;
- имя представителя;
- фамилия представителя;
- телефон;
- дополнительная информация (необязательное поле).

После ввода информации нужно нажать на кнопку «Добавить»

При необходимости удаления сущности необходимо выделить и нажать кнопку «Удалить». Данная кнопка не удаляет навсегда поставщика или покупателя из системы, все данные о них сохраняются, но в программе они перестают отображаться. Меню с поставщиками и покупателями изображено на рисунке 22.

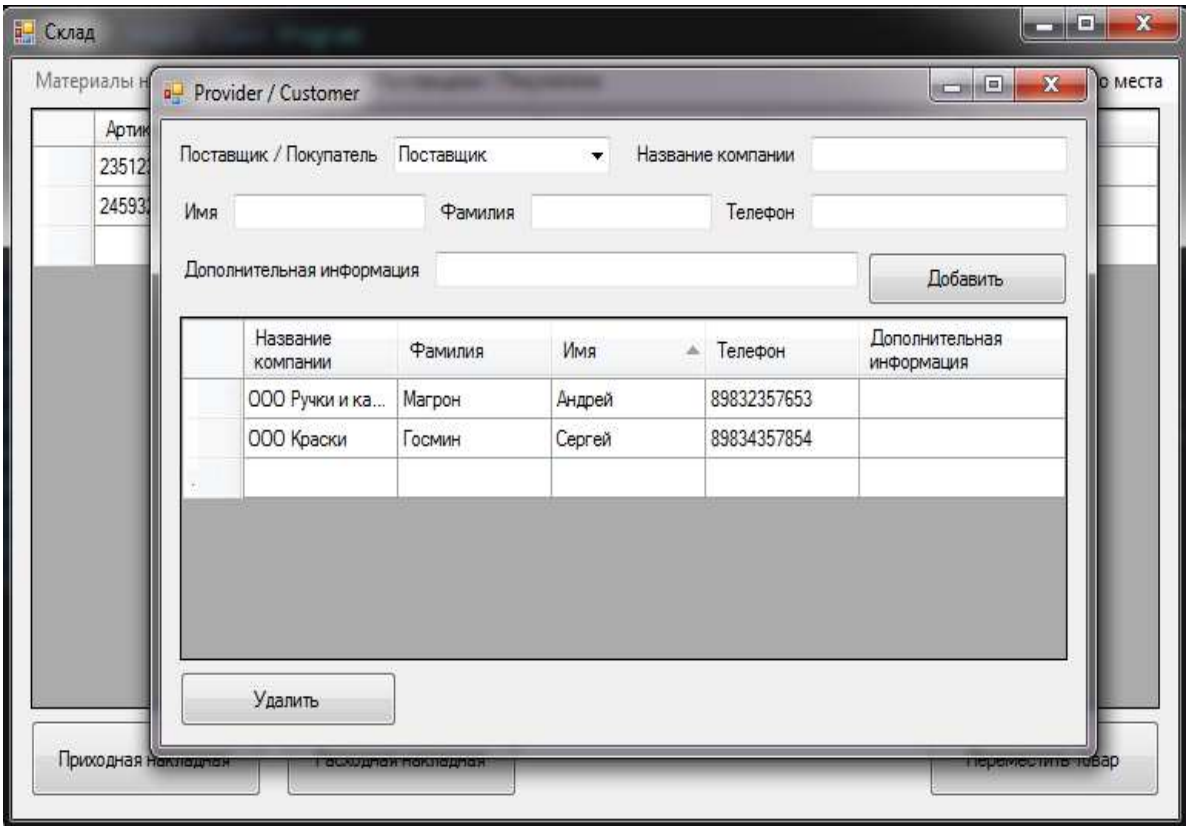

Рисунок 22 – Меню просмотра поставщиков и покупателей

4.1.4 Приход товара

При приходе товара на склад необходимо нажать на кнопку «Приходная накладная». Откроется меню регистрация товара. Нужно выбрать поставщика, который предоставляет данный товар. Если поставщика нет в списке, необходимо выйти из данного окна, перейти по кнопке «Поставщики / Покупатели» и добавить поставщика, а после вернутся в меню регистрации товара. Все необходимые данные заполняются в таблице. После того, как все

товары добавлены в таблицу, необходимо нажать на кнопку «Подтвердить приход». Окно регистрации товара закроется, товары добавятся в список, который находится на основном окне.

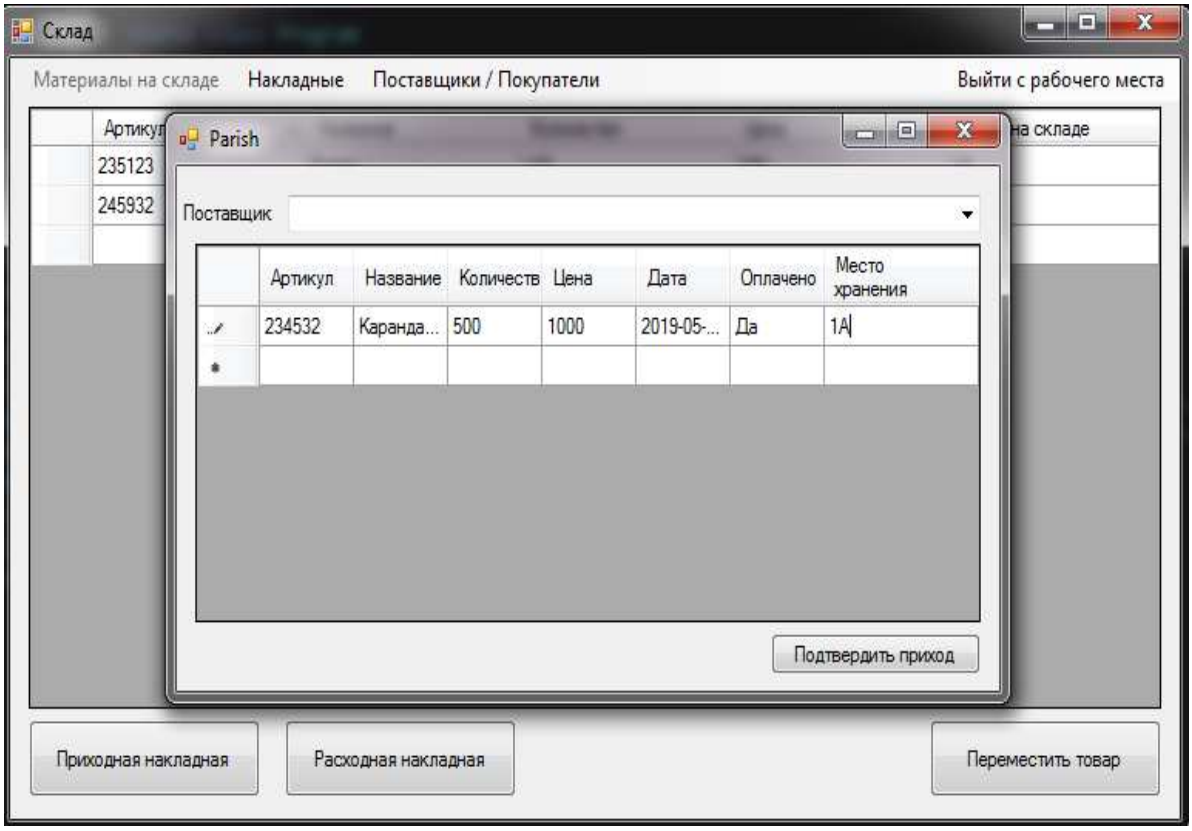

Рисунок 23 – Меню регистрации товара

4.1.5 Расход товара

При списании или покупки товара необходимо нажать на кнопку «Расходная накладная». Появится меню расхода товара. В выпадающем списке необходимо выбрать причину расхода товара. При выборе причины «Покупатель» появляется возможность выбора покупателя, а также цена покупки. Другой причиной может быть списание товара. Из первой таблицы нужно выбрать товар, ниже в поля вписать количество продажи или списания и цену, если товар продается, после нажать на кнопку «Добавить». Данный товар добавится в нижнюю таблицу, а из верхней таблицы отнимется данное количество товара. Если указанное количество товара будет больше, чем имеется

на складе, система не даст добавить товар в список расхода. Если произошла ошибка добавления товара, например, указано неправильное число, то необходимо выделить неправильной товар из второй таблицы и нажать на кнопку «Удалить товар» и добавить товар заново. После завершения добавления всей информации о расходе товаров необходимо нажать на кнопку «Подтвердить расход». Окно расхода товара представлено на рисунке 24.

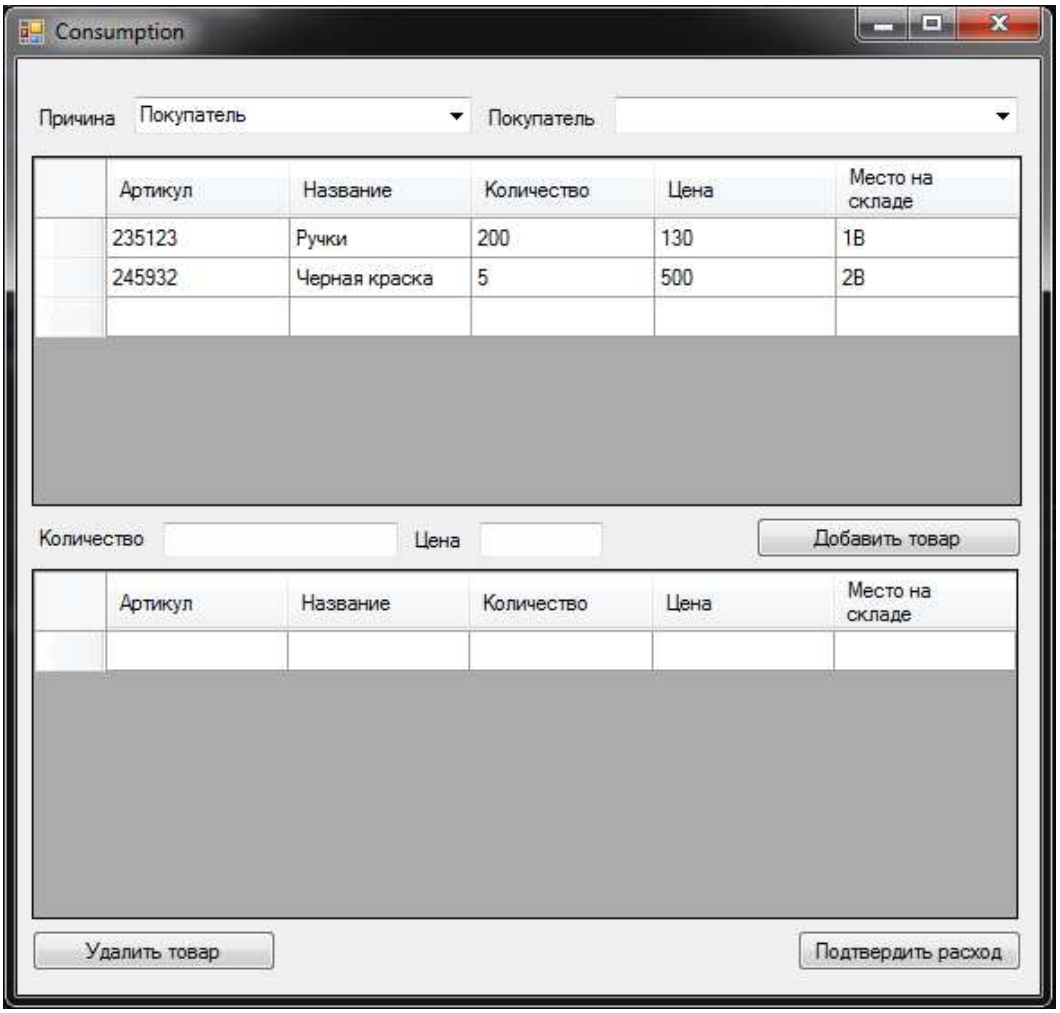

Рисунок 24 – Меню расхода товара

4.2 Инструкция для администратора

Администратор, помимо основных функций, которые доступны кладовщику, имеет ряд дополнительный возможностей:

- изменять информацию или удалять накладную в случае ошибки;

- восстанавливать удаленных поставщиков или покупателей;

- добавлять или удалять работников склада из системы;

- изменять информацию о работников склада в системе.

При необходимости администратор может выполнять обязанности кладовщика.

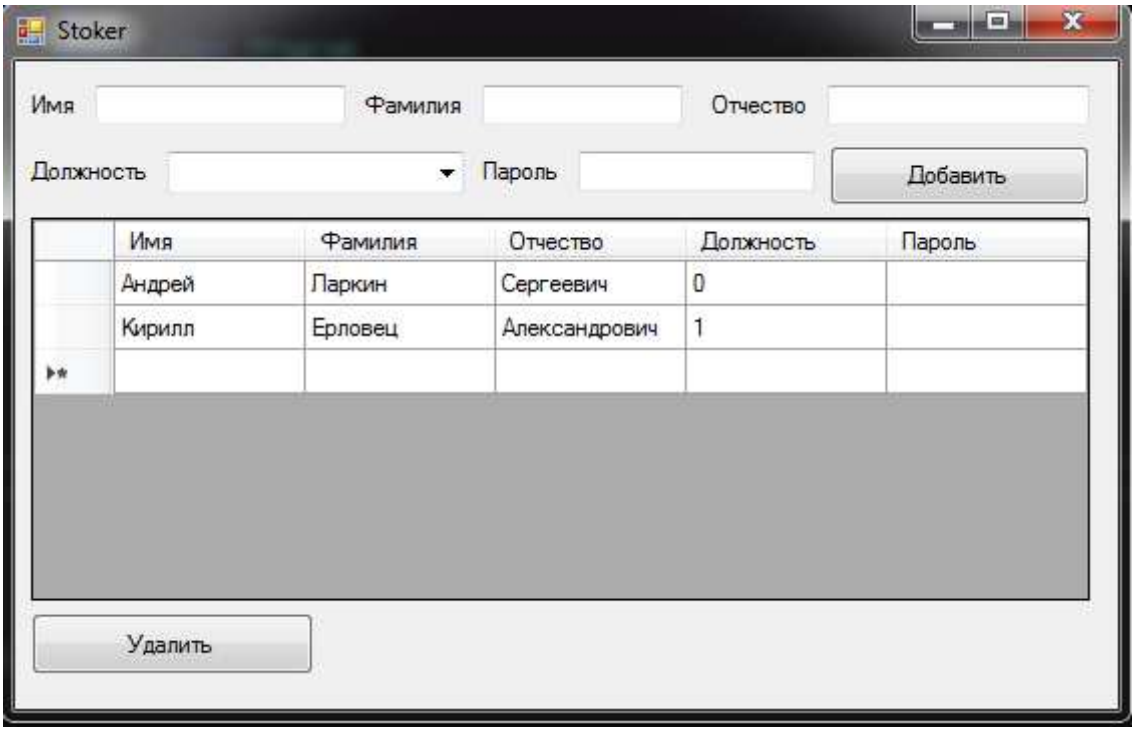

Рисунок 25 – Меню редактирования работников склада

Для того, чтобы добавить работника необходимо заполнить поля данного окна:

- имя;

- фамилия;

- отчество;

- должность;

- пароль.

После заполнения всех полей необходимо нажать кнопку «Добавить».

Чтобы удалить работника склада необходимо выделить необходимого работника в списке и нажать кнопку «Удалить». После этих действий информация об пользователе все равно сохранится в системе, но посмотреть информацию о нем или восстановить его будет уже невозможно. Для того, чтобы поменять пароль пользователю необходимо в таблице с пользователем вписать новый пароль.

ЗАКЛЮЧЕНИЕ

В результате выполнения бакалаврской работы была осуществлена автоматизация процесса учета материалов на складе. Требования, поставленные перед началом работы, были выполнены.

Для достижения поставленной цели были решены следующие задачи:

- изучена предметная область решаемой задачи;
- выявлены требования к функционалу приложения;
- выбраны технологии для реализации;
- разработана база данных;
- описана работа разработанной системы.

Были выявлены функциональные требования к ИС:

- авторизация в приложении;
- быстрый и удобный интерфейс для работы пользователя;
- организованное хранение материалов;
- всевозможные операции с материалами, которые находятся на складе;
- возможность автоматического составления отчета.

Информационная система была создана с помощью паттерна MVP, с использованием объектно-ориентированного языка программирования C#, интерфейса программирования приложений Windows Forms. Для хранение информации была использована система управления базами данных MS SQL Server. Результатом бакалаврской работы стала разработанная информационная система управления материалами на складе.

СПИСОК СОКРАЩЕНИЙ

ИС – информационная система

БД – база данных

СУБД – система управления базой данных

ПК – персональный компьютер

API – application programming interface

MVP – model, view, presenter

MS – microsoft

SQL - structured query language

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Рихтер Д. CLR via C#. Программирование на платформе Microsoft .NET Framework 4.5 на языке C#. 4-е изд. / Д.Рихтер – Санкт-Петербург: Питер, 2018. – 896 с.

2. Албахари Д., Албахари Б. C# 7.0. Карманный справочник – Санкт-Петербург: Диалектика, 2017. – 224 с.

3. Грофф Дж. Р., Вайнберг П.Н., Оппелъ Э. Дж. SQL. Полное руководство – Вильямс, 2015 – 187 с.

4. Рогачев С. Обобщенный Model-View-Controller. – Филиал ООО Лукойл-Информ, 2016 – 37 с.

5. Фримен А. ASP.NET Core MVP 2 с примерами на C# для профессионалов 7-е изд. / А.Фримен – Киев: Диалектика, 2019. – 1008 с.: ил.

6. Гвоздева Т.В. Проектирование информационных систем. – Ростов-на Дону: Феникс, 2010. – 512 с.

7. Model-View-Controller URL [Электронный ресурс]. – Режим доступа – https://ru.wikipedia.org/wiki/Model-View-Controller

8. Дэвидсон, Луис Проектирование баз данных на SQL Server 2000 / Луис Дэвидсон. - Москва: Бином. Лаборатория знаний, 2003. - 662 с.

9. Водяхо А.И. Архитектура информационных систем. – Москва: Академия, 2012. – 288 с.

10. Когаловский М.Р. Базы дынных. Проектирование, реализация и сопровождение. – Санкт-Петербург: Вильямс, 2009. – 365 с.

11. Емельянова Н.З. Проектирование информационных систем. – Москва: Форум, 2010. – 432 с.

12. Макарьева В.И. Учет материально-производственных запасов/ В.И. Макарьева. – Москва, 2002. – 235 с.

13. Белобежецкий И.А. Бухгалтерский учет материалов и МБП. 1995г. №8 с.21-25.

14. Григорьев Ю.А. Учет материально-производственных запасов // Консультант - 1996. - с.32-58.

15. Microsoft Visual Studio [Электронный ресурс]. – Режим доступа – https://msdn.microsoft.com/ru-ru/visualstudio

16. Проектирование ИС [Электронный ресурс]. – Режим доступа – https://sites.google.com/site/metodsybd/blok-5-etapy-ziznennogo-cikla/5-2-proektirovanie-is

17. Microsoft Visual Studio. [Электронный ресурс]. – Режим доступа – https://ru.wikipedia.org/wiki/Microsoft_Visual_Studio

18. Иван Бодягин Model-View-Presenter и сопутствующие паттерны RSDN Magazine – 2006. – с.16-54.

19. Особенности реализации MVP для Windows Forms. [Электронный ресурс]. – Режим доступа – https://habr.com/ru/post/211899/

20. Паттерны для новичков: MVC vs MVP vs MVVM. [Электронный ресурс]. – Режим доступа – https://habr.com/ru/post/215605/

21. Джозеф Албахари, Бен Албахари - C# 5.0. Справочник. Полное описание языка, 5 издание, 2008 – 239 с.

Федеральное государственное автономное
образовательное учреждение
высшего образования
«СИБИРСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»
Институт космических и информационных технологий
институт
Информационных систем
кафедра

УТВЕРЖДАЮ
Заведующий кафедрой ИС
_____ Дьячук П.П.
подпись фамилия, инициалы
« 21 » июня 2019 г.

БАКАЛАВРСКАЯ РАБОТА

09.03.02 Информационные системы и технологии

Автоматизация процесса учета материалов на складе

Руководитель _____ 21.06.19 канд. техн. наук, доцент И.А. Легалов
подпись, дата ученая степень, должность инициалы, фамилия

Студент КИ15-13Б 031509052 21.06.19 И.В. Дышлюк
номер группы номер зачетной книжки подпись, дата инициалы, фамилия

Нормоконтролер 21.06.19 Ю.В. Шмагрис
подпись, дата инициалы, фамилия

Красноярск 2019