**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ**

**РОССИЙСКОЙ ФЕДЕРАЦИИ**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

федеральное государственное автономное образовательное учреждение

высшего образования

«Санкт-Петербургский политехнический университет Петра Великого»

(ФГАОУ ВО «СПБПУ»)

**Институт среднего профессионального образования**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Отделение информационных технологий

Специальность: 09.02.07 Информационные системы и программирование

Дисциплина: МДК.11.01 Технология разработки и защиты баз данных

Группа: 32919/7

**КУРСОВОЙ ПРОЕКТ**

**на тему:**

**«Информационная система магазина архивной одежды»**

Студент(ка) Косовский Семён Андреевич

(ФИО) (подпись)

Руководитель Челищева Лилия Дмитриевна

(ФИО) (подпись)

Оценка

(оценка) (подпись) (ФИО)

Санкт - Петербург

2024

**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ**

федеральное государственное автономное образовательное учреждение высшего образования

«Санкт-Петербургский политехнический университет Петра Великого» (ФГАОУ ВО «СПБПУ»)

**Институт среднего профессионального образования**

Отделение информационных технологий

**Задание на курсовое проектирование**

**по МДК.11.01 «Технология разработки и защиты баз данных»**

Специальность: 09.02.07 Информационные системы и программирование

**студенту(ке) Косовскому Семёну Андреевичу** группа 32919/7

(фамилия, инициалы)

**Тема: Информационная система магазина архивной одежды**

**Исходные данные к проекту:**

1. Документация по Microsoft SQL [Электронный ресурс] -https://learn.microsoft.com/ru-ru/sql/?view=sql-server-ver16 (дата обращения 09.09.2024)
2. Виды информационных систем [Электронный ресурс] - <https://iis.guu.ru/blog/vidi-informacionnig-sistem/> (дата обращения 01.09.2024)
3. Виды баз данных. Большой обзор типов СУБД [Электронный ресурс] - <https://habr.com/ru/companies/amvera/articles/754702/> (дата обращения 02.09.2024)
4. Хэш-функция MD5 [Электронный ресурс] https://habr.com/ru/sandbox/26876 (дата обращения 03.09.2024)
5. Информационная система: Взгляд на понятие [Электронный ресурс] - https://cyberleninka.ru/article/n/informatsionnaya-sistema-vzglyad-na-ponyatie (дата обращения 08.09.2024)

**Перечень подлежащих разработке вопросов:**

1. Аналитический обзор программных средств, технологий, аналогов ПП для создания базы данных и интерфейсов пользователей.
2. Проектирование реляционной базы данных (инфологическая и даталогическая модели с указанием связей (с пояснениями) между таблицами).
3. Разработка механизмов управления данными в БД (добавление, удаление и обновление данных).
4. Организация выборки информации из разработанной базы данных (сформулировать не менее 5 запросов всех типов, реализуемых средствами выбранной СУБД).
5. Проработка технологии доступа к базе данных (определение круга пользователей базы данных и оценка возможности разграничения полномочий пользователей).
6. Разработка ПО для решения поставленной задачи.
7. Разработка алгоритмов программных модулей.
8. Тестирование работы приложения.
9. Оформление документации (руководство системного программиста).
10. Оформление пояснительной записки.

**Руководитель**

(подпись, дата) (инициалы, фамилия)

**Задание принял к выполнению:**

**Студент**

(подпись, дата) (инициалы, фамилия)

СОДЕРЖАНИЕ

[ВВЕДЕНИЕ 3](#_Toc169548676)

[1 Общая часть 4](#_Toc169548677)

[1.1 Описание предметной области 4](#_Toc169548678)

[1.2 Обзор рынка существующих решений 5](#_Toc169548679)

[1.3 Анализ и выбор технологий 6](#_Toc169548680)

[1.4 Требования 6](#_Toc169548681)

[2. Специальная часть 8](#_Toc169548682)

[2.1 Диаграмма вариантов использования 8](#_Toc169548683)

[2.2 Концептуальная, логическая и физическая модели 9](#_Toc169548684)

[2.3 Проект основных окон 11](#_Toc169548685)

[2.4 Исключительные ситуации 14](#_Toc169548686)

[2.5 Реализация добавления, удаления и обновления 14](#_Toc169548687)

[2.6 Разработанные запросы 16](#_Toc169548688)

[2.7 Разграничение прав доступа. Защита данных 17](#_Toc169548689)

[2.8 Разработка алгоритмов программных модулей 19](#_Toc169548690)

[2.9 Справочная система 24](#_Toc169548691)

[ЗАКЛЮЧЕНИЕ 25](#_Toc169548692)

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 26](#_Toc169548693)

[ПРИЛОЖЕНИЕ А 27](#_Toc169548694)

[ПРИЛОЖЕНИЕ Б 28](#_Toc169548695)

[ПРИЛОЖЕНИЕ В 32](#_Toc169548696)

# ВВЕДЕНИЕ

Информационные системы играют важнейшую роль в современном мире, выполняя задачи по управлению, обработке и анализу огромных объемов данных, которые ежедневно создаются. Эти системы востребованы в различных сферах — от бизнеса и здравоохранения до образования и индустрии развлечений.

С каждым годом растет число людей, увлекающихся архивной одеждой и коллекционированием редких предметов гардероба, что приводит к повышению спроса на уникальные и редкие вещи. Архивная одежда становится всё более популярной и разнообразной, что в свою очередь, требует создания эффективных информационных систем для управления продажами, запасами и данными клиентов. В качестве темы курсовой работы по информационным системам был выбран магазин архивной одежды.

# Общая часть

## Описание предметной области

Магазин архивной одежды — это специализированное торговое предприятие, предлагающее уникальный ассортимент архивной и исторической одежды, а также аксессуаров, которые могут удовлетворить разнообразные потребности ценителей стиля и истории моды. В ассортимент магазина входят элементы одежды, которые были популярны в различные исторические эпохи, включая классические костюмы, вечерние платья, аксессуары и обувь. Магазин может предложить, как редкие находки, такие как архивные пальто и платья, так и исторические аксессуары, такие как шляпы, перчатки и ювелирные изделия.

Одной из ключевых задач магазина архивной одежды является предоставление высококачественного сервиса, включая профессиональные консультации по выбору и стилю одежды, помощь в идентификации исторических периодов и модных трендов, а также возможность примерки и тестирования одежды. Персонал магазина обычно состоит из экспертов в области моды и истории, что позволяет им эффективно взаимодействовать с клиентами и помогать им делать осознанный выбор.

В современных условиях магазин архивной одежды активно использует информационные технологии для улучшения качества обслуживания и расширения клиентской базы. Онлайн-платформы и интернет-магазины позволяют клиентам изучать ассортимент, читать отзывы, сравнивать цены и совершать покупки, не покидая дом. Это особенно важно в условиях глобализации и быстрого развития интернет-технологий, которые изменяют способы ведения бизнеса и взаимодействия с клиентами.

Магазин архивной одежды также может выполнять функции культурного центра, организуя выставки, мастер-классы и лекции по истории моды. Такие мероприятия способствуют популяризации исторической моды, созданию сообщества энтузиастов и укреплению связи между продавцами и покупателями. Таким образом, магазин архивной одежды — это не только место для покупки уникальных предметов одежды, но и центр культурного обмена, источник профессиональных знаний и важный элемент модной индустрии, способствующий сохранению и популяризации исторического наследия в моде [5].

## Обзор рынка существующих решений

На рынке информационных систем для магазинов архивной одежды доступны разнообразные решения, которые помогают автоматизировать бизнес-процессы, улучшить взаимодействие с клиентами и повысить эффективность управления. Такие системы имеют особую значимость для магазинов, которые работают с уникальными и коллекционными предметами, требующими тщательного учета и особого подхода.

Lightspeed Retail представляет собой облачную POS-систему, которая идеально подходит для управления магазинами архивной одежды. Она предлагает инструменты для инвентаризации, управления продажами и клиентской базой. Интеграция с онлайн-платформами и платежными системами делает ее удобной для розничной торговли и интернет-продаж. Особенность Lightspeed — поддержка функции виртуальных витрин и возможности для управления уникальными предметами, что критично для магазинов, специализирующихся на архивной одежде.

Square for Retail — это комплексная система, которая сочетает функции управления продажами, инвентарем и аналитикой. Для магазинов архивной одежды важна возможность отслеживания редких и уникальных предметов, а также управления ими с помощью детализированных отчетов и аналитики. Square также поддерживает интеграцию с платформами электронной коммерции и предоставляет инструменты для маркетинга, что позволяет эффективно привлекать и удерживать клиентов.

Vend — облачная POS-система, ориентированная на розничную торговлю. Она включает мощные инструменты для управления инвентарем и анализа продаж, а также позволяет синхронизировать данные между несколькими торговыми точками. Эта функция может быть полезна для магазинов архивной одежды с несколькими локациями или онлайн-платформами, обеспечивая единый и актуальный учет всех товаров.

Erply предлагает многофункциональную POS и ERP-систему, которая поддерживает функции управления инвентарем, CRM, бухгалтерского учета и аналитики. Erply может быть адаптирован под специфические нужды бизнеса архивной одежды благодаря своей гибкости и возможности интеграции с различными платформами электронной коммерции и маркетинговыми инструментами.

ShopKeep — это интуитивно понятная POS-система, которая автоматизирует продажи и управление запасами. Ее функциональность включает анализ продаж, управление клиентами и маркетинговые инструменты, что делает ее подходящей для небольших и средних магазинов архивной одежды. Поддержка интеграции с платежными системами и инструментами для анализа трендов помогает эффективно управлять уникальными товарами и привлекать внимание покупателей.

В выборе подходящей системы для магазина архивной одежды важно учитывать такие факторы, как размер бизнеса, объем продаж, особенности инвентаря и требования к аналитике. Каждое из этих решений обладает уникальными функциями, которые можно адаптировать под специфические нужды и особенности магазина, обеспечивая эффективное управление уникальными и редкими предметами одежды [2].

## Анализ и выбор технологий

В современном мире каждую секунду генерируется огромное количество данных, создавая большие объемы информации, которые необходимо хранить, управлять и анализировать. На рынке представлены различные системы управления базами данных, которые компании выбирают в зависимости от своих потребностей. Практически каждый сервис, будь то социальные сети, платежные приложения или системы бронирования, работает с огромными объемами данных на своих серверах.

База данных представляет собой организированное хранилище информации, которое упрощает доступ к данным в любое время. База данных может существовать не только на компьютерах; любая сохраненная информация, такая как бумажные архивы или контакты на мобильном телефоне, также считается базой данных.

Среди наиболее известных систем управления базами данных можно выделить следующие: Oracle, разработанная Oracle Corporation в 1977 году Лоуренсом Эллисомоном; MySQL, популярная открытая база данных, разработанная в 1995 году; и Microsoft SQL Server, созданная компанией Microsoft в 1989 году с использованием языков C и C++ [3].

Для выполнения курсовой работы был выбран MS SQL Server по нескольким ключевым причинам. Эта система управления базами данных, разработанная Microsoft, используется многими крупными компаниями по всему миру. Долгий срок существования и высокая устойчивость к сбоям обеспечивают надежное хранение и доступ к данным. MS SQL Server обеспечивает отличную производительность благодаря своей клиент-серверной архитектуре и мощным механизмам обработки запросов, что позволяет эффективно управлять большими объемами данных и выполнять сложные запросы быстро.

Кроме того, MS SQL Server предоставляет расширенные возможности безопасности, включая авторизацию и шифрование данных, что важно для защиты конфиденциальной информации и предотвращения несанкционированного доступа. Система также включает множество инструментов для управления базами данных, таких как SQL Server Management Studio (SSMS), которые упрощают администрирование и разработку баз данных. Обширная документация и поддержка делают работу с MS SQL Server более удобной и эффективной. Вдобавок, MS SQL Server тесно интегрирован с языком программирования C#, и наличие множества библиотек и фреймворков, таких как Entity Framework, делает работу с базой данных из C# еще проще и эффективнее. Эта совместимость и интеграция делают MS SQL Server отличным выбором для проектов, разрабатываемых на C# .

## Требования

Информационная система должна быть эффективной и легкой в использовании. Эффективность включает высокую производительность, быструю обработку данных, надежность и безопасность. Легкость в использовании подразумевает интуитивно понятный интерфейс и простую навигацию. базы данных, а также содержать как минимум одну отчётную форму.

Также для нормального функционирования данной информационной системы необходим компьютер, клавиатура, мышь и следующие технические средства:

* процессор Pentium 3 и выше;
* объем свободной оперативной памяти ~256 Мб;
* объем необходимой памяти на жестком диске ~5 Мб;
* стандартный VGA-монитор или совместимый;
* стандартная клавиатура;
* наличие интернет-соединения;
* манипулятор «мышь».

## 2. Специальная часть

## 2.1 Диаграмма вариантов использования

Рисунок 1 показывает диаграмму вариантов использования программы для различных ролей, предусмотренных в информационной системе. При регистрации пользователю автоматически присваивается роль пользователя (Рисунок 1).

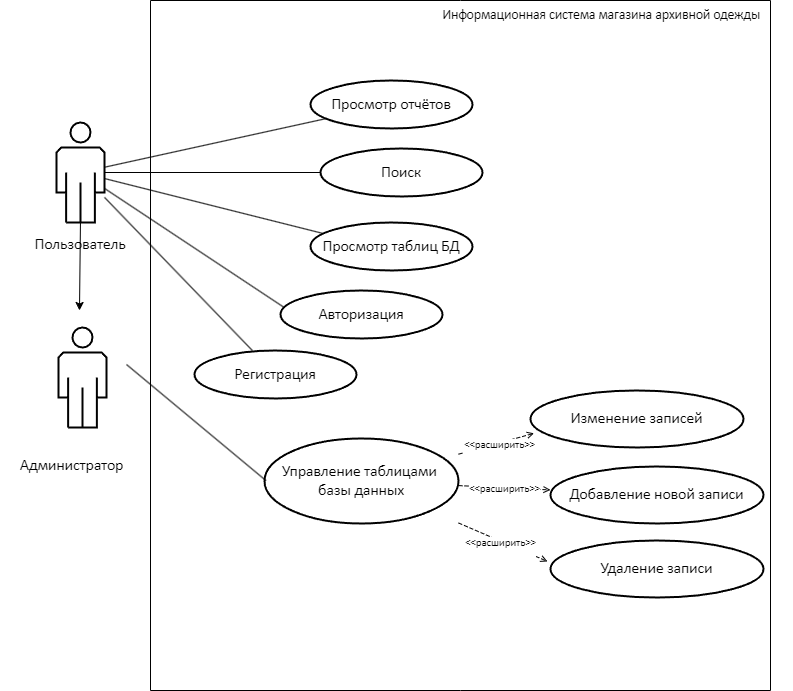


Рисунок 1 - Диаграмма претендентов

## 2.2 Концептуальная, логическая и физическая модели

Ниже изображены модели разрабатываемой БД (Рисунок 2-4).

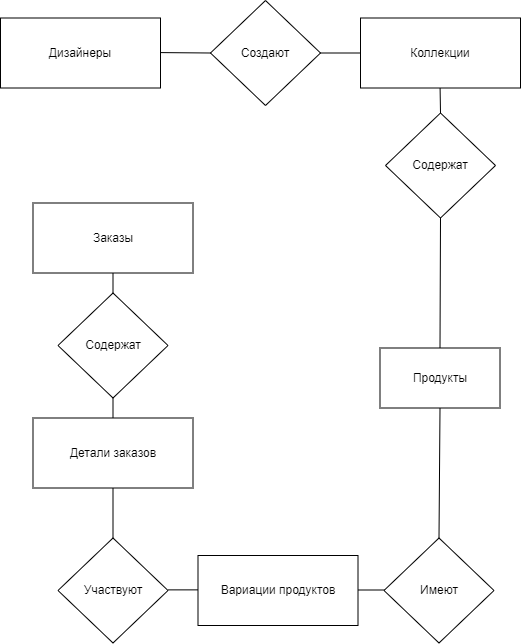


Рисунок 2 - Концептуальная модель базы данных

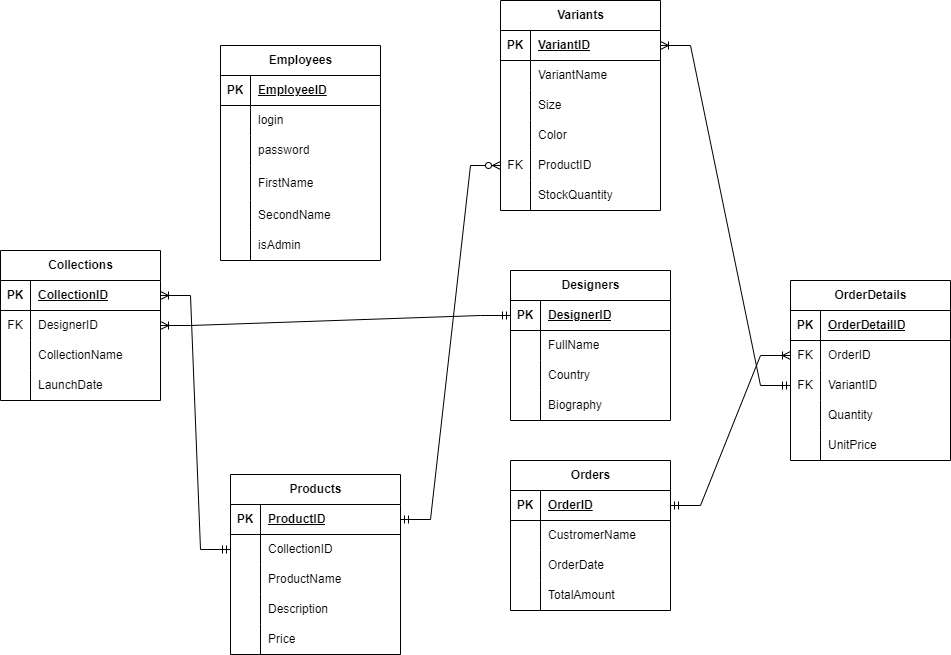


Рисунок 3 - Логическая модель базы данных

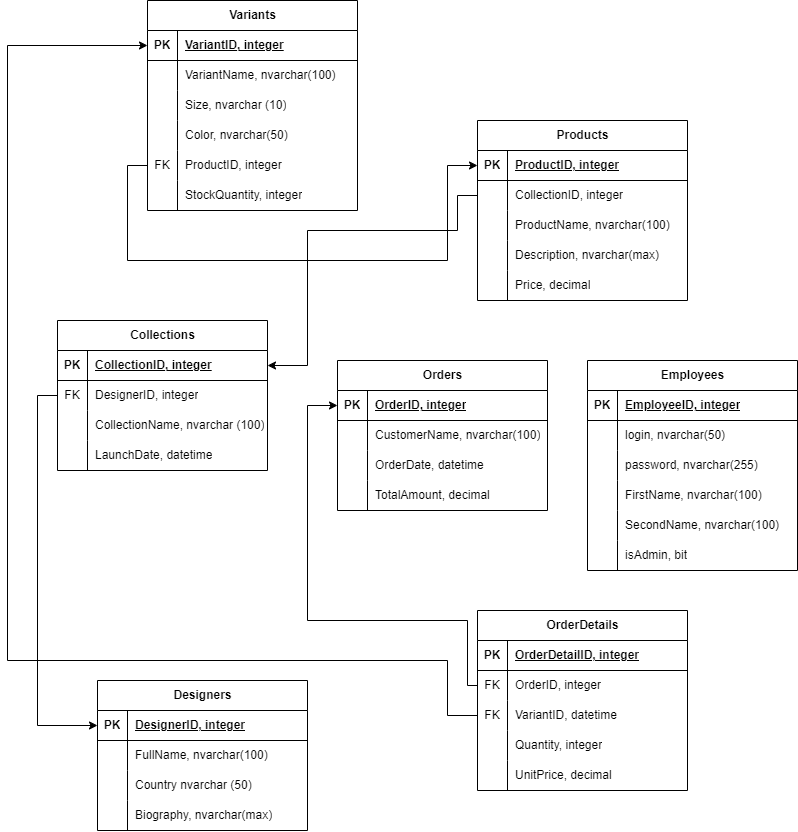


Рисунок 4 – Физическая модель базы данных

## 2.3 Проект основных окон

Приложение состоит из пяти рабочих окон: форма авторизации, форма регистрации, окно для работы с базой данных, форма отчёта и форма для добавления новой записи.

При запуске программы открывается форма авторизации, где пользователь может войти в уже существующий аккаунт или создать новый (Рисунки 5-6).

После успешной авторизации или регистрации открывается окно для работы с базой данных (Рисунок 7).



Рисунок 5 – Форма авторизации

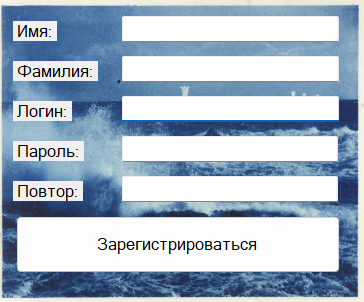


Рисунок 6 - Форма регистрации

В информационной системе администратор обладает полными правами управления базой данных, включая добавление, удаление и изменение записей. При нажатии на кнопку "Отчёт" открывается форма, позволяющая создать отчёт продаж магазина за выбранный период времени или за весь период (Рисунок 8).

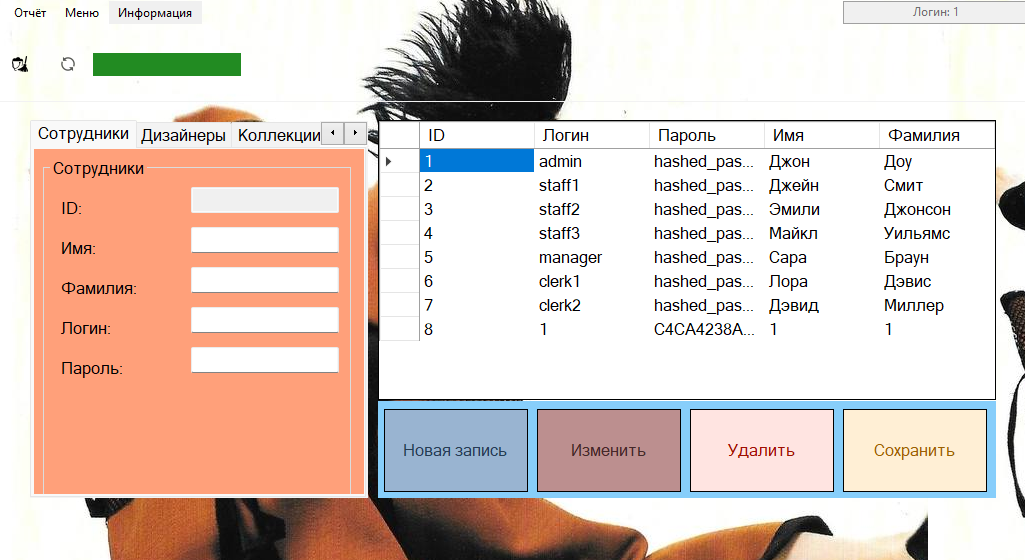


Рисунок 7 - Форма для работы с базой данных

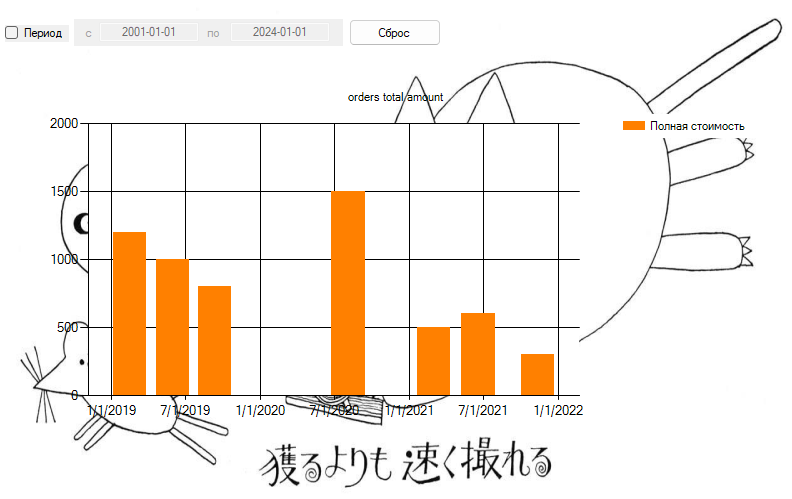


Рисунок 8 - Форма отчёта

При нажатии на кнопку «Новая запись» открывается окно для добавления записи в соответствующую таблицу базы данных (Рисунок 9).

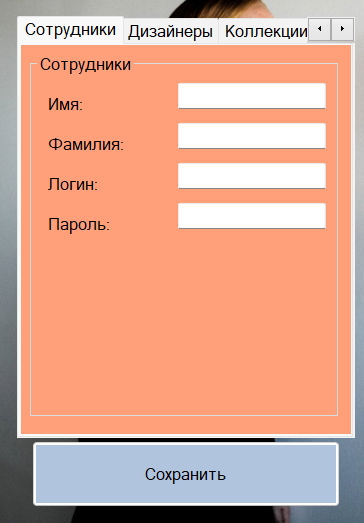


Рисунок 9 - Форма для добавления новых записей

## 2.4 Исключительные ситуации

В процессе взаимодействия с информационной системой могут возникать исключительные ситуации, когда пользователь выбирает неверные параметры или выполняет действия, противоречащие логике программы. В таблице 1 приведены описания этих исключений и способы их обработки.

Таблица 1 – Исключительные ситуации при работе с приложением

| **Ситуация** | **Способ обработки** |
| --- | --- |
| Некорректный логин пользователя при авторизации | Вывод сообщения и оповещение пользователя об ошибке авторизации |
| Некорректный (или занятый) логин пользователя при регистрации | Вывод сообщения и оповещение пользователя об ошибке авторизации |
| Некорректное заполнение полей формы пользователем при добавлении новой записи | Вывод сообщения и оповещения об ошибке администратора при добавлении записи |
| Некорректное заполнение полей формы для изменения записи | Вывод сообщения об ошибке при изменении записи администратору |
| Некорректное заполнение полей формы отчёта | Вывод сообщения об ошибке пользователю, изменение полей на корректные |

## 2.5 Реализация добавления, удаления и обновления

Приложение взаимодействует с базой данных информационной системы через добавление, удаление и обновление данных с использованием программных модулей, которые выполняют определённые SQL-запросы [1].

При добавлении новой записи пользователем открывается форма для ввода данных. После сохранения изменений информация записывается в таблицу в виде новой записи.

В процессе работы с базой данных пользователь может редактировать строки в DataGridView. При выделении строки соответствующие данные заполняются в поля формы, которые затем могут быть изменены пользователем. После внесения необходимых изменений и сохранения данных, обновлённая информация сохраняется в базе данных (Рисунок 10).



Рисунок 10 - Метод изменения записи

Также при работе с таблицами пользователь может удалять нужные записи, выделив их в компоненте DataGridView. После выделения записи, она может быть удалена из базы данных (Рисунок 11).



Рисунок 11 - Метод удаления записи

## 2.6 Разработанные запросы

Приложение реализует запросы, приведенные в таблице 2.

Таблица 2 – Список запросов

| **Запрос** | **Описание** |
| --- | --- |
| SELECT \* FROM Employees | Получение всех записей из таблицы Employees |
| SELECT \* FROM Products | Получение всех записей из таблицы Producs |
| SELECT \* FROM Orders | Получение всех записей из таблицы Order |
| SELECT \* FROM OrderDetails | Получение всех записей из таблицы OrderDetails |
| SELECT \* FROM Variants | Получение всех записей из таблицы Variants |
| SELECT \* FROM Collections | Получение всех записей из таблицы Collections |
| SELECT \* FROM Designers | Получение всех записей из таблицы Designers |
| SELECT \* FROM [Variants] WHERE CONCAT (table\_fields) LIKE @search | Получение всех похожих записей на текст пользователя из таблицы Variants. |
| INSERT INTO [Collections] (table\_fields) | Вставка новой записи в таблицу Collections |
| DELETE FROM [Employees] WHERE EmployeeID = ID | Удаление записи в таблице Employee, где EmployeeID соответствует параметру |
| UPDATE [Products] SET table\_fields WHERE ProductID = ‘{id}’ | Обновление записи в таблице Products, где ID соответствует параметру |
| SELECT OrderDate, TotalAmount FROM Orders WHERE OrderDate > ‘{startDate}’ AND OrderDate < ‘{endDate}’ | Получение сведений из таблицы Orders для составления отчёта, где startDate и endDate соответствуют параметрам |
| SELECT [password] FROM Employees WHERE [login] = @login | Получение пароля текущего пользователя из таблицы Employees |
| UPDATE Employees SET [password] = @newPassword WHERE [login] = @login | Обновление пароля текущего пользователя, где newPassword и login соответствуют параметрам |

## 2.7 Разграничение прав доступа. Защита данных

В информационной системе реализовано разграничение прав доступа. Пользователи имеют доступ к основной информации, что включает работу с записями в таблицах и просмотр отчетов. При регистрации каждому пользователю по умолчанию присваивается роль «Пользователь».

Администратор обладает полным доступом к данным приложения, что позволяет ему удалять, изменять и добавлять записи и создавать отчеты.

Для обеспечения безопасности данных используется алгоритм хеширования MD5 [4]. Хеширование происходит непосредственно в приложении: после ввода пароля пользователем он преобразуется в хэш-код, который затем сохраняется в базе данных (Рисунок 12).

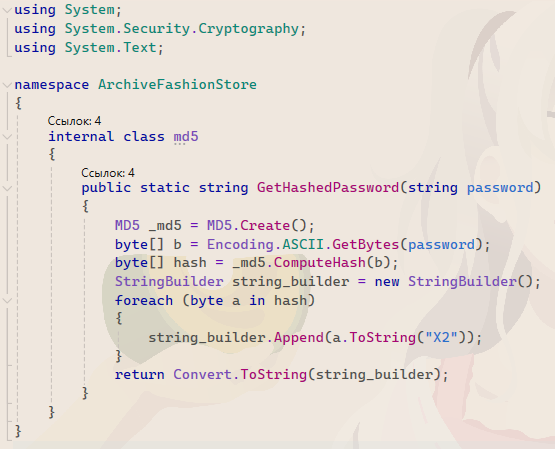


Рисунок 12 - Реализация алгоритма хеширования MD5

## 2.8 Разработка алгоритмов программных модулей

Ниже представлены разработанные блок-схемы алгоритмов программных модулей для управления базой данных и их функционирование. Блок-схема авторизации, регистрации и добавления новой записи (Рисунок 13-15).

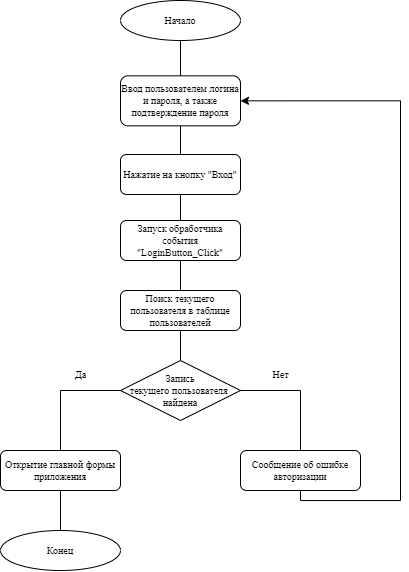


Рисунок 13 - Блок-схема алгоритма авторизации

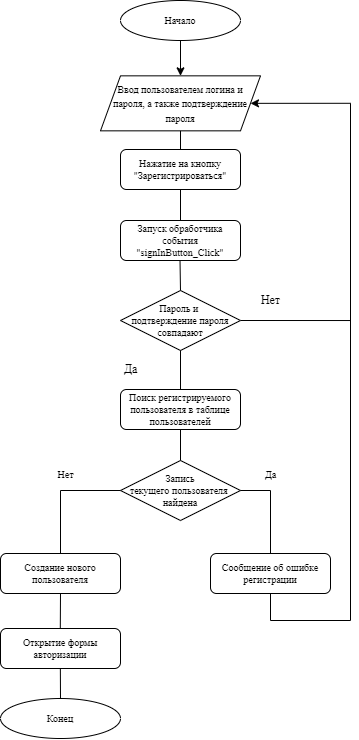


Рисунок 14 - Блок-схема алгоритма регистрации

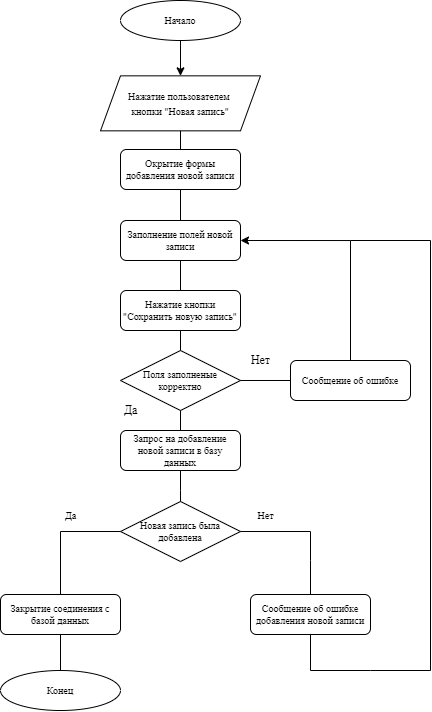


Рисунок 15 - Блок-схема алгоритма добавления новой записи

Блок-схема удаления и изменения записи (Рисунок 16-17).

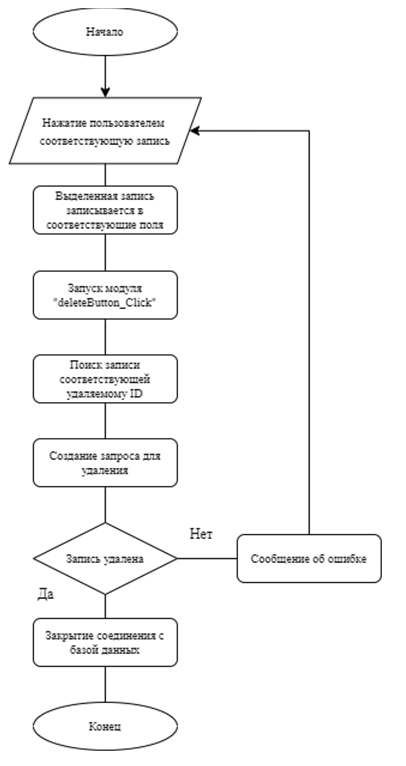


Рисунок 16 - Блок-схема алгоритма удаление записи

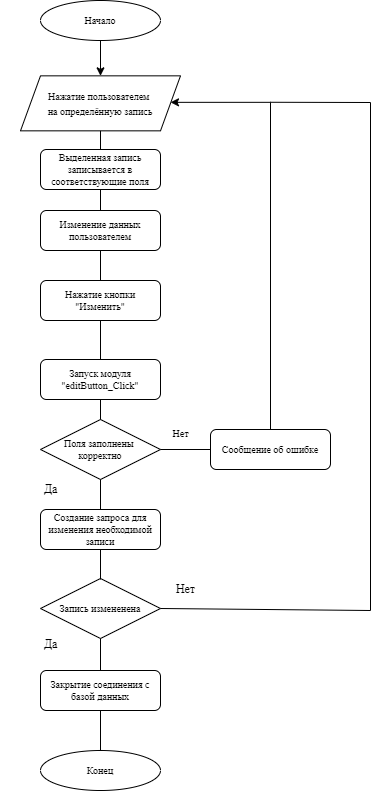


Рисунок 17 - Блок-схема алгоритма изменения записи

## 2.9 Справочная система

Для получения детальной информации об информационной системе пользователю нужно нажать кнопку «Информация» на главной форме. В результате откроется диалоговое окно (Рисунок 18).

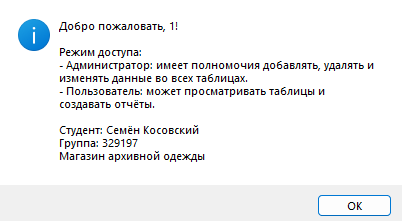


Рисунок 18 - Справочная форма

# ЗАКЛЮЧЕНИЕ

В ходе разработки была создана информационная система для управления базой данных магазина архивной одежды. Эта система позволяет эффективно добавлять, удалять и обновлять записи в базе данных, а также генерировать отчёты. Основной целью разработки было создание удобного и эффективного инструмента для управления данными, что было достигнуто благодаря интуитивно понятному интерфейсу, обеспечивающему простоту и удобство работы.

Все намеченные задачи были выполнены: была спроектирована структура базы данных, внедрены функции для работы с записями, а также разработан интерфейс для взаимодействия с пользователями и отчётами. В результате была создана функциональная и удобная система, упрощающая управление базой данных магазина архивной одежды.

В перспективе система может быть дополнена новыми функциональными возможностями, такими как новые типы отчётов, интеграция с внешними системами и сервисами для улучшения обмена данными. Также можно рассмотреть возможность оптимизации производительности системы и повышения её безопасности для защиты пользовательских данных.

# СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Документация по Microsoft SQL [Электронный ресурс] -https://learn.microsoft.com/ru-ru/sql/?view=sql-server-ver16 (дата обращения 09.09.2024)
2. Виды информационных систем [Электронный ресурс] - <https://iis.guu.ru/blog/vidi-informacionnig-sistem/> (дата обращения 01.09.2024)
3. Виды баз данных. Большой обзор типов СУБД [Электронный ресурс] - <https://habr.com/ru/companies/amvera/articles/754702/> (дата обращения 02.09.2024)
4. Хэш-функция MD5 [Электронный ресурс] https://habr.com/ru/sandbox/26876 (дата обращения 03.09.2024)
5. Информационная система: Взгляд на понятие [Электронный ресурс] - https://cyberleninka.ru/article/n/informatsionnaya-sistema-vzglyad-na-ponyatie (дата обращения 08.09.2024)

# ПРИЛОЖЕНИЕ А

(обязательное)

**Характеристики программного и аппаратного обеспечения**

Аппаратное обеспечение:

* Операционная система – Windows;
* Процессор – Минимальные требования: Процессор с частотой 1 ГГц или выше. Рекомендуемые требования: Процессор с частотой 2 ГГц или выше для обеспечения плавной работы приложения и базы данных;
* Оперативная память – Минимальные требования: 2 ГБ RAM. Рекомендуемые требования: 4 ГБ RAM или больше для обеспечения стабильной работы приложения и базы данных, особенно при работе с большими объемами данных;
* Объем необходимой памяти на жестком диске от 100Мб.

Программное обеспечение:

* Microsoft SQL Server: Использовалась как серверная база данных для хранения, обработки и управления данными. Microsoft SQL Server поддерживает различные модели данных, включая реляционные, и предлагает широкие возможности для управления данными, безопасности, репликации и т.д.

# ПРИЛОЖЕНИЕ Б

(обязательное)

**Листинг SQL-скрипта**

-- Сброс таблиц, если они уже существуют

DROP TABLE IF EXISTS OrderDetails;

DROP TABLE IF EXISTS Orders;

DROP TABLE IF EXISTS Variants;

DROP TABLE IF EXISTS Products;

DROP TABLE IF EXISTS Collections;

DROP TABLE IF EXISTS Designers;

DROP TABLE IF EXISTS Employees;

-- Создание таблицы Employees (Сотрудники)

CREATE TABLE Employees

(

EmployeeID INT PRIMARY KEY IDENTITY(1, 1),

[login] NVARCHAR(50) NOT NULL

UNIQUE,

[password] NVARCHAR(255) NOT NULL,

FirstName NVARCHAR(100) NOT NULL,

SecondName NVARCHAR(100) NOT NULL,

isAdmin BIT NOT NULL

);

-- Создание таблицы Designers (Дизайнеры)

CREATE TABLE Designers

(

DesignerID INT PRIMARY KEY IDENTITY(1, 1),

FullName NVARCHAR(100) NOT NULL,

Country NVARCHAR(50),

Biography NVARCHAR(MAX)

);

-- Создание таблицы Collections (Коллекции)

CREATE TABLE Collections

(

CollectionID INT PRIMARY KEY IDENTITY(1, 1),

DesignerID INT NOT NULL,

CollectionName NVARCHAR(100) NOT NULL,

LaunchDate DATE NOT NULL,

FOREIGN KEY (DesignerID) REFERENCES Designers (DesignerID)

);

-- Создание таблицы Products (Продукты)

CREATE TABLE Products

(

ProductID INT PRIMARY KEY IDENTITY(1, 1),

CollectionID INT NOT NULL,

ProductName NVARCHAR(100) NOT NULL,

Description NVARCHAR(MAX),

Price DECIMAL(10, 2) NOT NULL,

FOREIGN KEY (CollectionID) REFERENCES Collections (CollectionID)

);

-- Создание таблицы Variants (Варианты продуктов)

CREATE TABLE Variants

(

VariantID INT PRIMARY KEY IDENTITY(1, 1),

ProductID INT NOT NULL,

VariantName NVARCHAR(100) NOT NULL,

Size NVARCHAR(10),

Color NVARCHAR(50),

StockQuantity INT NOT NULL,

FOREIGN KEY (ProductID) REFERENCES Products (ProductID)

);

-- Создание таблицы Orders (Заказы)

CREATE TABLE Orders

(

OrderID INT PRIMARY KEY IDENTITY(1, 1),

CustomerName NVARCHAR(100) NOT NULL,

OrderDate DATE NOT NULL,

TotalAmount DECIMAL(10, 2) NOT NULL

);

-- Создание таблицы OrderDetails (Детали заказа)

CREATE TABLE OrderDetails

(

OrderDetailID INT PRIMARY KEY IDENTITY(1, 1),

OrderID INT NOT NULL,

VariantID INT NOT NULL,

Quantity INT NOT NULL,

UnitPrice DECIMAL(10, 2) NOT NULL,

FOREIGN KEY (OrderID) REFERENCES Orders (OrderID),

FOREIGN KEY (VariantID) REFERENCES Variants (VariantID)

);

-- Вставка данных в таблицу Employees (Сотрудники)

INSERT INTO Employees

(

[login],

[password],

FirstName,

SecondName,

isAdmin

)

VALUES

('admin', 'hashed\_password\_1', 'Джон', 'Доу', 1),

('staff1', 'hashed\_password\_2', 'Джейн', 'Смит', 0),

('staff2', 'hashed\_password\_3', 'Эмили', 'Джонсон', 0),

('staff3', 'hashed\_password\_4', 'Майкл', 'Уильямс', 0),

('manager', 'hashed\_password\_5', 'Сара', 'Браун', 0),

('clerk1', 'hashed\_password\_6', 'Лора', 'Дэвис', 0),

('clerk2', 'hashed\_password\_7', 'Дэвид', 'Миллер', 0);

-- Вставка данных в таблицу Designers (Дизайнеры)

INSERT INTO Designers

(

FullName,

Country,

Biography

)

VALUES

('Йоджи Ямамото', 'Япония', 'Известен авангардным дизайном и увеличенными силуэтами.'),

('Иссей Мияке', 'Япония', 'Известен плиссированными тканями и инновационными обработками тканей.'),

('Кензо Такада', 'Япония', 'Известен яркими принтами и живыми дизайнами.'),

('Рэй Кавакубо', 'Япония', 'Пионер в деконструированном моде.'),

('Томо Коидзуми', 'Япония', 'Известен своими причудливыми, объемными платьями.'),

('Хироши Фудзивара', 'Япония', 'Влиятельный дизайнер уличной моды и основатель Fragment Design.'),

('Джунья Ватанабе', 'Япония', 'Известен экспериментальными дизайнами и сотрудничеством.');

DECLARE @StartDate AS date;

DECLARE @EndDate AS date;

SELECT @StartDate = '2019-01-01',

@EndDate = '2021-12-31';

-- Вставка данных в таблицу Collections (Коллекции)

INSERT INTO Collections

(

DesignerID,

CollectionName,

LaunchDate

)

VALUES

(1,

'Коллекция Осень 2024',

DATEADD(DAY, RAND(CHECKSUM(NEWID())) \* (1 + DATEDIFF(DAY, @StartDate, @EndDate)), @StartDate)

),

(2,

'Pleats Please Осень 2024',

DATEADD(DAY, RAND(CHECKSUM(NEWID())) \* (1 + DATEDIFF(DAY, @StartDate, @EndDate)), @StartDate)

),

(3,

'Коллекция Весна 2024',

DATEADD(DAY, RAND(CHECKSUM(NEWID())) \* (1 + DATEDIFF(DAY, @StartDate, @EndDate)), @StartDate)

),

(4,

'Деконструированная А/З 2024',

DATEADD(DAY, RAND(CHECKSUM(NEWID())) \* (1 + DATEDIFF(DAY, @StartDate, @EndDate)), @StartDate)

),

(5, 'Причудливые Чудеса', DATEADD(DAY, RAND(CHECKSUM(NEWID())) \* (1 + DATEDIFF(DAY, @StartDate, @EndDate)), @StartDate)),

(6,

'Fragment Design А/З 2024',

DATEADD(DAY, RAND(CHECKSUM(NEWID())) \* (1 + DATEDIFF(DAY, @StartDate, @EndDate)), @StartDate)

),

(7,

'Экспериментальная Весна 2024',

DATEADD(DAY, RAND(CHECKSUM(NEWID())) \* (1 + DATEDIFF(DAY, @StartDate, @EndDate)), @StartDate)

);

-- Вставка данных в таблицу Products (Продукты)

INSERT INTO Products

(

CollectionID,

ProductName,

Description,

Price

)

VALUES

(1, 'Оверсайз пальто', 'Большое, теплое пальто из темной шерсти.', 1200.00),

(2, 'Плиссированная юбка', 'Стильная плиссированная юбка различных цветов.', 500.00),

(3, 'Платье с цветочным принтом', 'Платье с ярким цветочным принтом.', 800.00),

(4, 'Деконструированный жакет', 'Жакет с асимметричным дизайном.', 1000.00),

(5, 'Объемное платье', 'Платье с увеличенным объемом.', 1500.00),

(6, 'Уличная толстовка', 'Толстовка в уличном стиле.', 300.00),

(7, 'Экспериментальные брюки', 'Брюки с инновационными разрезами.', 600.00);

-- Вставка данных в таблицу Variants (Варианты продуктов)

INSERT INTO Variants

(

ProductID,

VariantName,

Size,

Color,

StockQuantity

)

VALUES

(1, 'Оверсайз пальто - черное', 'L', 'Черный', 10),

(2, 'Плиссированная юбка - красная', 'M', 'Красный', 20),

(3, 'Платье с цветочным принтом - синее', 'S', 'Синий', 15),

(4, 'Деконструированный жакет - серый', 'M', 'Серый', 5),

(5, 'Объемное платье - белое', 'L', 'Белый', 8),

(6, 'Уличная толстовка - черная', 'XL', 'Черный', 25),

(7, 'Экспериментальные брюки - серые', 'M', 'Серый', 12);

-- Вставка данных в таблицу Orders (Заказы)

INSERT INTO Orders

(

CustomerName,

OrderDate,

TotalAmount

)

VALUES

('Элис Джонсон', DATEADD(DAY, RAND(CHECKSUM(NEWID())) \* (1 + DATEDIFF(DAY, @StartDate, @EndDate)), @StartDate), 1500.00),

('Боб Смит', DATEADD(DAY, RAND(CHECKSUM(NEWID())) \* (1 + DATEDIFF(DAY, @StartDate, @EndDate)), @StartDate), 800.00),

('Кэрол Уайт', DATEADD(DAY, RAND(CHECKSUM(NEWID())) \* (1 + DATEDIFF(DAY, @StartDate, @EndDate)), @StartDate), 1200.00),

('Дэвид Браун', DATEADD(DAY, RAND(CHECKSUM(NEWID())) \* (1 + DATEDIFF(DAY, @StartDate, @EndDate)), @StartDate), 500.00),

('Ева Грин', DATEADD(DAY, RAND(CHECKSUM(NEWID())) \* (1 + DATEDIFF(DAY, @StartDate, @EndDate)), @StartDate), 300.00),

('Фрэнк Харрис', DATEADD(DAY, RAND(CHECKSUM(NEWID())) \* (1 + DATEDIFF(DAY, @StartDate, @EndDate)), @StartDate), 1000.00),

('Грейс Ли', DATEADD(DAY, RAND(CHECKSUM(NEWID())) \* (1 + DATEDIFF(DAY, @StartDate, @EndDate)), @StartDate), 600.00);

-- Вставка данных в таблицу OrderDetails (Детали заказа)

INSERT INTO OrderDetails

(

OrderID,

VariantID,

Quantity,

UnitPrice

)

VALUES

(1, 1, 1, 1200.00),

(2, 2, 1, 500.00),

(3, 4, 1, 1000.00),

(4, 6, 1, 300.00),

(5, 7, 1, 600.00),

(6, 3, 1, 800.00),

(7, 5, 1, 1500.00);

# ПРИЛОЖЕНИЕ В

(обязательное)

**Листинг приложения**

**LoginForm.cs**

using System;

using System.Data;

using System.Data.SqlClient;

using System.Drawing;

using System.Windows.Forms;

namespace ArchiveFashionStore

{

public partial class LoginForm : Form

{

Database database = new Database();

public bool ISADMIN;

public LoginForm()

{

InitializeComponent();

InitializeComponents();

Font = new Font("Roboto", 12, FontStyle.Regular);

}

private void InitializeComponents()

{

passwordTextBox.PasswordChar = '\*';

StartPosition = FormStartPosition.CenterScreen;

}

private void logInButton\_Click(object sender, EventArgs e)

{

try

{

SqlDataAdapter data\_adapter = new SqlDataAdapter();

DataTable data\_table = new DataTable();

string query\_string = $"SELECT EmployeeID, [login], [password], isAdmin FROM Employees WHERE [login] = '{loginTextBox.Text}' AND [password] = '{md5.GetHashedPassword(passwordTextBox.Text)}'";

SqlCommand sql\_command = new SqlCommand(query\_string, database.get\_connection());

data\_adapter.SelectCommand = sql\_command;

data\_adapter.Fill(data\_table);

if (data\_table.Rows.Count == 1)

{

var user = new User(data\_table.Rows[0].ItemArray[1].ToString(), Convert.ToBoolean(data\_table.Rows[0].ItemArray[3]));

MessageBox.Show("Вход выполнен!", "Вход", MessageBoxButtons.OK, MessageBoxIcon.Information);

MasterForm main\_form = new MasterForm(user);

this.Hide();

main\_form.ShowDialog();

this.Show();

}

else MessageBox.Show("Вход не выполнен!", "Вход", MessageBoxButtons.OK, MessageBoxIcon.Warning);

}

catch (Exception)

{

MessageBox.Show("Что-то пошло не так!", "Вход", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

private void signInLinkLabel\_LinkClicked(object sender, LinkLabelLinkClickedEventArgs e)

{

SignInForm signin\_form = new SignInForm();

this.Hide();

signin\_form.ShowDialog();

this.Close();

}

}

}

**ReportForm.cs**

using System;

using System.Data;

using System.Data.SqlClient;

using System.Windows.Forms;

namespace ArchiveFashionStore

{

public partial class ReportForm : Form

{

private Database database = new Database();

public ReportForm()

{

InitializeComponent();

StartPosition = FormStartPosition.CenterScreen;

}

private void ReportForm\_Load(object sender, EventArgs e)

{

FillChart();

}

private void ResetChart() { reportChart.Titles.Clear(); }

System.DateTime datetime = System.DateTime.Now;

/// <summary>

/// Заполнение диаграммы

/// </summary>

private void FillChart()

{

database.open\_connection();

DataTable data\_table = new DataTable();

string query = periodCheckBox.Checked ? $"SELECT OrderDate, TotalAmount FROM Orders WHERE OrderDate > '{startMaskedTextBox.Text}' AND OrderDate < '{endMaskedTextBox.Text}'"

: "SELECT OrderDate, TotalAmount FROM Orders";

SqlDataAdapter sql\_data\_adapter = new SqlDataAdapter(query, database.get\_connection());

try

{

sql\_data\_adapter.Fill(data\_table);

reportChart.DataSource = data\_table;

reportChart.Series["Полная стоимость"].YValueMembers = "TotalAmount";

reportChart.Series["Полная стоимость"].XValueMember = "OrderDate";

reportChart.Titles.Add("orders total amount");

}

catch (System.Exception)

{

MessageBox.Show("Что-то пошло не так, пожалуйста введите данные корректно!", "Отчёт", MessageBoxButtons.OK, MessageBoxIcon.Error);

startMaskedTextBox.Text = "2022-01-01";

endMaskedTextBox.Text = "2024-01-01";

}

}

/// <summary>

/// Полное обновление диаграммы

/// </summary>

private void SetChart()

{

ResetChart();

FillChart();

}

private void refreshButton\_Click(object sender, EventArgs e)

{

SetChart();

}

private void periodCheckBox\_CheckedChanged(object sender, EventArgs e)

{

periodPanel.Enabled = periodCheckBox.Checked;

SetChart();

}

}

}

**SigninForms.cs**

using System;

using System.Data;

using System.Data.SqlClient;

using System.Drawing;

using System.Windows.Forms;

namespace ArchiveFashionStore

{

public partial class SignInForm : Form

{

Database database = new Database();

private string access\_admin\_password = "администратор";

public SignInForm()

{

InitializeComponent();

StartPosition = FormStartPosition.CenterScreen;

Font = new Font("Roboto", 12, FontStyle.Regular);

}

private void signInButton\_Click(object sender, EventArgs e)

{

if (passwordTextBox.Text.Length == 0 || loginTextBox.Text.Length == 0 ||

firstNameTextBox.Text.Length == 0 || lastNameTextBox.Text.Length == 0)

{

MessageBox.Show("Пожалуйста заполните все поля!", "Регистрация", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);

return;

}

try

{

if (passwordTextBox.Text == repasswordTextBox.Text)

{

if (CheckUser()) return;

string query\_string = $"INSERT INTO Employees([login], [password], FirstName, SecondName, isAdmin) VALUES('{loginTextBox.Text}', '{md5.GetHashedPassword(passwordTextBox.Text)}', '{firstNameTextBox.Text}', '{lastNameTextBox.Text}', {CheckAdmin()})";

SqlCommand sql\_command = new SqlCommand(query\_string, database.get\_connection());

database.open\_connection();

try

{

if (Convert.ToBoolean(sql\_command.ExecuteNonQuery() == 1))

{

MessageBox.Show("Регистрация прошла успешно!", "Информация", MessageBoxButtons.OK, MessageBoxIcon.Information);

LoginForm login\_form = new LoginForm();

this.Hide();

login\_form.ShowDialog();

}

else MessageBox.Show("Вы не прошли регистрацию!", "Регистрация", MessageBoxButtons.OK, MessageBoxIcon.Information);

}

catch (Exception ex) { MessageBox.Show(ex.Message, "Регистрация", MessageBoxButtons.OK, MessageBoxIcon.Error); }

database.close\_connection();

}

else MessageBox.Show("Оба пароля должны совпадать!", "Регистрация", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);

}

catch (Exception)

{

MessageBox.Show("Что-то пошло не так!", "Регистрация", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

private int CheckAdmin() { return passwordTextBox.Text == access\_admin\_password ? 1 : 0; }

private bool CheckUser()

{

string login = loginTextBox.Text;

string user = passwordTextBox.Text;

SqlDataAdapter sql\_adapter = new SqlDataAdapter();

DataTable data\_table = new DataTable();

string query\_string = $"SELECT EmployeeID, [login], [password], FirstName, SecondName, isAdmin FROM Employees WHERE [login] = '{login}' and [password] = '{user}'";

SqlCommand sql\_command = new SqlCommand(query\_string, database.get\_connection());

sql\_adapter.SelectCommand = sql\_command;

sql\_adapter.Fill(data\_table);

if (data\_table.Rows.Count > 0)

{

MessageBox.Show("Пользователь уже существует!", "Регистрация", MessageBoxButtons.OK, MessageBoxIcon.Information);

return true;

}

else return false;

}

private void SigninForm\_FormClosed(object sender, FormClosedEventArgs e)

{

LoginForm login\_form = new LoginForm();

this.Hide();

login\_form.ShowDialog();

}

private void SignInForm\_Load(object sender, EventArgs e)

{

passwordTextBox.UseSystemPasswordChar = repasswordTextBox.UseSystemPasswordChar = true;

}

}

}

**AddForm.cs**

using System;

using System.Data.SqlClient;

using System.Drawing;

using System.Windows.Forms;

namespace ArchiveFashionStore

{

public partial class AddForm : Form

{

Database database = new Database();

public AddForm()

{

InitializeComponent();

InitializeFont();

InitializeColors();

StartPosition = FormStartPosition.CenterScreen;

}

private void InitializeColors()

{

ForeColor = Color.Black;

productsTabPage.BackColor = Color.LightBlue;

ordersTabPage.BackColor = Color.LightPink;

order\_details\_TabPage.BackColor = Color.LightSlateGray;

productVariantsTabPage.BackColor = Color.LightYellow;

employeesTabPage.BackColor = Color.LightSalmon;

designersTabPage.BackColor = Color.LightGoldenrodYellow;

collectionsTabPage.BackColor = Color.LightCyan;

saveButton.BackColor = Color.LightSteelBlue;

}

private void InitializeFont() { Font = new Font("Roboto", 12, System.Drawing.FontStyle.Regular); }

private void saveButton\_Click(object sender, EventArgs e)

{

database.open\_connection();

try

{

var add\_query = "";

if (fieldsTabControl.SelectedTab == collectionsTabPage) //smt

{

add\_query = $"INSERT INTO Collections ([DesignerID], [CollectionName], [LaunchDate]) " +

$"VALUES ({collections\_designer\_ID\_TextBox.Text}, '{collections\_name\_TextBox.Text}', '{collections\_launch\_date\_TextBox.Text}')";

}

else if (fieldsTabControl.SelectedTab == designersTabPage)

{

add\_query = $"INSERT INTO Designers ([FullName], [Country], [Biography]) " +

$"VALUES ('{designer\_full\_name\_TextBox.Text}', '{designer\_country\_TextBox.Text}', '{designer\_biography\_TextBox.Text}')";

}

else if (fieldsTabControl.SelectedTab == employeesTabPage)

{

add\_query = $"INSERT INTO Employees ([login], [password], FirstName, SecondName, isAdmin) " +

$"VALUES ('{employee\_login\_TextBox.Text}', '{md5.GetHashedPassword(employee\_password\_TextBox.Text)}'," +

$" '{employee\_first\_name\_TextBox.Text}', '{employee\_last\_name\_TextBox.Text}', 0)";

}

else if (fieldsTabControl.SelectedTab == ordersTabPage)

{

add\_query = $"INSERT INTO Orders (CustomerName, OrderDate, TotalAmount) " +

$"VALUES ('{order\_customer\_id\_TextBox.Text}', '{order\_order\_date\_TextBox.Text}', {order\_total\_amount\_TextBox.Text})";

}

else if (fieldsTabControl.SelectedTab == order\_details\_TabPage)

{

add\_query = $"INSERT INTO OrderDetails (OrderID, VariantID, Quantity, UnitPrice) " +

$"VALUES ({order\_detail\_order\_id\_TextBox.Text}, {order\_detail\_variant\_id\_TextBox.Text}, " +

$"{order\_detail\_quantity\_TextBox.Text}, {order\_detail\_unit\_price\_TextBox.Text})";

}

else if (fieldsTabControl.SelectedTab == productsTabPage)

{

add\_query = $"INSERT INTO Products (CollectionID, ProductName, Description, Price) " +

$"VALUES ('{product\_name\_TextBox.Text}', " +

$"'{product\_description\_TextBox.Text}', {product\_price\_TextBox.Text})";

}

else if (fieldsTabControl.SelectedTab == productVariantsTabPage)

{

add\_query = $"INSERT INTO Variants (ProductID, VariantName, [Size], [Color], [StockQuantity]) " +

$"VALUES ({product\_variants\_product\_ID\_TextBox.Text}, '{product\_variants\_name\_TextBox.Text}', " +

$"'{product\_variants\_size\_TextBox.Text}', '{product\_variants\_color\_TextBox.Text}', {product\_variants\_stock\_quantity\_TextBox.Text})";

}

var command = new SqlCommand(add\_query, database.get\_connection());

command.ExecuteNonQuery();

MessageBox.Show("Новая запись успешно добавлена!", "Новая запись", MessageBoxButtons.OK, MessageBoxIcon.Information);

}

catch (Exception)

{

MessageBox.Show("Что-то пошло не так!", "Новая запись", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);

}

}

}

}