**КОМИТЕТ ПО НАУКЕ И ВЫСШЕЙ ШКОЛЕ СПБ**

Санкт-Петербургское государственное бюджетное профессиональное образовательное учреждение

«Санкт-Петербургский технический колледж управления и коммерции»

**КУРСОВОЙ ПРОЕКТ**

|  |
| --- |
| по МДК 01.01 Разработка программных модулей |
| на тему: Разработка информационной системы для парфюмерного салона |
|  |
|  |

Специальность: 09.02.07 Информационные системы и программирование

|  |  |
| --- | --- |
| Студент: | Кузяхметов В.И. / / |
| Группа: | 9ПО-41 |
| Преподаватель: | Смирнова А.В. / / |
| Оценка: | / / |

Санкт-Петербург

2023 г.

СОДЕРЖАНИЕ

[ВВЕДЕНИЕ 3](#_Toc89726919)

[1. ПРОЕКТИРОВАНИЕ ИНФОРМАЦИОННОЙ СИСТЕМЫ 4](#_Toc89726920)

[1.1. Постановка задачи 4](#_Toc89726921)

[1.2. Проектирование функциональных возможностей системы 5](#_Toc89726922)

[1.3. Разработка концептуальной и логической модели данных 5](#_Toc89726923)

[2. РАЗРАБОТКА ИНФОРМАЦИОННОЙ СИСТЕМЫ 7](#_Toc89726924)

[2.1. Выбор средств и среды разработки 7](#_Toc89726925)

[2.2. Реализация базы данных 8](#_Toc89726926)

[2.3. Разработка пользовательского приложения 13](#_Toc89726927)

[2.4. Тестирование разработанной автоматизированной системы 31](#_Toc89726928)

[ЗАКЛЮЧЕНИЕ 35](#_Toc89726929)

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ И ЛИТЕРАТУРЫ 36](#_Toc89726930)

[ПРИЛОЖЕНИЯ 37](#_Toc89726931)

# ВВЕДЕНИЕ

В настоящее время сложно представить себе современное предприятие, которое не использует в своей работе информационные системы. Как отмечает в своей работе Б. Гейтс, использование современных информационных систем – обязательное условие получения «преимущества в конкурентной борьбе».

В данном курсовой проекте рассматривается разработка информационной подсистемы для парфюмерного салона, реализующего сопутствующие товары.

Ещё в Древнем Египте люди разработали парфюмерное ремесло, заключающееся в изготовлении изделий, применяемых для ароматизации тела человека, одежды, белья и воздуха в помещениях. Изначально понятие «парфюмерия» относилось к веществам, применяемым для благовонного окуривания воздуха.

В современном мире парфюмерия и косметика занимают особое место в жизни людей, являясь неотъемлемой частью их образа и стиля. Парфюмерный салон представляет собой уникальное место, где каждый может найти аромат, отражающий его индивидуальность и настроение.

При работе парфюмерного салона, реализующего продукции собственного производства, непременно возникает необходимость вести учет обращений клиентов, учета товаров, сырья и заказов.

Целью курсового проектирования является автоматизация хранения и обработки данных в информационной системе «Парфюмерный салон». Проект представляет собой базу данных и пользовательский интерфейс, позволяющий обеспечить доступ к ресурсам базы данных.

Для реализации цели необходимо решить ряд задач:

1. Провести анализ предметной области;
2. Спроектировать информационную систему;
3. Реализовать базу данных в выбранной СУБД;
4. Разработать приложение;
5. Провести тестирование

# 1. ПРОЕКТИРОВАНИЕ ИНФОРМАЦИОННОЙ СИСТЕМЫ

## 1.1. Постановка задачи

Салон представляет собой розничную торговлю парфюмерной продукцией.

Функциями салона являются:

* Разработка новых ароматов для парфюмерной продукции;
* Продажа продукции;
* Доставка продукции.

Для эффективной работы любой автоматизированной системы нужно постоянно поддерживать информацию, содержащуюся в ней в актуальном состоянии.

Входными документами будут являться: заказы клиентов.

Выходными данными будут накладные на продажу товаров клиентам и отчеты по продажам и товарам.

Назначение и цели создания программы: автоматизированная система парфюмерного салона должна вести учет заказов клиентов и организовывать выдачу заказов клиентам.

Автоматизированная система должна содержать следующие функции:

* Добавление, просмотр, редактирование информации о клиентах парфюмерного салона;
* Добавление, просмотр, редактирование информации о заказах на покупку парфюмерной продукции;
* Добавление, просмотр, редактирование информации об избранной продукции;
* Добавление, просмотр, редактирование информации о товарах в корзине
* Просмотр информации о совершенных заказах.

## 1.2. Проектирование функциональных возможностей системы

В соответствии с постановкой задачи были определены пользователи и функциональный состав ИС представлен в виде диаграммы:

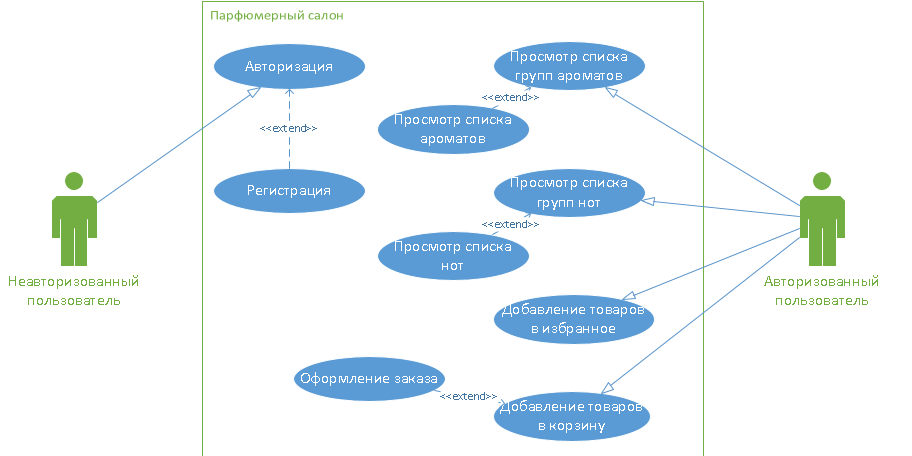


Рисунок 1 - Функциональная схема приложения

## 1.3. Разработка концептуальной и логической модели данных

Концептуальное (инфологическое) проектирование – построение сематической модели предметной области, то есть информационной модели наиболее высоко уровня абстракции. Такая модель создается без ориентации на какую-либо конкретную СУБД и модель данных. Термины «сематическая модель», «концептуальная модель» и «инфологическая модель» являются синонимами. Кроме того, в этом контекст равноправно могут использоваться слова «модель базы данных» и «модель предметной области», поскольку такая модель является как образом реальности, так и образом проектируемой базы данных для этой реальности.

Конкретный вид и содержание концептуальной модели базы данных определяется выбранным для этого формальным аппаратом. Обычно используются графические нотации, подобные ER-диаграммам.

На основе анализа предметной области была разработана концептуальная модель данных

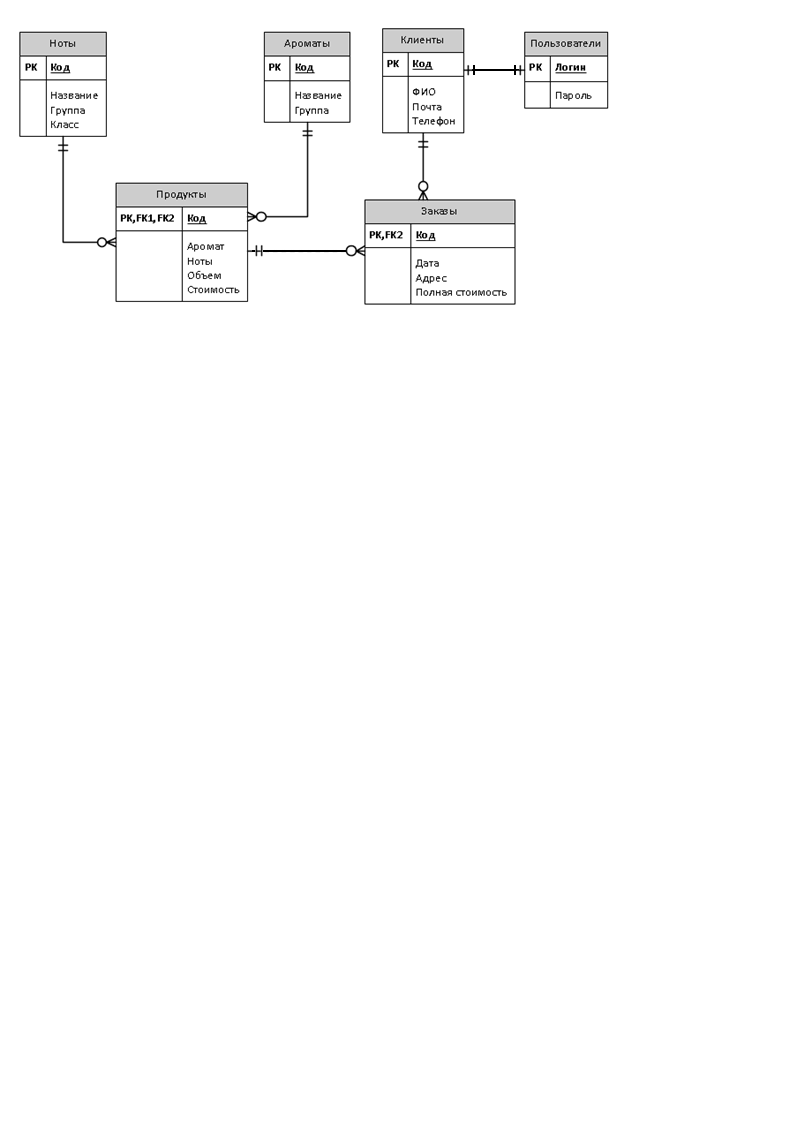


Рисунок 2 - Инфологическая схема

На основе концептуальной модели была создана логическая модель (ЕР диаграмма)

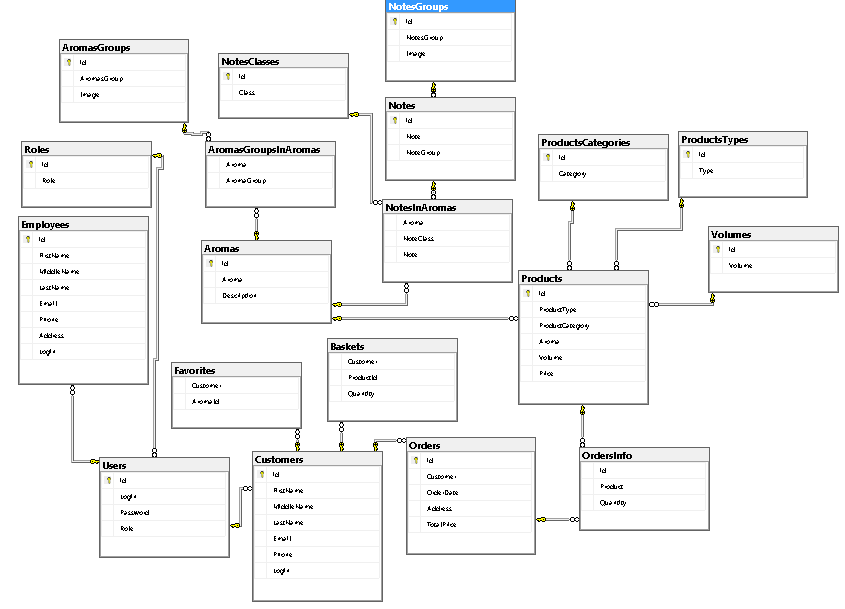


Рисунок 3 - Диаграмма базы данных

# 2. РАЗРАБОТКА ИНФОРМАЦИОННОЙ СИСТЕМЫ

## 2.1. Выбор средств и среды разработки

Для реализации базы данных будет использованы среда разработкиSQL Server Management Studio (SSMS) — это интегрированная среда для управления любой инфраструктурой SQL. SSMS предоставляет средства для настройки, наблюдения и администрирования экземпляров SQL Server и баз данных. С помощью SSMS можно развертывать, отслеживать и обновлять компоненты уровня данных, используемые вашими приложениями, а также создавать запросы и скрипты.

Инструментом разработки интерфейса пользователя приложения является интегрированная среда разработки (IDE) программного обеспечения от компании Microsoft Visual Studio 2019. Помимо стандартного редактора и отладчика, которые существуют в большинстве сред IDE, Visual Studio включает в себя компиляторы, средства автозавершения кода, графические конструкторы и многие другие функции для упрощения процесса разработки.

Разработка интерфейса приложения будет происходить с помощью технологии WPF.

Windows Presentation Foundation (WPF) — аналог WinForms, система для построения клиентских приложений Windows с визуально привлекательными возможностями взаимодействия с пользователем, графическая (презентационная) подсистема в составе .NET Framework (начиная с версии 3.0), использующая язык XAML.

Программная часть интерфейса реализована с использованием объектно-ориентированного языка программирования C#. На сегодняшний момент язык программирования C# один из самых мощных, быстро развивающихся и востребованных языков в ИТ-отрасли. В настоящий момент на нем пишутся самые различные приложения: от небольших десктопных программок до крупных веб-порталов и веб-сервисов, обслуживающих ежедневно миллионы пользователей.

## 2.2. Реализация базы данных

База данных может включать много таблиц, в каждой из которых хранятся данные по отдельным объектам. Каждая таблица может содержать много полей с данными различного типа, включая текст, числа, даты и гиперссылки.

Для информационной системы была создана база данных PerfumeSalon.

На рисунке 4 показан список всех таблиц.

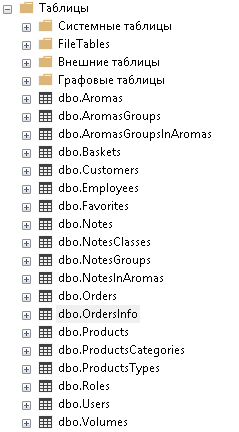


Рисунок 4 - Список всех таблиц БД

Структура таблицы «AromasGroups» представлена на рисунке 5.

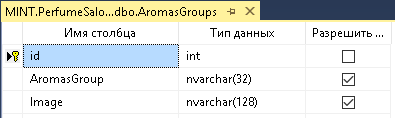


Рисунок 5 - Структура таблицы «AromasGroups»

Структура таблицы «Aromas» представлена на рисунке 6.

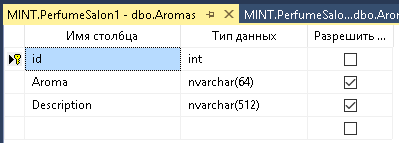


Рисунок 6 - Структура таблицы «Aromas»

Структура таблицы «AromasGroupsInAromas» представлена на рисунке 7.

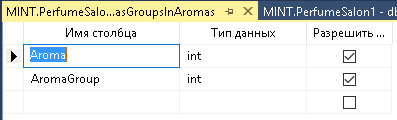


Рисунок 7 - Структура таблицы «AromasGroupsInAromas»

Структура таблицы «NotesClasses» представлена на рисунке 8.

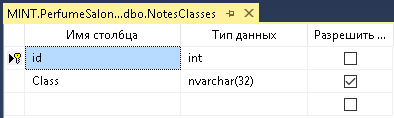


Рисунок 8 - Структура таблицы «NotesClasses»

Структура таблицы «NotesGroups» представлена на рисунке 9.

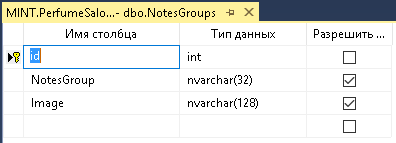


Рисунок 9 - Структура таблицы «NotesGroups»

Структура таблицы «Notes» представлена на рисунке 10.

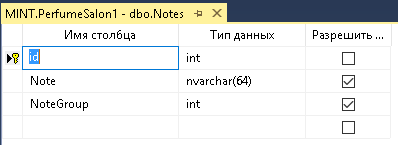


Рисунок 10 - Структура таблицы «Notes»

Структура таблицы «NotesInAromas» представлена на рисунке 11.

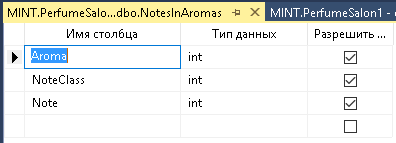


Рисунок 11 - Структура таблицы «NotesInAromas»

Структура таблицы «Products» представлена на рисунке 12.

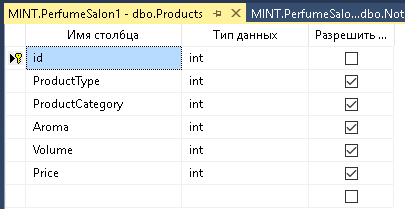


Рисунок 12 - Структура таблицы «Products»

Структура таблицы «ProductsCateries» представлена на рисунке 13.

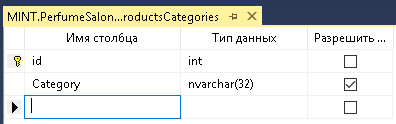


Рисунок 13 - Структура таблицы «ProductsCateries»

Структура таблицы «ProductsTypes» представлена на рисунке 14.

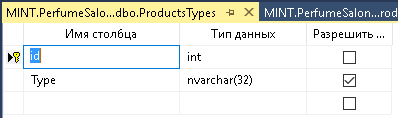


Рисунок 14 - Структура таблицы «ProductsTypes»

Структура таблицы «Volumes» представлена на рисунке 15.

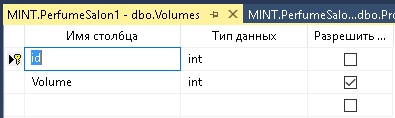


Рисунок 15 - Структура таблицы «Volumes»

Структура таблицы «Roles» представлена на рисунке 16.

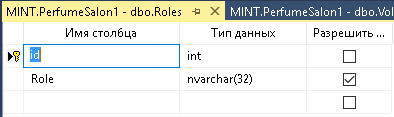


Рисунок 16 - Структура таблицы «Roles»

Структура таблицы «Users» представлена на рисунке 17.

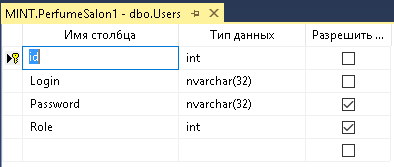


Рисунок 17 - Структура таблицы «Users»

Структура таблицы «Customers» представлена на рисунке 18.

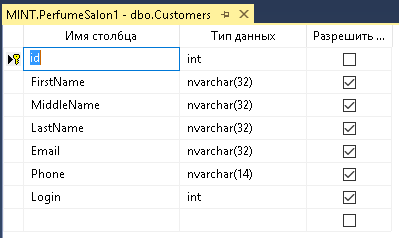


Рисунок 18 - Структура таблицы «Customers»

Структура таблицы «Employees» представлена на рисунке 19.

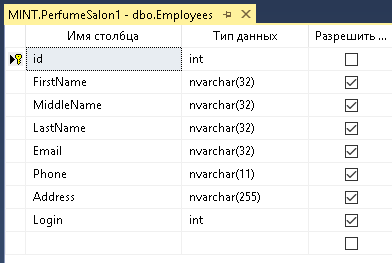


Рисунок 19 - Структура таблицы «Employees»

Структура таблицы «Baskets» представлена на рисунке 20.

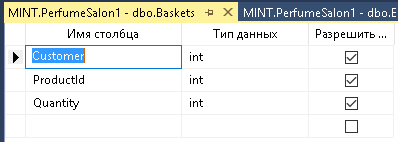


Рисунок 20 - Структура таблицы «Baskets»

Структура таблицы «Favorites» представлена на рисунке 21.

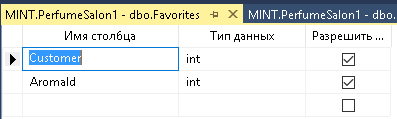


Рисунок 21 - Структура таблицы «Favorites»

Структура таблицы «Orders» представлена на рисунке 22.

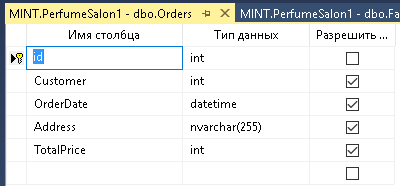


Рисунок 22 - Структура таблицы «Orders»

Структура таблицы «OrdersInfo» представлена на рисунке 23.

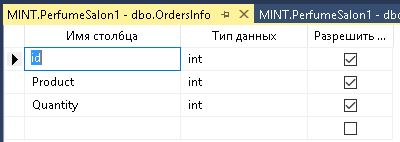


Рисунок 23 - Структура таблицы «OrdersInfo»

Полный скрипт на создание базы данных и таблиц представлен в приложении А.

## 2.3. Разработка пользовательского приложения

При запуске программы открывается домашняя страница, которая предназначена для предоставления права просмотра информации о продукции неавторизированным пользователям (Рисунок 24).

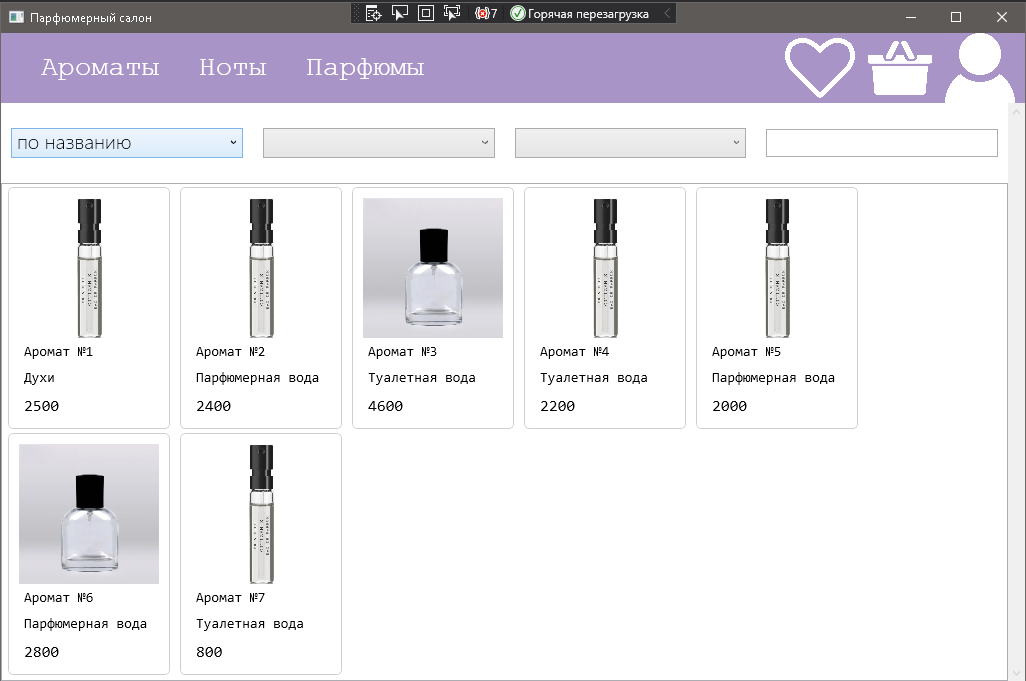


Рисунок 24 - Домашняя страница

На домашней странице отображается каталог парфюмерной продукции, который можно отфильтровать по группе ароматов, концентрации аромата, отсортировать по названию, цене, осуществить поиск по тексту в поле. Результат этих действий представлен на рисунке 25.

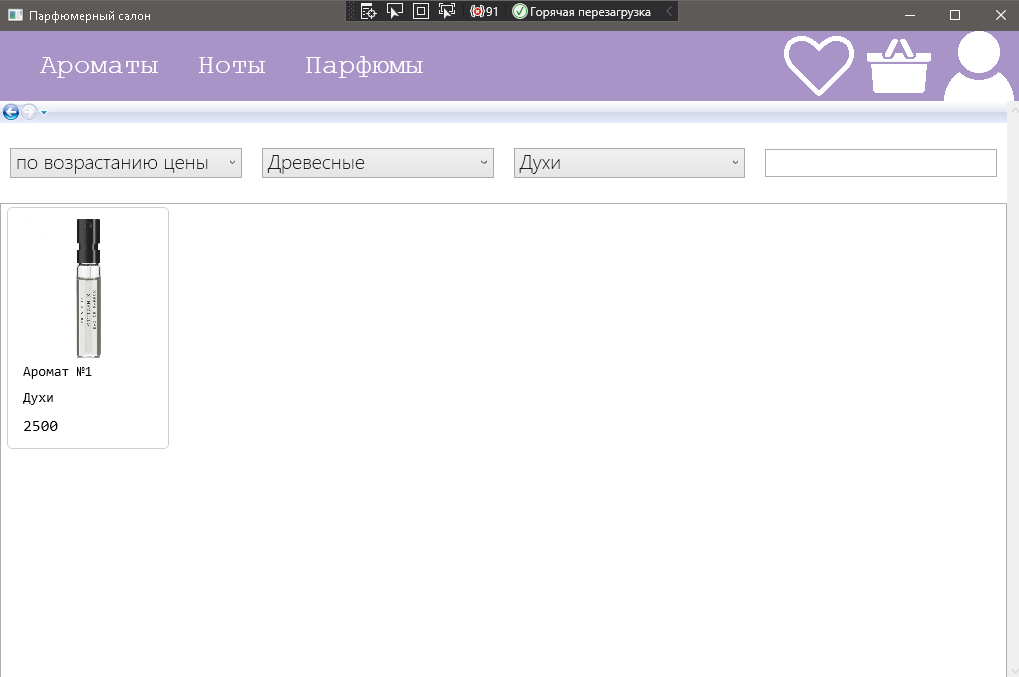


Рисунок 25 - Результат фильтрации, сортировки

После нажатия на кнопку «Ароматы», пользователя встретит страница со списком групп ароматов. Число, стоящее рядом с группой, отражает количество ароматов, которые можно отнести к этой группе. Страница представлена на рисунке 26.

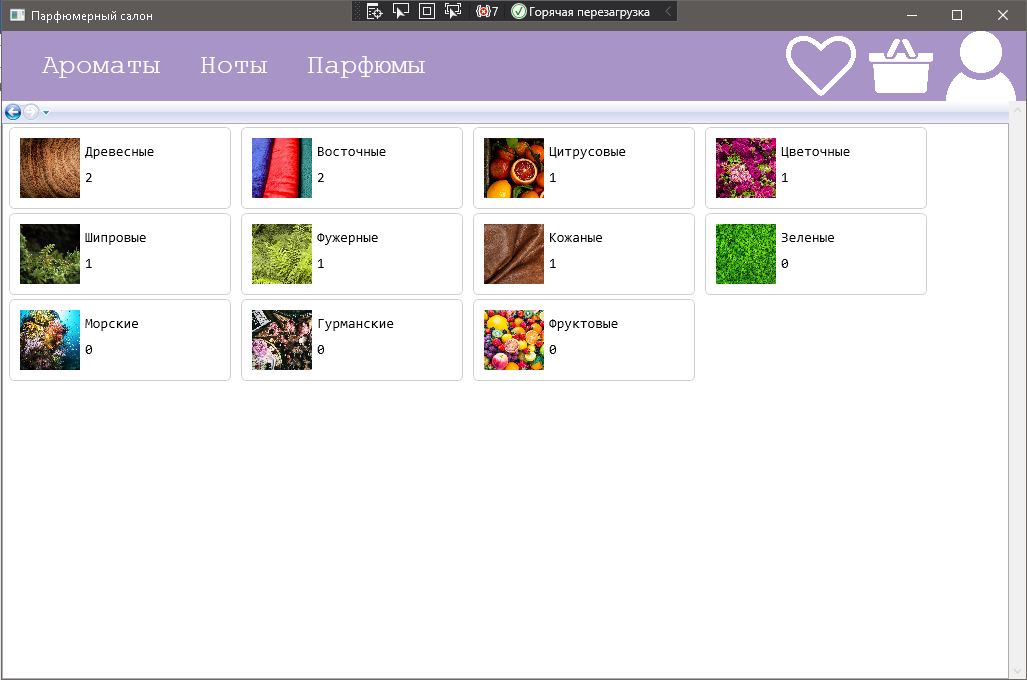


Рисунок 26 - Страница "Ароматы"

После нажатия на элемент из списка, пользователя встретит страница с парфюмерной продукции соответствующей группы. Результат нажатия на группу ароматов «Восточные» показан на рисунке 27.

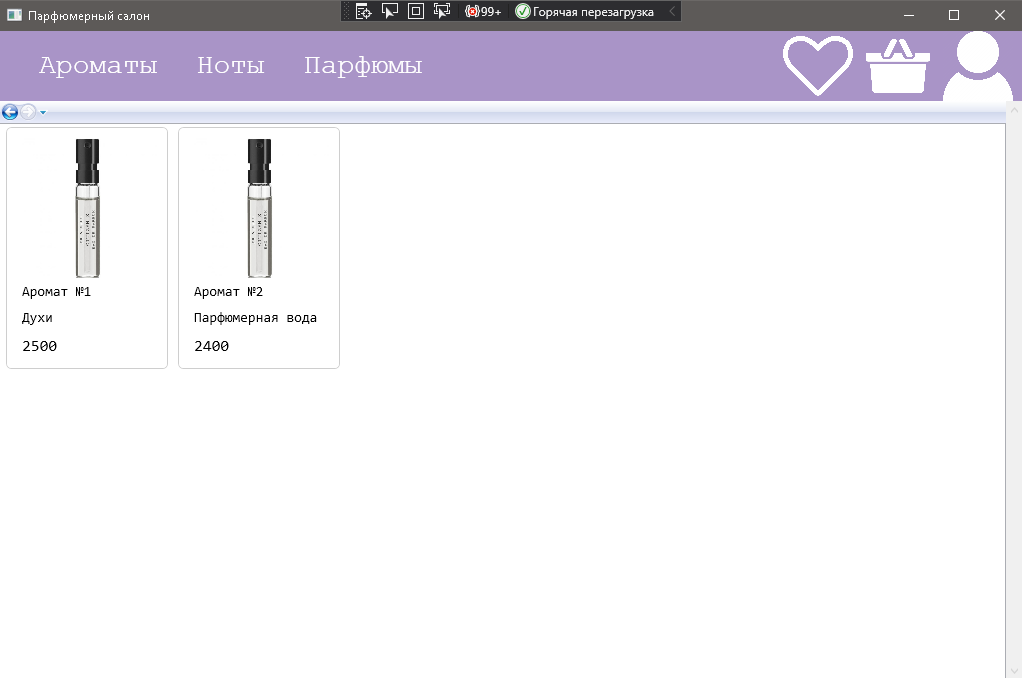


Рисунок 27 - Результат нажатия на группу "Восточные"

На этой странице отображается список, каждый элемент которого представляет из себя изображение продукта, его название и тип, а также стоимость аромата в минимальном доступном объеме.

После нажатия на элемент из списка, пользователь перейдет на страницу с подробной информацией о выбранной парфюмерной продукции. Результат нажатия на элемент «Аромат №1» из списка приведен на рисунках 28-29.



Рисунок 28 - Страница аромата

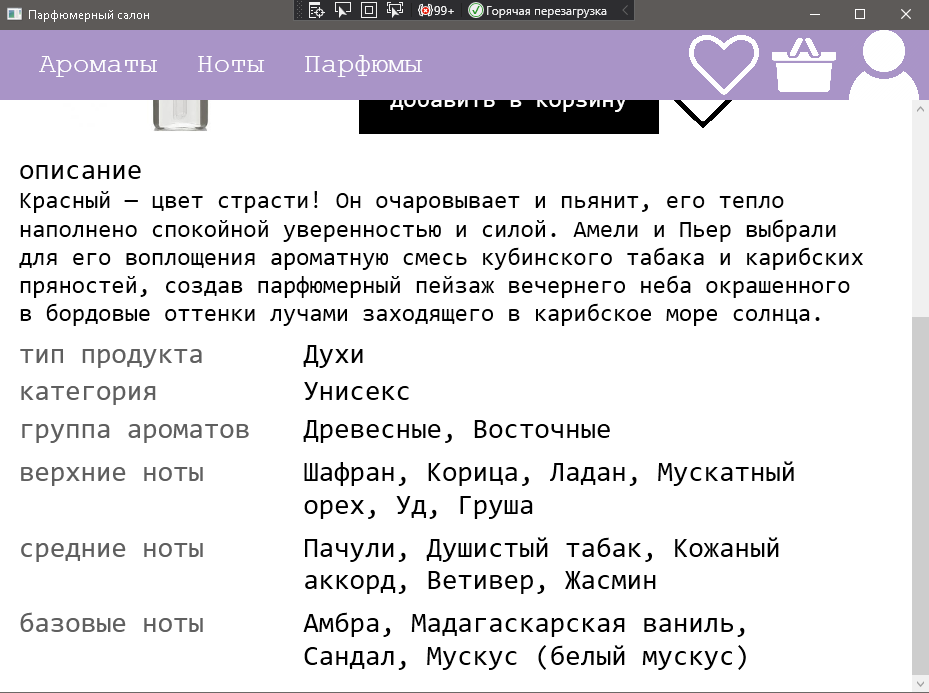


Рисунок 29 - Страница аромата

Страница аромата представляет из себя совокупную информацию о выбранном аромате, включая описание, тип, категорию, группы, ноты, с возможностью добавления в корзину продукта выбранного объема и добавления в избранное. При добавлении аромата в избранное, иконка «Избранное» («Сердечко»), перекрашивается в красный (см. Рисунок 30).



Рисунок 30 - Результат добавления аромата в избранное

При нажатии на кнопку «добавить в корзину», в корзину пользователя добавится аромат в выбранном объеме. В том случае если такой уже имеется в корзине, его количество увеличится на 1 штуку.

В том случае, если добавить продукт в корзину или избранное пытается неавторизированный пользователь, его перекидывает на страницу авторизации (см. Рисунок 31).

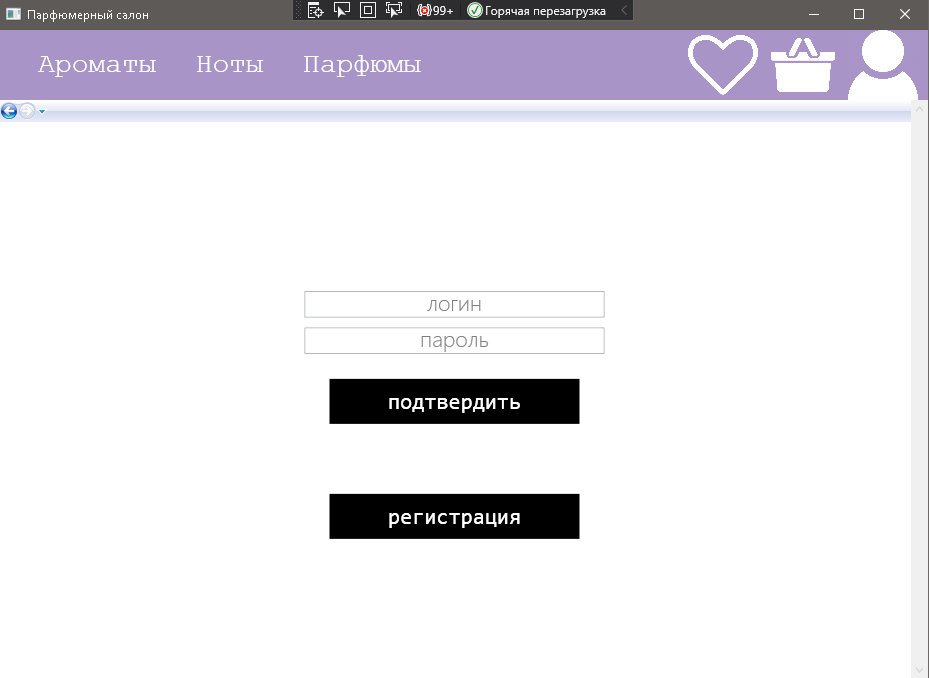


Рисунок 31 - Страница авторизации

При нажатии на кнопку «подтвердить» осуществляется проверка введенных в поля значений, и, если они заполнены правильно, пользователь попадет на страницу с персональными данными (см. Рисунок 32).

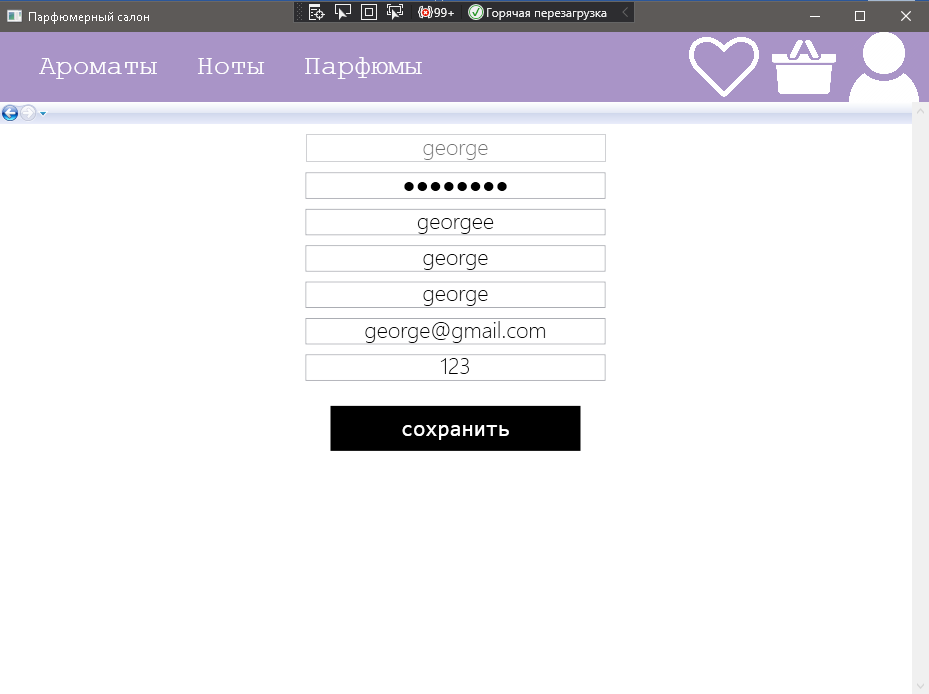


Рисунок 32 - Страница с персональными данными

На этой странице пользователь также имеет возможность изменить такие данные как пароль, имя, фамилию, отчество, почту и телефон.

Если же нажать на кнопку «регистрация», пользователя перенесет на страницу с регистрацией (см. Рисунок 33).

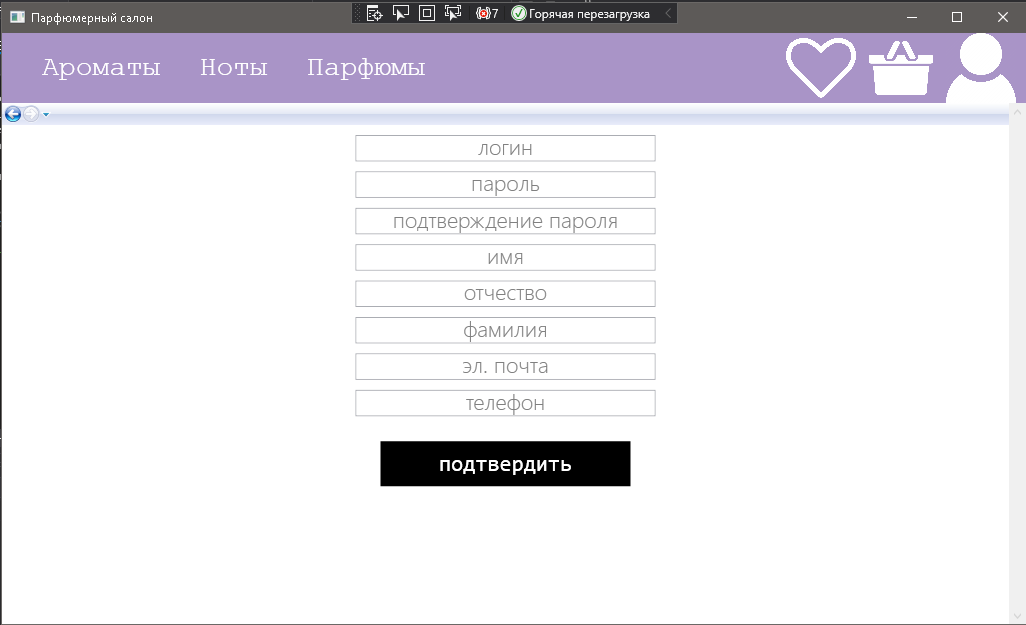


Рисунок 33 - Страница регистрации

Идентификация пользователя осуществляется если:

* регистрационные данные введены и продублированы (пароль) без ошибок, и добавлены в базу данных. После этого пользователя перенаправляет на страницу с персональными данными. Если же в них допущена ошибка, то появляется окно с предупреждением о неправильном вводе регистрационных данных и предоставляется повторная возможность ввода данных;
* авторизация проходит по регистрационным данным и возможна, если они совпадают с данными базы данных. После этого пользователя перенаправляют на страницу с персональными данными. Если же в них допущена ошибка, то появляется окно с предупреждением о неправильном вводе регистрационных данных и предоставляется повторная возможность ввода данных;
* регистрационный логин идентичен и исключает возможности использоваться другим пользователем.

При нажатии на кнопку «Ноты», пользователя перенаправит на страницу со списком групп нот (см. Рисунок 34), а при нажатии на какую-либо группу из списка, пользователя перенаправит на страницу со списком нот соответствующей группы (см. Рисунок 35).

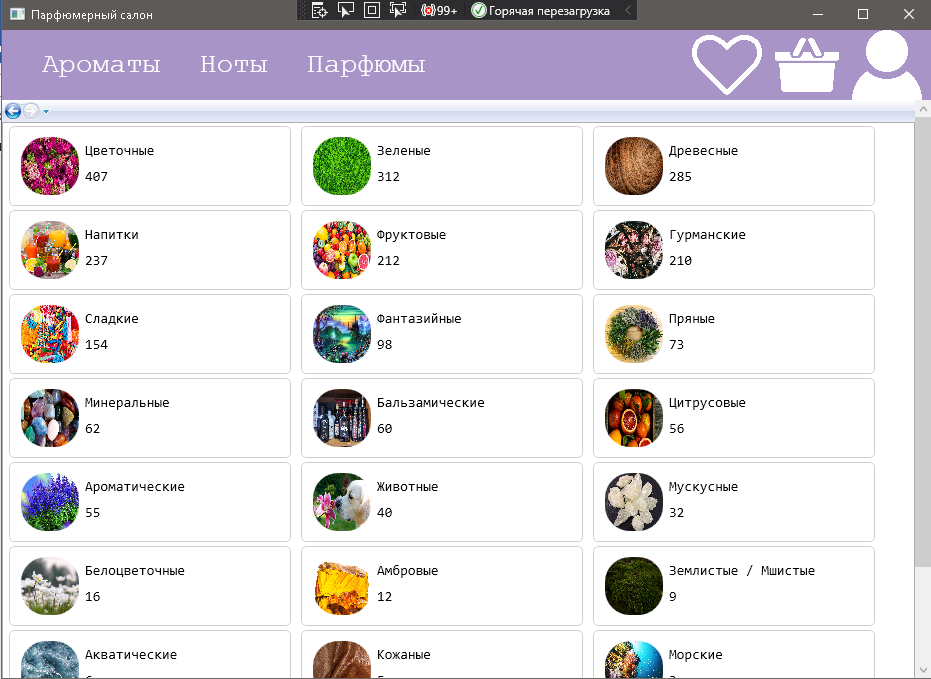


Рисунок 34 - Страница со списком групп нот

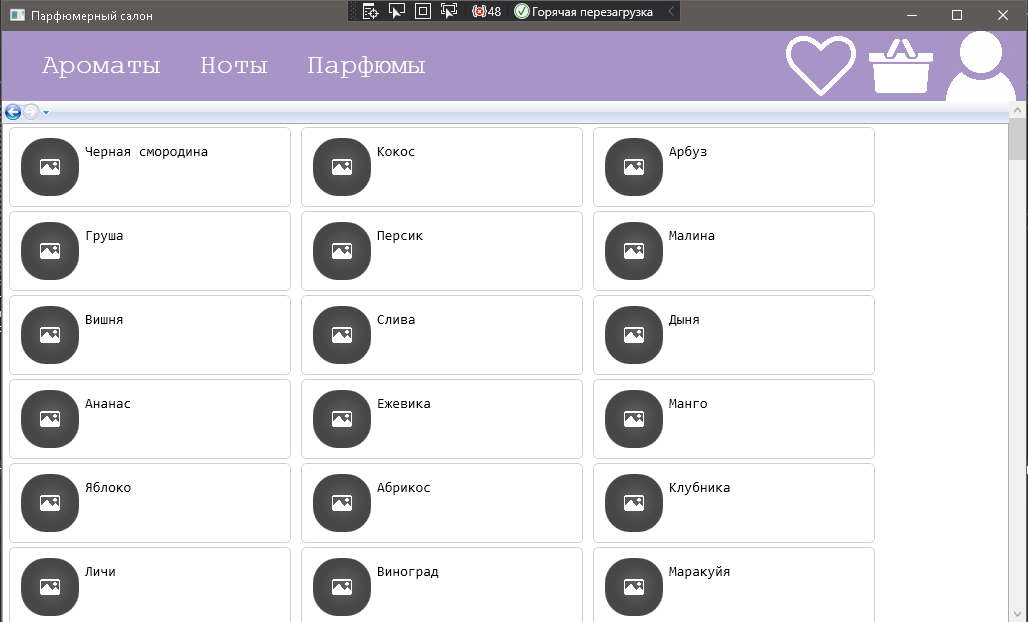


Рисунок 35 - Страница со списком нот при нажатии на группы "Фруктовые"

При нажатии на кнопку «Парфюмы», пользователя перенаправит на домашнюю страницу (см. Рисунок 24).

При нажатии на иконку «Избранное» («Сердечко»), пользователя перенаправит на страницу, представляющую из себя список ароматов, добавленных в избранное (см. Рисунок 36).

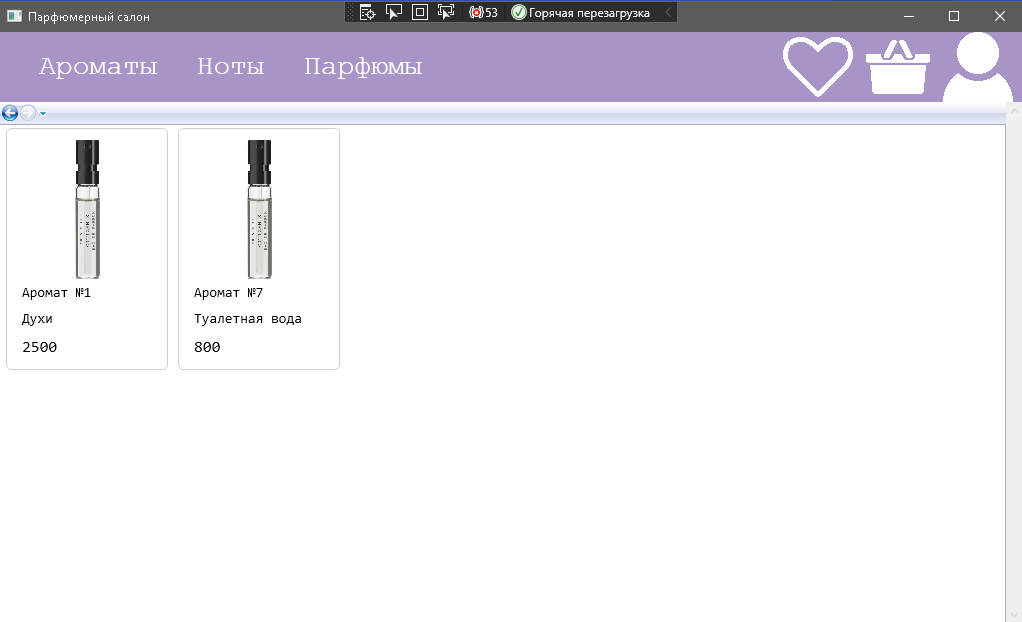


Рисунок 36 - Страница "Избранное"

При нажатии на иконку «Корзины», пользователя перенаправит на страницу с добавленными в нее продуктами (см. Рисунок 37).

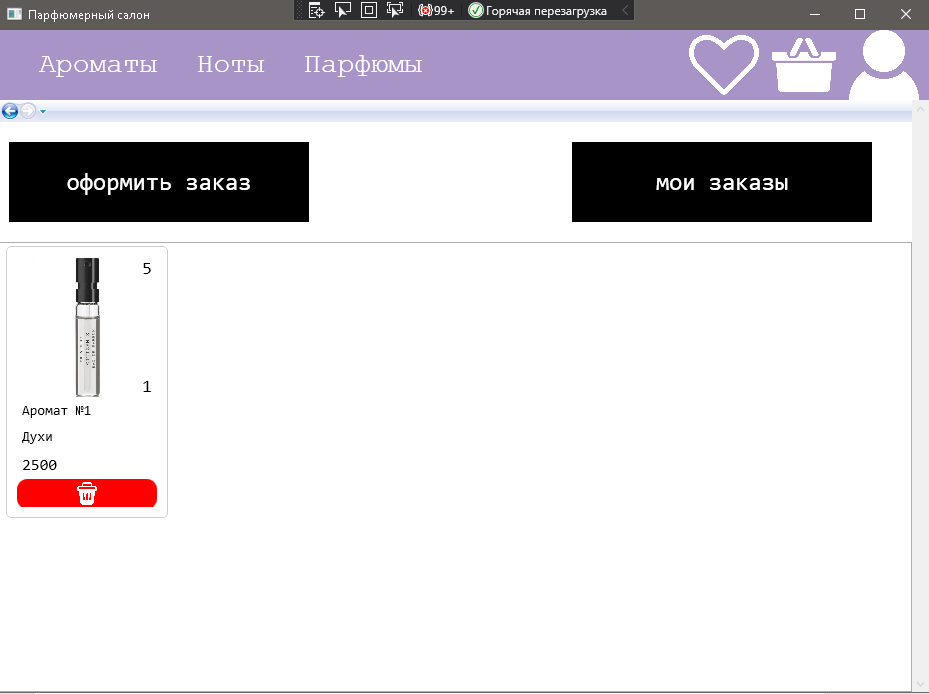


Рисунок 37 - Страница корзины

На этой странице пользователь имеет возможность оформить заказ на приобретение продукции из корзины, просмотреть список уже совершенных заказов, удалить продукт из корзины. Результат нажатия на кнопку «оформить заказ» представлен на рисунке 38.

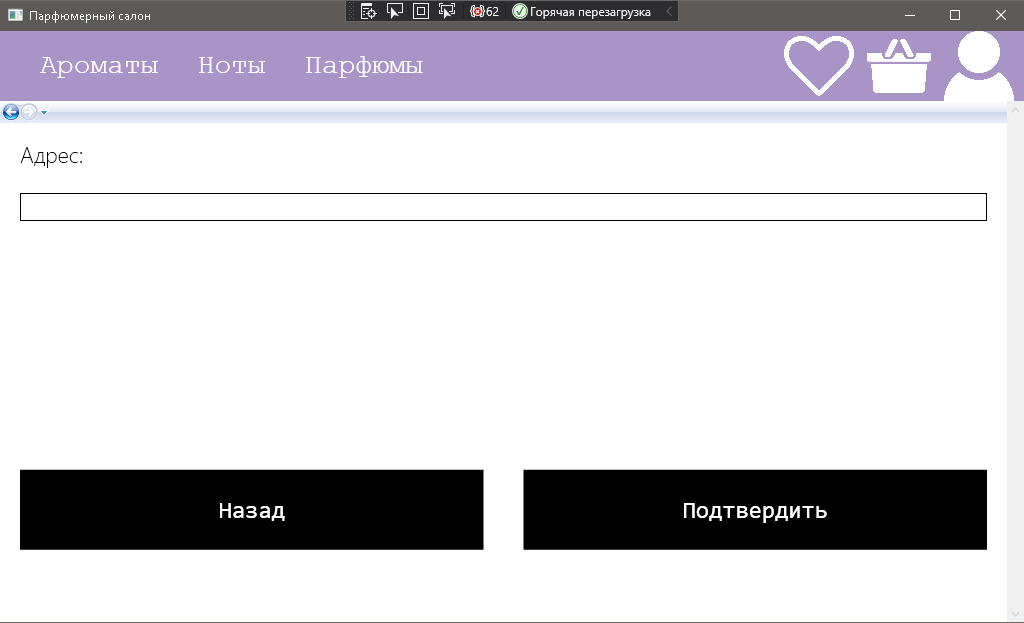


Рисунок 38 - Страница оформления заказа

На этой странице пользователь может отменить оформление заказа, нажав на кнопку «Назад», оформить заказ, нажав кнопку «Подтвердить» при заполнении поля «Адрес».

При нажатии на кнопку «мои заказы» пользователя перенаправит на страницу с историей совершенных им заказов, с возможностью просмотра содержимого каждого их них (см. Рисунки 39-40).

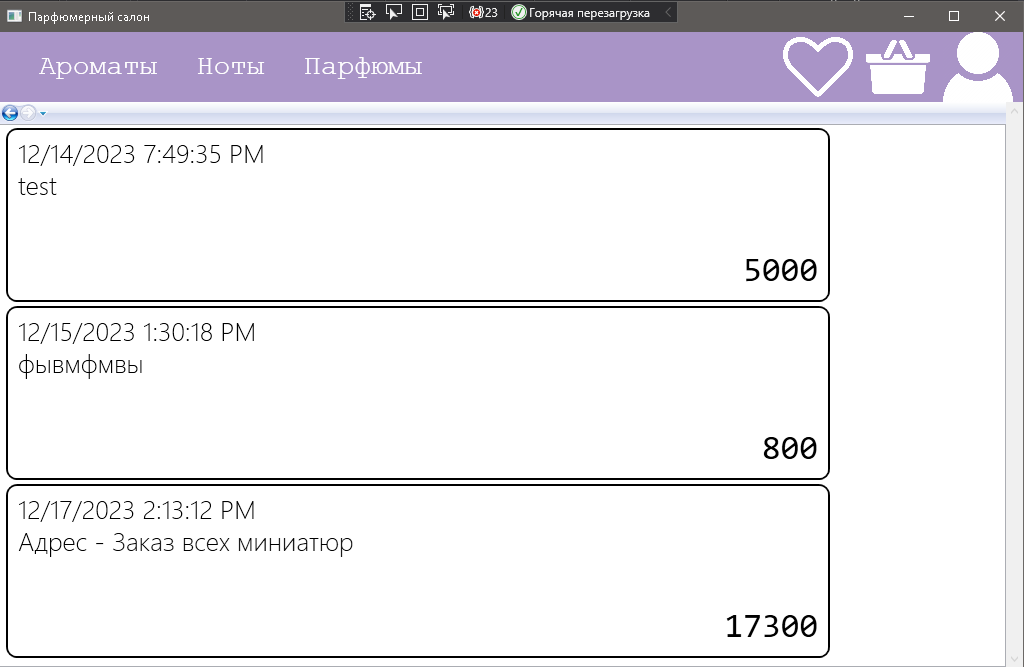


Рисунок 39 - Страница с историей заказов

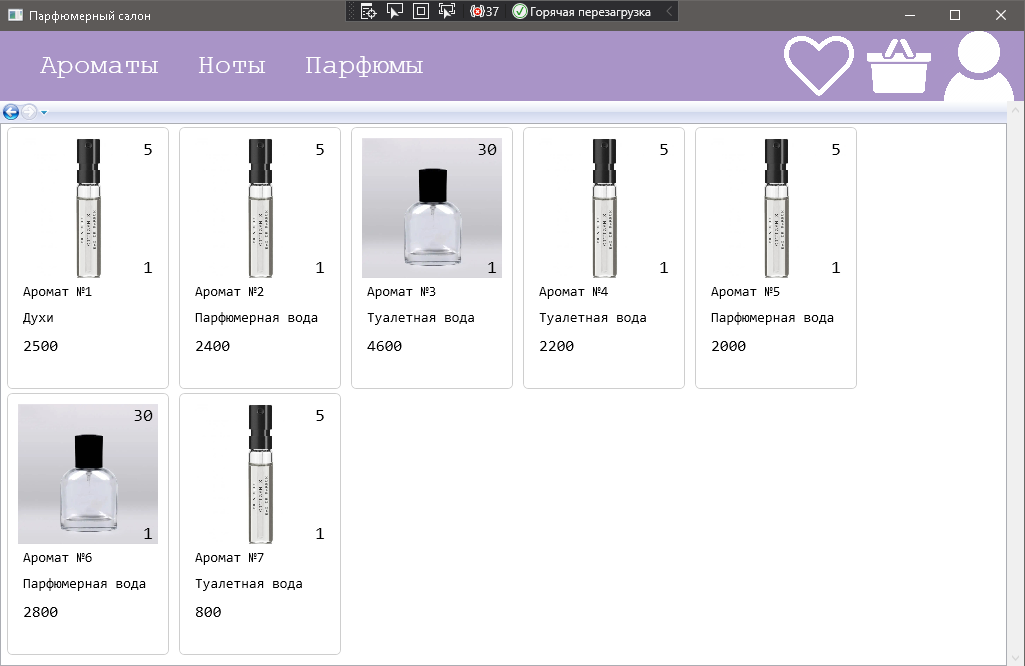


Рисунок 40 - Содержимое заказа от 12/17/2023

В данной автоматизированной системе учета оптовых продаж реализована функция авторизации по ролям для предотвращения несанкционированного изменения данных. Существует три роли авторизации: менеджер, пользователь и кладовщик. Рассмотрим процесс работы при авторизации менеджером.

Результат удаления продукта из корзины представлено на рисунке 41.

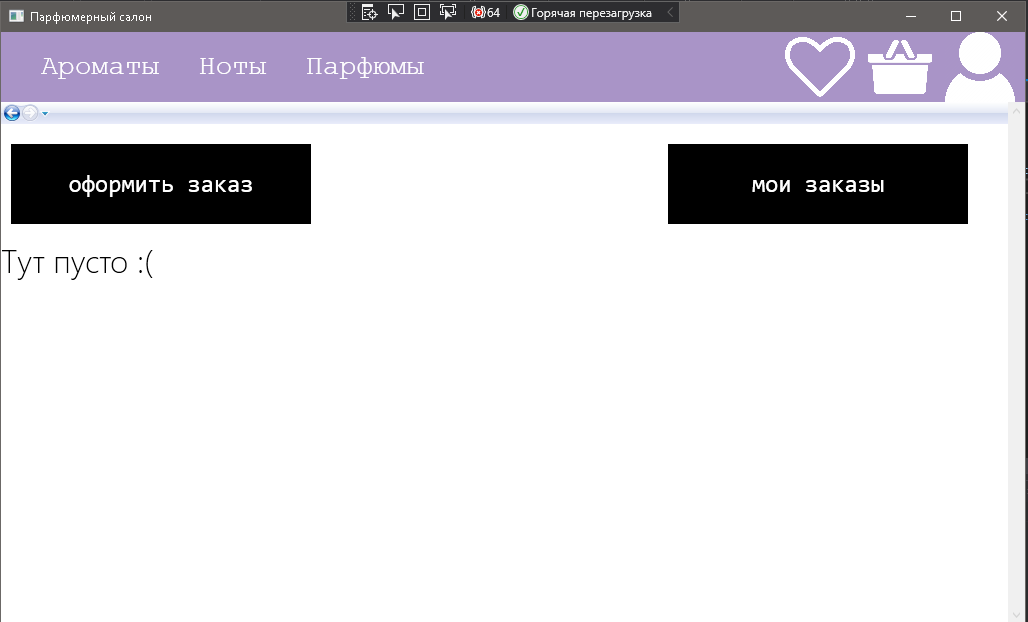
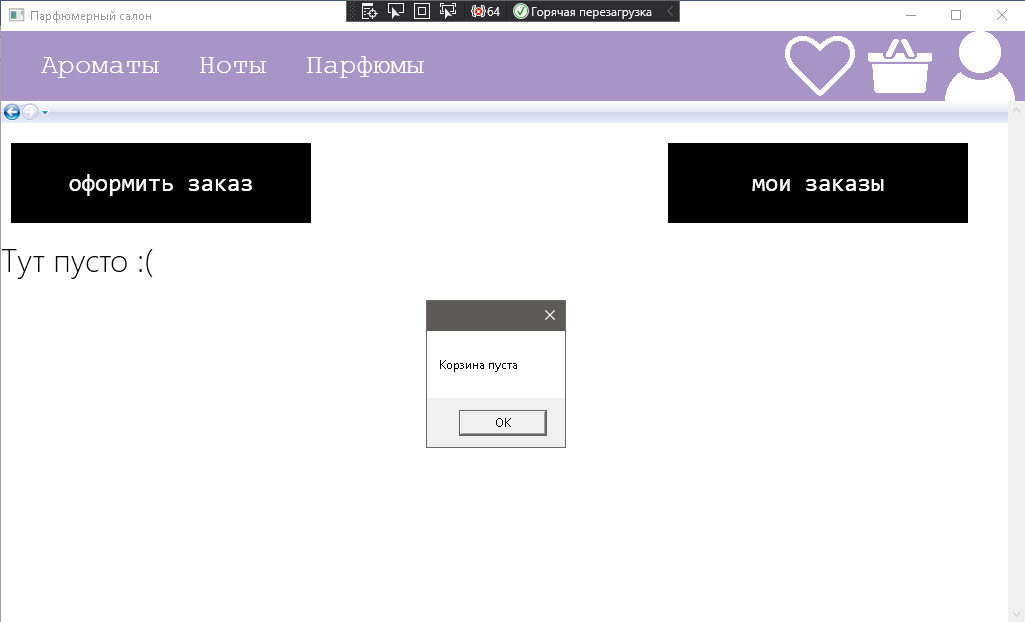


Рисунок 41 - Результат удаления продукта из корзины

## 2.4. Тестирование разработанной автоматизированной системы



Для подтверждения работоспособности системы также рассмотрим все возможные варианты ввода входных данных, тем самым имитируются все возможные ситуации, которые могут произойти при дальнейшей эксплуатации программы.

Рассмотрим вариант, при котором пользователь пробует войти в программу используя неверный логин. В этом случае пользователь получит сообщение о том, что логин введен неверно (Рисунок 42).

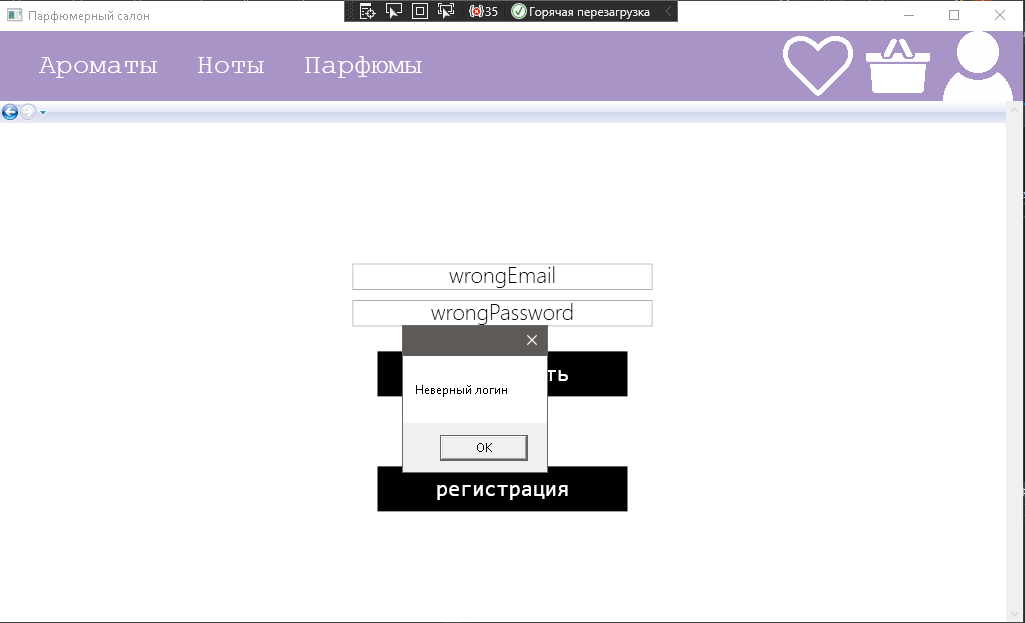


Рисунок 42 - Сообщение о неправильности логина

Однако, в том случае, если логин введен верно, в отличие от пароля, выведется сообщение о неверности пароля (см. Рисунок 43).

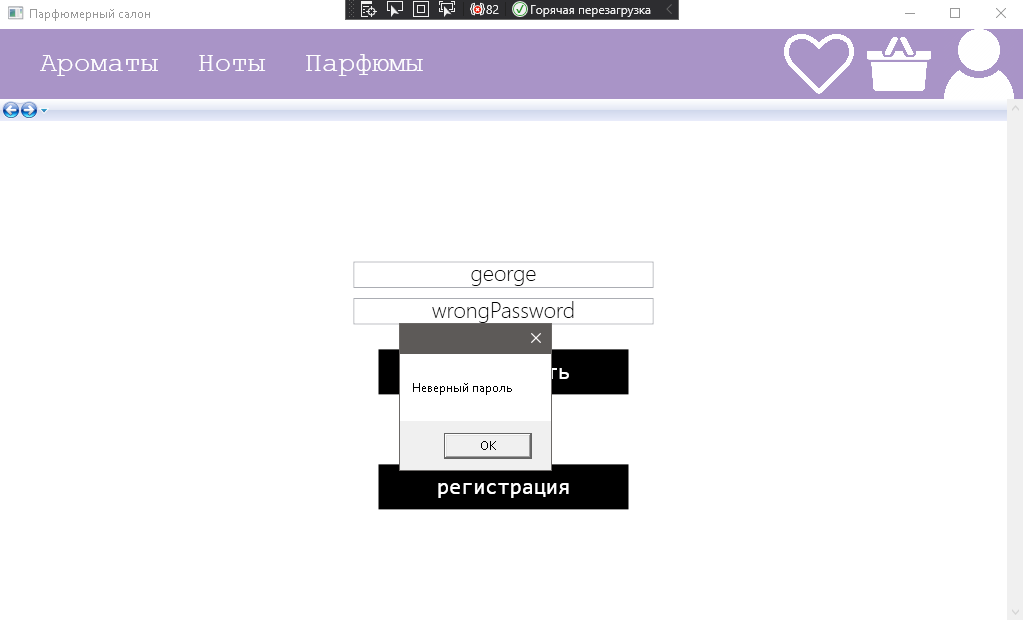


Рисунок 43 - Сообщение о неверности пароля

Рассмотрим следующий пример. Пользователь пытается оформить заказ при пустой корзине. В этом случае он получит сообщение о том, что корзина пуста (см. Рисунок 44).

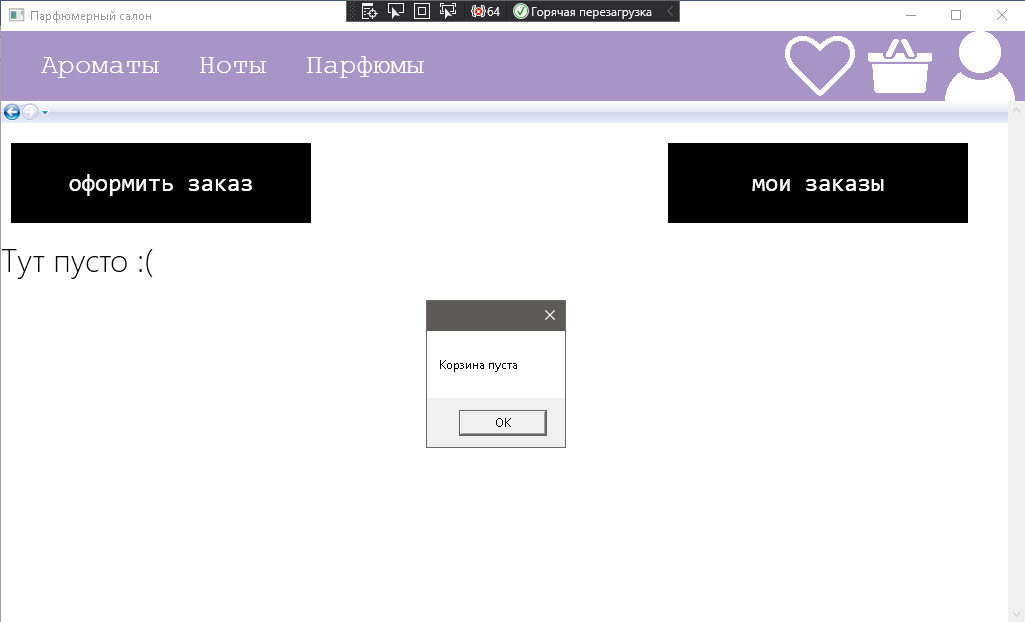


Рисунок 44 - Сообщение о пустой корзине

Рассмотрим следующий пример, в котором пользователь george редактирует пользовательские данные. На рисунке 45 представлено сообщение об успешности изменения данных.

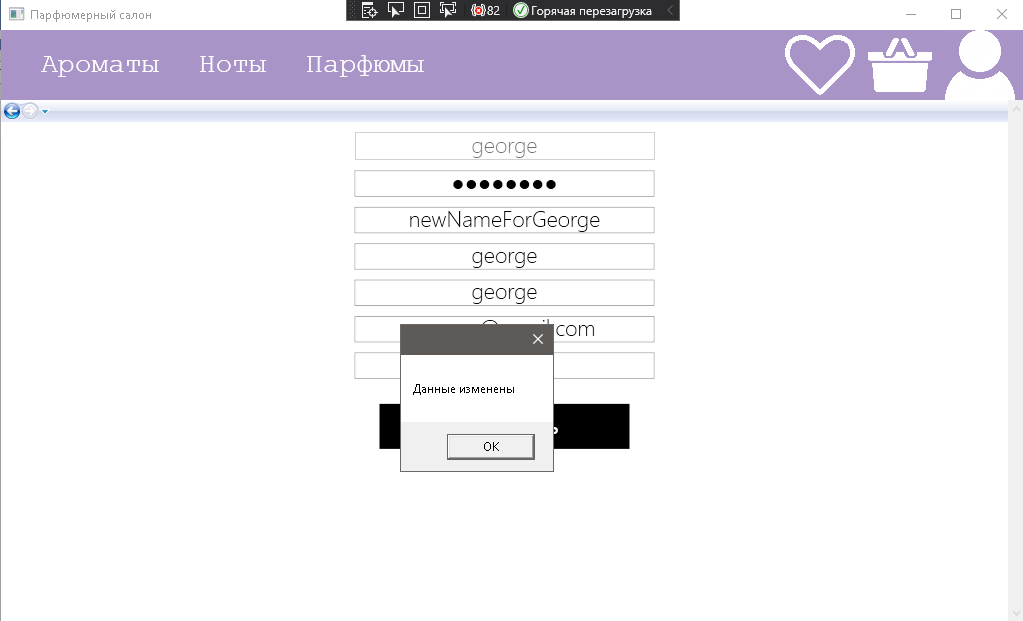


Рисунок 45 - Успешное изменение данных

# ЗАКЛЮЧЕНИЕ

Учитывая, что для разработки решения была использована не профессиональная база, а учебная, некоторые данные не были представлены в качестве сущностей.

Разработанная информационная система отвечает всем требованиям предметной области, таблицы созданной базы данных отвечают требованиям нормализации, что позволяет обеспечить целостность и непротиворечивость информации.

Созданная информационная система позволяет решать все задачи, сформулированные в задании на курсовой проект. Это позволяет сделать вывод, что задание выполнено полностью.

Разработанная база данных легко дополняется при необходимости разработки профессиональной базы данных и может быть использована в дальнейших проектах.

# СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ И ЛИТЕРАТУРЫ

1. Биллиг, В. А. Основы объектного программирования на C# (C# 3.0, Visual Studio 2008) / В.А. Биллиг. - М.: Интернет-университет информационных технологий, Бином. Лаборатория знаний, 2017. - 584 c
2. Брайан, Джонсон Основы Microsoft Visual Studio .NET 2003 / Джонсон Брайан. - М.: Русская Редакция, 2020. - 362 c.
3. Дунаев, В. В. Базы данных. Язык SQL для студента / В.В. Дунаев. - М.: БХВ-Петербург, 2017. - 288 c.
4. Карвин, Билл Программирование баз данных SQL. Типичные ошибки и их устранение / Билл Карвин. - М.: Рид Групп, 2018. - 336 c
5. Культин Н. Microsoft Visual C# в задачах и примерах; БХВ-Петербург М., 2015. - 320 c
6. Маркин, А. В. Построение запросов и программирование на SQL. Учебное пособие / А.В. Маркин. - М.: Диалог-Мифи, 2014. - 384 c.
7. Постолит Visual Studio .NET: разработка приложений баз данных / Постолит, Анатолий. - М.: СПб: БХВ, 2019. - 544 c.
8. Савин, В. И. Организация складской деятельности / В.И. Савин. - М.: Дело и сервис, 2016. - 544 c.
9. Справочник по Transact-SQL (компонент Database Engine) [Электронный ресурс] // Режим доступа: <https://docs.microsoft.com/ru-ru/sql/t-sql/language-reference?view=sql-server-ver15>
10. Общие сведения о Visual Studio | Microsoft Docs [Электронный ресурс] // Режим доступа: <https://docs.microsoft.com/ru-ru/visualstudio/get-started/visual-studio-ide?view=vs-2019>
11. Обзор языка C# - руководство по C# | Microsoft Docs [Электронный ресурс] // Режим доступа: <https://docs.microsoft.com/ru-ru/dotnet/csharp/tour-of-csharp/>

# ПРИЛОЖЕНИЯ

Приложение А. Скрипт на создание БД

CREATE DATABASE [PerfumeSalon]

USE [PerfumeSalon]

CREATE TABLE [dbo].[Products](

[id] [int] IDENTITY(1,1) NOT NULL,

[ProductType] [int] NULL,

[ProductCatery] [int] NULL,

[Aroma] [int] NULL,

[Volume] [int] NULL,

[Price] [int] NULL,

PRIMARY KEY CLUSTERED

(

[id] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON, OPTIMIZE\_FOR\_SEQUENTIAL\_KEY = OFF) ON [PRIMARY]

) ON [PRIMARY]

SET ANSI\_NULLS ON

SET QUOTED\_IDENTIFIER ON

CREATE TABLE [dbo].[ProductsTypes](

[id] [int] IDENTITY(1,1) NOT NULL,

[Type] [nvarchar](32) NULL,

PRIMARY KEY CLUSTERED

(

[id] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON, OPTIMIZE\_FOR\_SEQUENTIAL\_KEY = OFF) ON [PRIMARY]

) ON [PRIMARY]

/\*\*\*\*\*\* Object: Table [dbo].[Users] Script Date: 22.12.2023 17:57:12 \*\*\*\*\*\*/

SET ANSI\_NULLS ON

SET QUOTED\_IDENTIFIER ON

CREATE TABLE [dbo].[Users](

[id] [int] IDENTITY(1,1) NOT NULL,

[Login] [nvarchar](32) NOT NULL,

[Password] [nvarchar](32) NULL,

[Role] [int] NULL,

CONSTRAINT [PK\_Users] PRIMARY KEY CLUSTERED

(

[id] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON, OPTIMIZE\_FOR\_SEQUENTIAL\_KEY = OFF) ON [PRIMARY]

) ON [PRIMARY]

/\*\*\*\*\*\* Object: Table [dbo].[Aromas] Script Date: 22.12.2023 17:57:12 \*\*\*\*\*\*/

SET ANSI\_NULLS ON

SET QUOTED\_IDENTIFIER ON

CREATE TABLE [dbo].[Aromas](

[id] [int] IDENTITY(1,1) NOT NULL,

[Aroma] [nvarchar](64) NULL,

[Description] [nvarchar](512) NULL,

CONSTRAINT [PK\_\_Aromas\_\_3213E83F755EE6A8] PRIMARY KEY CLUSTERED

(

[id] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON, OPTIMIZE\_FOR\_SEQUENTIAL\_KEY = OFF) ON [PRIMARY]

) ON [PRIMARY]

SET ANSI\_NULLS ON

SET QUOTED\_IDENTIFIER ON

CREATE TABLE [dbo].[Customers](

[id] [int] IDENTITY(1,1) NOT NULL,

[FirstName] [nvarchar](32) NULL,

[MiddleName] [nvarchar](32) NULL,

[LastName] [nvarchar](32) NULL,

[Email] [nvarchar](32) NULL,

[Phone] [nvarchar](14) NULL,

[Login] [int] NULL,

CONSTRAINT [PK\_\_Customer\_\_3213E83F634540CB] PRIMARY KEY CLUSTERED

(

[id] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON, OPTIMIZE\_FOR\_SEQUENTIAL\_KEY = OFF) ON [PRIMARY]

) ON [PRIMARY]

/\*\*\*\*\*\* Object: Table [dbo].[Favorites] Script Date: 22.12.2023 17:57:12 \*\*\*\*\*\*/

SET ANSI\_NULLS ON

SET QUOTED\_IDENTIFIER ON

CREATE TABLE [dbo].[Favorites](

[Customer] [int] NULL,

[AromaId] [int] NULL

) ON [PRIMARY]

SET ANSI\_NULLS ON

SET QUOTED\_IDENTIFIER ON

create function [dbo].[GetAromaFromFavorites]

(

@id int

)

returns table as return

(

select A.Aroma from Aromas A

join Products P on P.Aroma = A.id

join ProductsTypes PT on PT.id = P.ProductType

join Favorites F on F.AromaId = A.Id

join Customers C on C.id = F.Customer

join Users U on U.id = C.Login

where U.id = @id

group by A.Aroma

)

SET ANSI\_NULLS ON

SET QUOTED\_IDENTIFIER ON

CREATE TABLE [dbo].[AromasGroups](

[id] [int] IDENTITY(1,1) NOT NULL,

[AromasGroup] [nvarchar](32) NULL,

[Image] [nvarchar](128) NULL,

CONSTRAINT [PK\_\_AromasGr\_\_3213E83FED49DFFF] PRIMARY KEY CLUSTERED

(

[id] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON, OPTIMIZE\_FOR\_SEQUENTIAL\_KEY = OFF) ON [PRIMARY]

) ON [PRIMARY]

/\*\*\*\*\*\* Object: Table [dbo].[AromasGroupsInAromas] Script Date: 22.12.2023 17:57:13 \*\*\*\*\*\*/

SET ANSI\_NULLS ON

SET QUOTED\_IDENTIFIER ON

CREATE TABLE [dbo].[AromasGroupsInAromas](

[Aroma] [int] NULL,

[AromaGroup] [int] NULL

) ON [PRIMARY]

SET ANSI\_NULLS ON

SET QUOTED\_IDENTIFIER ON

CREATE TABLE [dbo].[Volumes](

[id] [int] IDENTITY(1,1) NOT NULL,

[Volume] [int] NULL,

CONSTRAINT [PK\_\_Volumes\_\_3213E83FA0080133] PRIMARY KEY CLUSTERED

(

[id] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON, OPTIMIZE\_FOR\_SEQUENTIAL\_KEY = OFF) ON [PRIMARY]

) ON [PRIMARY]

SET ANSI\_NULLS ON

SET QUOTED\_IDENTIFIER ON

CREATE TABLE [dbo].[Notes](

[id] [int] IDENTITY(1,1) NOT NULL,

[Note] [nvarchar](64) NULL,

[NoteGroup] [int] NULL,

CONSTRAINT [PK\_\_Notes\_\_3213E83F9DFE496C] PRIMARY KEY CLUSTERED

(

[id] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON, OPTIMIZE\_FOR\_SEQUENTIAL\_KEY = OFF) ON [PRIMARY]

) ON [PRIMARY]

SET ANSI\_NULLS ON

SET QUOTED\_IDENTIFIER ON

CREATE TABLE [dbo].[NotesInAromas](

[Aroma] [int] NULL,

[NoteClass] [int] NULL,

[Note] [int] NULL

) ON [PRIMARY]

SET ANSI\_NULLS ON

SET QUOTED\_IDENTIFIER ON

CREATE function [dbo].[SearchAromas]

(

@aromasGroup nvarchar(64),

@productType nvarchar(64),

@search nvarchar(64)

)

returns table as return

(

select A.[id],A.[Aroma],[Description],PT.Type,MIN([Price]) as 'MinPrice', Min(V.Volume) as 'MinVolume'

from Aromas A join Products P on P.Aroma = A.id

join ProductsTypes PT on PT.id = P.ProductType

join AromasGroupsInAromas AGA on AGA.Aroma = A.id

join Volumes V on V.id = P.Volume

join AromasGroups AG on AG.id = AGA.AromaGroup

join NotesInAromas NIA on NIA.Aroma = A.id

join Notes N on N.id = NIA.Note

where AG.AromasGroup like('%'+@aromasGroup+'%')

and PT.Type like('%' +@productType+ '%')

and (A.Description like('%' + @search + '%')

or A.Aroma like('%' + @search + '%')

or AG.AromasGroup like('%' + @search + '%')

or PT.Type like('%' + @search + '%')

or N.Note like('%' + @search + '%'))

group by A.[id], A.[Aroma],[Description], PT.Type

)

SET ANSI\_NULLS ON

SET QUOTED\_IDENTIFIER ON

CREATE TABLE [dbo].[ProductsCateries](

[id] [int] IDENTITY(1,1) NOT NULL,

[Catery] [nvarchar](32) NULL,

PRIMARY KEY CLUSTERED

(

[id] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON, OPTIMIZE\_FOR\_SEQUENTIAL\_KEY = OFF) ON [PRIMARY]

) ON [PRIMARY]

SET ANSI\_NULLS ON

SET QUOTED\_IDENTIFIER ON

CREATE TABLE [dbo].[Baskets](

[Customer] [int] NULL,

[ProductId] [int] NULL,

[Quantity] [int] NULL

) ON [PRIMARY]

SET ANSI\_NULLS ON

SET QUOTED\_IDENTIFIER ON

create function [dbo].[GetProductsFromBasket]

(

@id int

)

returns table as return

(

select P.id,PT.Type,PC.Catery,A.Aroma,V.Volume,Price,B.Quantity from Products P

join ProductsTypes PT on PT.id = P.ProductType

join ProductsCateries PC on PC.id = P.ProductCatery

join Volumes V on V.id = P.Volume

join Aromas A on A.id = P.Aroma

join Baskets B on B.ProductId = P.id

join Customers C on C.id = B.Customer

join Users U on U.id = C.Login

where U.id = @id

group by P.id,PT.Type,PC.Catery,A.Aroma,V.Volume,Price,B.Quantity

)

/\*\*\*\*\*\* Object: UserDefinedFunction [dbo].[IsLoginTaken] Script Date: 22.12.2023 17:57:13 \*\*\*\*\*\*/

SET ANSI\_NULLS ON

SET QUOTED\_IDENTIFIER ON

create function [dbo].[IsLoginTaken]

(

@login nvarchar(32)

)

returns table as return

(select \* from [dbo].[Users]

where Login = @login)

/\*\*\*\*\*\* Object: UserDefinedFunction [dbo].[IsLoginExists] Script Date: 22.12.2023 17:57:13 \*\*\*\*\*\*/

SET ANSI\_NULLS ON

SET QUOTED\_IDENTIFIER ON

create function [dbo].[IsLoginExists]

(

@login nvarchar(32)

)

returns table as return

(select \* from [dbo].[Users]

where Login = @login)

/\*\*\*\*\*\* Object: UserDefinedFunction [dbo].[IsEmailTaken] Script Date: 22.12.2023 17:57:13 \*\*\*\*\*\*/

SET ANSI\_NULLS ON

SET QUOTED\_IDENTIFIER ON

create function [dbo].[IsEmailTaken]

(

@email nvarchar(32)

)

returns table as return

(

select \* from [dbo].[Customers]

where Email = @email)

/\*\*\*\*\*\* Object: UserDefinedFunction [dbo].[IsPhoneTaken] Script Date: 22.12.2023 17:57:13 \*\*\*\*\*\*/

SET ANSI\_NULLS ON

SET QUOTED\_IDENTIFIER ON

create function [dbo].[IsPhoneTaken]

(

@phone nvarchar(14)

)

returns table as return

(

select \* from [dbo].[Customers]

where Phone = @phone)

/\*\*\*\*\*\* Object: UserDefinedFunction [dbo].[GetProfileByUserId] Script Date: 22.12.2023 17:57:13 \*\*\*\*\*\*/

SET ANSI\_NULLS ON

SET QUOTED\_IDENTIFIER ON

create function [dbo].[GetProfileByUserId]

(

@userId int

)

returns table as return

(

select U.Login, U.Password, C.id, C.FirstName, C.MiddleName, C.LastName, C.Email, C.Phone from Customers C

join Users U on U.id = C.Login

where U.id = @userId

)

/\*\*\*\*\*\* Object: UserDefinedFunction [dbo].[GetFavoriteProducts] Script Date: 22.12.2023 17:57:13 \*\*\*\*\*\*/

SET ANSI\_NULLS ON

SET QUOTED\_IDENTIFIER ON

CREATE function [dbo].[GetFavoriteProducts]

(

@id int

)

returns table as return

(

select A.id, A.Aroma, A.Description, PT.Type, Min(P.Price) as MinPrice from Aromas A

join Products P on P.Aroma = A.id

join ProductsTypes PT on PT.id = P.ProductType

join Favorites F on F.AromaId = A.Id

where F.Customer = @id

group by A.id, A.Aroma, A.Description, PT.Type

)

/\*\*\*\*\*\* Object: UserDefinedFunction [dbo].[GetFavoriteAromas] Script Date: 22.12.2023 17:57:13 \*\*\*\*\*\*/

SET ANSI\_NULLS ON

SET QUOTED\_IDENTIFIER ON

CREATE function [dbo].[GetFavoriteAromas]

(

@id int

)

returns table as return

(

select A.id, A.Aroma, A.Description, PT.Type, Min(P.Price) as MinPrice, Min(V.Volume) as MinVolume

from Aromas A

join Products P on P.Aroma = A.id

join ProductsTypes PT on PT.id = P.ProductType

join Favorites F on F.AromaId = A.Id

join Customers C on C.id = F.Customer

join Users U on U.id = C.Login

join Volumes V on V.id = P.Volume

where U.id = @id

group by A.id, A.Aroma, A.Description, PT.Type

)

/\*\*\*\*\*\* Object: UserDefinedFunction [dbo].[GetProductIdFromFavorites] Script Date: 22.12.2023 17:57:13 \*\*\*\*\*\*/

SET ANSI\_NULLS ON

SET QUOTED\_IDENTIFIER ON

create function [dbo].[GetProductIdFromFavorites]

(

@id int

)

returns table as return

(

select P.id from Aromas A

join Products P on P.Aroma = A.id

join ProductsTypes PT on PT.id = P.ProductType

join Favorites F on F.AromaId = A.Id

join Customers C on C.id = F.Customer

join Users U on U.id = C.Login

where U.id = @id

group by P.id

)

/\*\*\*\*\*\* Object: Table [dbo].[Orders] Script Date: 22.12.2023 17:57:13 \*\*\*\*\*\*/

SET ANSI\_NULLS ON

SET QUOTED\_IDENTIFIER ON

CREATE TABLE [dbo].[Orders](

[id] [int] IDENTITY(1,1) NOT NULL,

[Customer] [int] NULL,

[OrderDate] [datetime] NULL,

[Address] [nvarchar](255) NULL,

[TotalPrice] [int] NULL,

PRIMARY KEY CLUSTERED

(

[id] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON, OPTIMIZE\_FOR\_SEQUENTIAL\_KEY = OFF) ON [PRIMARY]

) ON [PRIMARY]

/\*\*\*\*\*\* Object: Table [dbo].[OrdersInfo] Script Date: 22.12.2023 17:57:13 \*\*\*\*\*\*/

SET ANSI\_NULLS ON

SET QUOTED\_IDENTIFIER ON

CREATE TABLE [dbo].[OrdersInfo](

[id] [int] NULL,

[Product] [int] NULL,

[Quantity] [int] NULL

) ON [PRIMARY]

/\*\*\*\*\*\* Object: UserDefinedFunction [dbo].[ShowOrderInfo] Script Date: 22.12.2023 17:57:13 \*\*\*\*\*\*/

SET ANSI\_NULLS ON

SET QUOTED\_IDENTIFIER ON

CREATE function [dbo].[ShowOrderInfo]

(

@orderId int

)

returns table as return

(

select P.id,PT.Type,PC.Catery,A.Aroma,V.Volume,Price,OI.Quantity from Orders O

join OrdersInfo OI on OI.id = O.id

join Products P on P.id = OI.Product

join ProductsTypes PT on PT.id = P.ProductType

join ProductsCateries PC on PC.id = P.ProductCatery

join Volumes V on V.id = P.Volume

join Aromas A on A.id = P.Aroma

where O.id = @orderId

group by P.id,PT.Type,PC.Catery,A.Aroma,V.Volume,Price,OI.Quantity

)

/\*\*\*\*\*\* Object: UserDefinedFunction [dbo].[ShowOrders] Script Date: 22.12.2023 17:57:13 \*\*\*\*\*\*/

SET ANSI\_NULLS ON

SET QUOTED\_IDENTIFIER ON

CREATE function [dbo].[ShowOrders]

(

@userId int

)

returns table as return

(

select O.id,[OrderDate],[Address],[TotalPrice] from Orders O

join Customers C on C.id = O.Customer

join Users U on U.id = C.Login

where U.id = @userId

)

/\*\*\*\*\*\* Object: View [dbo].[vProducts] Script Date: 22.12.2023 17:57:13 \*\*\*\*\*\*/

SET ANSI\_NULLS ON

SET QUOTED\_IDENTIFIER ON

create view [dbo].[vProducts]

as

select P.id, PT.Type, PC.Catery, A.Aroma, V.Volume, Price from Products P

join Aromas A on A.id = P.Aroma

join ProductsCateries PC on PC.id = P.ProductCatery

join ProductsTypes PT on PT.id = P.ProductType

join Volumes V on V.id = P.Volume

/\*\*\*\*\*\* Object: Table [dbo].[Employees] Script Date: 22.12.2023 17:57:13 \*\*\*\*\*\*/

SET ANSI\_NULLS ON

SET QUOTED\_IDENTIFIER ON

CREATE TABLE [dbo].[Employees](

[id] [int] IDENTITY(1,1) NOT NULL,

[FirstName] [nvarchar](32) NULL,

[MiddleName] [nvarchar](32) NULL,

[LastName] [nvarchar](32) NULL,

[Email] [nvarchar](32) NULL,

[Phone] [nvarchar](11) NULL,

[Address] [nvarchar](255) NULL,

[Login] [int] NULL,

PRIMARY KEY CLUSTERED

(

[id] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON, OPTIMIZE\_FOR\_SEQUENTIAL\_KEY = OFF) ON [PRIMARY]

) ON [PRIMARY]

/\*\*\*\*\*\* Object: Table [dbo].[NotesClasses] Script Date: 22.12.2023 17:57:13 \*\*\*\*\*\*/

SET ANSI\_NULLS ON

SET QUOTED\_IDENTIFIER ON

CREATE TABLE [dbo].[NotesClasses](

[id] [int] IDENTITY(1,1) NOT NULL,

[Class] [nvarchar](32) NULL,

PRIMARY KEY CLUSTERED

(

[id] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON, OPTIMIZE\_FOR\_SEQUENTIAL\_KEY = OFF) ON [PRIMARY]

) ON [PRIMARY]

/\*\*\*\*\*\* Object: Table [dbo].[NotesGroups] Script Date: 22.12.2023 17:57:13 \*\*\*\*\*\*/

SET ANSI\_NULLS ON

SET QUOTED\_IDENTIFIER ON

CREATE TABLE [dbo].[NotesGroups](

[id] [int] IDENTITY(1,1) NOT NULL,

[NotesGroup] [nvarchar](32) NULL,

[Image] [nvarchar](128) NULL,

PRIMARY KEY CLUSTERED

(

[id] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON, OPTIMIZE\_FOR\_SEQUENTIAL\_KEY = OFF) ON [PRIMARY]

) ON [PRIMARY]

/\*\*\*\*\*\* Object: Table [dbo].[Roles] Script Date: 22.12.2023 17:57:13 \*\*\*\*\*\*/

SET ANSI\_NULLS ON

SET QUOTED\_IDENTIFIER ON

CREATE TABLE [dbo].[Roles](

[id] [int] IDENTITY(1,1) NOT NULL,

[Role] [nvarchar](32) NULL,

CONSTRAINT [PK\_\_Roles\_\_3213E83F8069CB3B] PRIMARY KEY CLUSTERED

(

[id] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON, OPTIMIZE\_FOR\_SEQUENTIAL\_KEY = OFF) ON [PRIMARY]

) ON [PRIMARY]

ALTER TABLE [dbo].[AromasGroupsInAromas] WITH CHECK ADD CONSTRAINT [FK\_\_AromasGro\_\_Aroma\_\_403A8C7D] FOREIGN KEY([Aroma])

REFERENCES [dbo].[Aromas] ([id])

ALTER TABLE [dbo].[AromasGroupsInAromas] CHECK CONSTRAINT [FK\_\_AromasGro\_\_Aroma\_\_403A8C7D]

ALTER TABLE [dbo].[AromasGroupsInAromas] WITH CHECK ADD CONSTRAINT [FK\_\_AromasGro\_\_Aroma\_\_534D60F1] FOREIGN KEY([AromaGroup])

REFERENCES [dbo].[AromasGroups] ([id])

ALTER TABLE [dbo].[AromasGroupsInAromas] CHECK CONSTRAINT [FK\_\_AromasGro\_\_Aroma\_\_534D60F1]

ALTER TABLE [dbo].[Baskets] WITH CHECK ADD FOREIGN KEY([Customer])

REFERENCES [dbo].[Customers] ([id])

ALTER TABLE [dbo].[Customers] WITH CHECK ADD CONSTRAINT [FK\_Customers\_Users] FOREIGN KEY([Login])

REFERENCES [dbo].[Users] ([id])

ALTER TABLE [dbo].[Customers] CHECK CONSTRAINT [FK\_Customers\_Users]

ALTER TABLE [dbo].[Employees] WITH CHECK ADD CONSTRAINT [FK\_Employees\_Users] FOREIGN KEY([Login])

REFERENCES [dbo].[Users] ([id])

ALTER TABLE [dbo].[Employees] CHECK CONSTRAINT [FK\_Employees\_Users]

ALTER TABLE [dbo].[Favorites] WITH CHECK ADD FOREIGN KEY([Customer])

REFERENCES [dbo].[Customers] ([id])

ALTER TABLE [dbo].[Notes] WITH CHECK ADD CONSTRAINT [FK\_\_Notes\_\_NoteGroup\_\_34C8D9D1] FOREIGN KEY([NoteGroup])

REFERENCES [dbo].[NotesGroups] ([id])

ALTER TABLE [dbo].[Notes] CHECK CONSTRAINT [FK\_\_Notes\_\_NoteGroup\_\_34C8D9D1]

ALTER TABLE [dbo].[NotesInAromas] WITH CHECK ADD CONSTRAINT [FK\_\_NotesInAr\_\_Aroma\_\_44FF419A] FOREIGN KEY([Aroma])

REFERENCES [dbo].[Aromas] ([id])

ALTER TABLE [dbo].[NotesInAromas] CHECK CONSTRAINT [FK\_\_NotesInAr\_\_Aroma\_\_44FF419A]

ALTER TABLE [dbo].[NotesInAromas] WITH CHECK ADD FOREIGN KEY([NoteClass])

REFERENCES [dbo].[NotesClasses] ([id])

ALTER TABLE [dbo].[NotesInAromas] WITH CHECK ADD CONSTRAINT [FK\_\_NotesInAro\_\_Note\_\_3B75D760] FOREIGN KEY([Note])

REFERENCES [dbo].[Notes] ([id])

ALTER TABLE [dbo].[NotesInAromas] CHECK CONSTRAINT [FK\_\_NotesInAro\_\_Note\_\_3B75D760]

ALTER TABLE [dbo].[Orders] WITH CHECK ADD CONSTRAINT [FK\_\_Orders\_\_Customer\_\_47DBAE45] FOREIGN KEY([Customer])

REFERENCES [dbo].[Customers] ([id])

ALTER TABLE [dbo].[Orders] CHECK CONSTRAINT [FK\_\_Orders\_\_Customer\_\_47DBAE45]

ALTER TABLE [dbo].[OrdersInfo] WITH CHECK ADD FOREIGN KEY([Product])

REFERENCES [dbo].[Products] ([id])

ALTER TABLE [dbo].[OrdersInfo] WITH CHECK ADD FOREIGN KEY([id])

REFERENCES [dbo].[Orders] ([id])

ALTER TABLE [dbo].[Products] WITH CHECK ADD CONSTRAINT [FK\_\_Products\_\_Aroma\_\_4AB81AF0] FOREIGN KEY([Aroma])

REFERENCES [dbo].[Aromas] ([id])

ALTER TABLE [dbo].[Products] CHECK CONSTRAINT [FK\_\_Products\_\_Aroma\_\_4AB81AF0]

ALTER TABLE [dbo].[Products] WITH CHECK ADD FOREIGN KEY([ProductType])

REFERENCES [dbo].[ProductsTypes] ([id])

ALTER TABLE [dbo].[Products] WITH CHECK ADD FOREIGN KEY([ProductCatery])

REFERENCES [dbo].[ProductsCateries] ([id])

ALTER TABLE [dbo].[Products] WITH CHECK ADD CONSTRAINT [FK\_\_Products\_\_Volume\_\_46E78A0C] FOREIGN KEY([Volume])

REFERENCES [dbo].[Volumes] ([id])

ALTER TABLE [dbo].[Products] CHECK CONSTRAINT [FK\_\_Products\_\_Volume\_\_46E78A0C]

ALTER TABLE [dbo].[Users] WITH CHECK ADD CONSTRAINT [FK\_\_Users\_\_Role\_\_267ABA7A] FOREIGN KEY([Role])

REFERENCES [dbo].[Roles] ([id])

ALTER TABLE [dbo].[Users] CHECK CONSTRAINT [FK\_\_Users\_\_Role\_\_267ABA7A]

/\*\*\*\*\*\* Object: StoredProcedure [dbo].[AddProductToBasket] Script Date: 22.12.2023 17:57:13 \*\*\*\*\*\*/

SET ANSI\_NULLS ON

SET QUOTED\_IDENTIFIER ON

CREATE proc [dbo].[AddProductToBasket]

(

@userId int,

@productId int

)

as

begin

declare @cId int

set @cId = (select C.id from Customers C

join Users U on U.id = C.Login

where U.id = @userId)

declare @q int

set @q = 0

set @q = @q + ISNULL((select Quantity from Baskets B

join Customers C on C.id = B.Customer

join Users U on U.id = C.Login

where U.id = @userId

and B.ProductId = @productId),0)

if (@q = 0)

begin

insert into Baskets

values (@cId, @productId, 1)

end;

else

begin

update Baskets

set Quantity = Quantity + 1

where Customer = @cId

and ProductId = @productId

end;

end;

/\*\*\*\*\*\* Object: StoredProcedure [dbo].[CreateOrder] Script Date: 22.12.2023 17:57:13 \*\*\*\*\*\*/

SET ANSI\_NULLS ON

SET QUOTED\_IDENTIFIER ON

create proc [dbo].[CreateOrder]

(

@userId int,

@date Datetime,

@address nvarchar(255)

)

as

begin

declare @cId int

set @cId = (select C.id from Customers C

join Users U on U.id = C.Login

where U.id = @userId)

insert into Orders(Customer, OrderDate, Address, TotalPrice)

values

(@cId, @date, @address, 0)

declare @oId int

set @oId = (select id from Orders

where Customer = @cId

and OrderDate = @date

and Address = @address

and TotalPrice = 0)

end;

/\*\*\*\*\*\* Object: StoredProcedure [dbo].[CreateOrderInfo] Script Date: 22.12.2023 17:57:13 \*\*\*\*\*\*/

SET ANSI\_NULLS ON

SET QUOTED\_IDENTIFIER ON

CREATE proc [dbo].[CreateOrderInfo]

(

@orderId int,

@productId int,

@quantity int

)

as

begin

declare @cId int

set @cId = (select Customer from Orders

where id = @orderId)

insert into OrdersInfo

values

(@orderId, @productId, @quantity)

declare @totalPrice int

set @totalPrice = (select Price from Products

where id = @productId) \* @quantity

update Orders

set TotalPrice = TotalPrice + @totalPrice

where id = @orderId

delete from Baskets

where Customer = @cId

and ProductId = @productId

end;

/\*\*\*\*\*\* Object: StoredProcedure [dbo].[DeleteFromBasket] Script Date: 22.12.2023 17:57:13 \*\*\*\*\*\*/

SET ANSI\_NULLS ON

SET QUOTED\_IDENTIFIER ON

create proc [dbo].[DeleteFromBasket]

(

@userId int,

@productId int

)

as

begin

declare @cId int

set @cId = (select C.id from Customers C

join Users U on U.id = C.Login

where U.id = @userId)

delete from Baskets

where Customer = @cId

and ProductId = @productId

end;

/\*\*\*\*\*\* Object: StoredProcedure [dbo].[EditUser] Script Date: 22.12.2023 17:57:13 \*\*\*\*\*\*/

SET ANSI\_NULLS ON

SET QUOTED\_IDENTIFIER ON

create procedure [dbo].[EditUser]

(

@id int,

@login nvarchar(32),

@password nvarchar(32),

@f nvarchar(32),

@m nvarchar(32),

@l nvarchar(32),

@email nvarchar(32),

@phone nvarchar(32)

)

as begin

update Customers

set FirstName = @f

where id = @id

update Customers

set MiddleName = @m

where id = @id

update Customers

set LastName = @l

where id = @id

update Customers

set Email = @email

where id = @id

update Customers

set Phone = @phone

where id = @id

update Users

set Password = @password

where Login = @login

end;

/\*\*\*\*\*\* Object: StoredProcedure [dbo].[EditUserWithoutPassword] Script Date: 22.12.2023 17:57:13 \*\*\*\*\*\*/

SET ANSI\_NULLS ON

SET QUOTED\_IDENTIFIER ON

create procedure [dbo].[EditUserWithoutPassword]

(

@id int,

@f nvarchar(32),

@m nvarchar(32),

@l nvarchar(32),

@email nvarchar(32),

@phone nvarchar(32)

)

as begin

update Customers

set FirstName = @f

where id = @id

update Customers

set MiddleName = @m

where id = @id

update Customers

set LastName = @l

where id = @id

update Customers

set Email = @email

where id = @id

update Customers

set Phone = @phone

where id = @id

end;

/\*\*\*\*\*\* Object: StoredProcedure [dbo].[MarkAsFavorite] Script Date: 22.12.2023 17:57:13 \*\*\*\*\*\*/

SET ANSI\_NULLS ON

SET QUOTED\_IDENTIFIER ON

CREATE proc [dbo].[MarkAsFavorite]

(

@userId int,

@aroma nvarchar(64)

)

as

begin

declare @customerId int

set @customerId = (select C.id from Customers C

join Users U on U.id = C.Login

where U.id = @userId)

declare @aromaId int

set @aromaId = (select A.id from Aromas A

where A.Aroma = @aroma)

if exists(

select TOP(1) F.Customer, F.AromaId from Favorites F

join Aromas A on A.id = F.AromaId

join Products P on P.Aroma = A.id

join Customers C on C.id = F.Customer

join Users U on U.id = C.Login

where U.id = @userId

and A.Aroma = @aroma

)

begin

delete from Favorites

where Customer = @customerId

and AromaId = @aromaId

end;

else

begin

insert into Favorites values

(@customerId, @aromaId)

end;

end;

/\*\*\*\*\*\* Object: StoredProcedure [dbo].[RegisterNewUser] Script Date: 22.12.2023 17:57:13 \*\*\*\*\*\*/

SET ANSI\_NULLS ON

SET QUOTED\_IDENTIFIER ON

create procedure [dbo].[RegisterNewUser]

(

@login nvarchar(32),

@password nvarchar(32),

@f nvarchar(32),

@m nvarchar(32),

@l nvarchar(32),

@email nvarchar(32),

@phone nvarchar(32)

)

as begin

if not exists(

select \* from [dbo].[Customers]

where [FirstName] = @f

and [MiddleName] = @m

and [LastName] = @l)

begin

if not exists(

select \* from [dbo].[Users]

where Login = @login)

begin

insert into [dbo].[Users]([Login],[Password],[Role])

values (@login, @password, 2)

declare @userId int

set @userId = (select id from Users

where Login = @Login)

insert into [dbo].[Customers]([FirstName],[MiddleName],[LastName],[Email],[Phone],[Login])

values (@f,@m,@l,@email,@phone,@userId)

end;

end;

end;

Приложение Б №1. Код класса Connection.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace PerfumeSalon.Models

{

internal class Connection

{

//public static string connectionString = @"Data Source=172.20.105.2;User ID=9po11-20-15;Password=Nie3Ieh9;Connect Timeout=5;Encrypt=False;Initial Catalog=PerfumeSalon";

public static string connectionString = @"Data Source = MINT; User ID=vyach; Password=1;Connect Timeout=5;Encrypt=False;Initial Catalog=PerfumeSalon";

}

}

Приложение Б №2. Код класса Aromas.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Media.Imaging;

namespace PerfumeSalon.Classes

{

public class Aromas

{

public int Id { get; set; }

public string Aroma { get; set; }

public string Description { get; set; }

public string Type { get; set; }

public int Price { get; set; }

public BitmapImage Image { get; set; }

public Aromas(int id, string aroma, string description, string type, int price, int volume)

{

Id = id;

Aroma = aroma;

Description = description;

Type = type;

Price = price;

Uri uri = new Uri($"/Resources/Volumes/{volume}.jpg", UriKind.Relative);

Image = new BitmapImage(uri);

}

public Aromas() { }

}

}

Приложение Б №3. Код класса AromasGroups.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Controls;

using System.Windows.Media;

using System.Windows.Media.Imaging;

namespace PerfumeSalon.Classes

{

public class AromasGroups

{

public int Id { get; set; }

public string AromasGroup { get; set; }

public BitmapImage Image { get; set; }

public int Quantity { get; set; }

public AromasGroups(int id, string aromasGroup, Uri path, int quantity)

{

Id = id;

AromasGroup = aromasGroup;

Image = new BitmapImage(path);

Quantity = quantity;

}

}

}

Приложение Б №4. Код класса Baskets.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace PerfumeSalon.Classes

{

public class Baskets

{

public int CustomerId { get; set; }

public int ProductId { get; set; }

public int Quanity { get; set; }

public Baskets() { }

public Baskets(int customerId, int productId, int quantity)

{

CustomerId = customerId;

ProductId = productId;

Quanity = quantity;

}

}

}

Приложение Б №5. Код класса cmdDelete.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Input;

using GalaSoft.MvvmLight.Command;

namespace PerfumeSalon.Classes

{

public class cmdDelete

{

public ICommand deleteProduct

{

get

{

return new RelayCommand<object>(DeleteProduct);

}

}

public void DeleteProduct(object parameter)

{

var productId = Convert.ToInt32(parameter);

dbBaskets.DeleteProductFromBasket(UserId.userId, productId);

Pages.BasketPage.products = dbBaskets.GetProductsFromBasket(UserId.userId);

FrameClass.BodyFrame.Navigate(new Pages.BasketPage());

}

}

}

Приложение Б №6. Код класса Customers.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace PerfumeSalon.Classes

{

public class Customers

{

public int Id { get; set; }

public string FirstName { get; set; }

public string MiddleName { get; set; }

public string LastName { get; set; }

public string Email { get; set; }

public string Phone { get; set; }

public string Login { get; set; }

public Customers(int id, string fname, string mname, string lname, string email, string phone, string login)

{

Id = id;

FirstName = fname;

MiddleName = mname;

LastName = lname;

Email = email;

Phone = phone;

Login = login;

}

public Customers() { }

}

}

Приложение Б №7. Код класса FrameClass.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Controls;

namespace PerfumeSalon.Classes

{

class FrameClass

{

public static Frame BodyFrame;

}

}

Приложение Б №8. Код класса Notes.cs

using System;

using System.Collections;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Media.Imaging;

namespace PerfumeSalon.Classes

{

public class Notes

{

public int Id { get; set; }

public string Note { get; set; }

public int NoteGroup { get; set; }

public BitmapImage Image { get; set; }

public Notes(int id, string note, int noteGroup)

{

Id = id;

Note = note;

NoteGroup = noteGroup;

}

}

}

Приложение Б №9. Код класса NotesGroups.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Controls;

using System.Windows.Media;

using System.Windows.Media.Imaging;

namespace PerfumeSalon.Classes

{

public class NotesGroups

{

public int Id { get; set; }

public string NotesGroup { get; set; }

public BitmapImage Image { get; set; }

public int Quantity { get; set; }

public NotesGroups(int id, string notesGroup, Uri path, int quantity)

{

Id = id;

NotesGroup = notesGroup;

Image = new BitmapImage(path);

Quantity = quantity;

}

}

}

Приложение Б №10. Код класса NotesInAromas.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace PerfumeSalon.Classes

{

public class NotesInAromas

{

public int NoteClass { get; set; }

public string Note { get; set; }

public NotesInAromas(int noteClass ,string note)

{

NoteClass = noteClass;

Note = note;

}

}

}

Приложение Б №11. Код класса Orders.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace PerfumeSalon.Classes

{

public class Orders

{

public int Id { get; set; }

public DateTime Date { get; set; }

public string Address { get; set; }

public int TotalPrice { get; set; }

public Orders() { }

public Orders(int id, DateTime date, string address, int totalprice)

{

Id = id;

Date = date;

Address = address;

TotalPrice = totalprice;

}

}

}

Приложение Б №12. Код класса Products.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Media.Imaging;

namespace PerfumeSalon.Classes

{

public class Products

{

public int Id { get; set; }

public string Type { get; set; }

public string Category { get; set; }

public string Aroma { get; set; }

public int Volume { get; set; }

public int Price { get; set; }

public int Quantity { get; set; }

public BitmapImage Image { get; set; }

public Products() { }

public Products(int id, string type, string category, string aroma, int volume, int price)

{

Id = id;

Type = type;

Category = category;

Aroma = aroma;

Volume = volume;

Price = price;

Uri uri = new Uri($"/Resources/Volumes/{volume}.jpg", UriKind.Relative);

Image = new BitmapImage(uri);

}

public Products(int id, string type, string category, string aroma, int volume, int price, int quantity)

{

Id = id;

Type = type;

Category = category;

Aroma = aroma;

Volume = volume;

Price = price;

Quantity = quantity;

Uri uri = new Uri($"/Resources/Volumes/{volume}.jpg", UriKind.Relative);

Image = new BitmapImage(uri);

}

}

}

Приложение Б №13. Код класса Profiles.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace PerfumeSalon.Classes

{

public class Profiles

{

public string Login { get; set; }

public string Password { get; set; }

public int Id { get; set; }

public string FirstName { get; set; }

public string MiddleName { get; set; }

public string LastName { get; set; }

public string Email { get; set; }

public string Phone { get; set; }

public Profiles(string login, string password, int id,

string fname, string mname, string lname,

string email, string phone)

{

Login = login;

Password = password;

Id = id;

FirstName = fname;

MiddleName = mname;

LastName = lname;

Email = email;

Phone = phone;

}

public Profiles() { }

}

}

Приложение Б №14. Код класса UserId.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace PerfumeSalon.Classes

{

class UserId

{

public static int userId = 0;

public static int roleId = 2;

}

}

Приложение Б №15. Код класса Users.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace PerfumeSalon.Classes

{

public class Users

{

public int Id { get; set; }

public string Login { get; set; }

public string Password { get; set; }

public int Role { get; set; }

public Users(int id, string login, string password, int role)

{

Id = id;

Login = login;

Password = password;

Role = role;

}

public Users() { }

}

}

Приложение Б №16. Код класса Volumes.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace PerfumeSalon.Classes

{

public class Volumes

{

public int Id { get; set; }

public int Volume { get; set; }

public Volumes (int id, int volume)

{

Id = id;

Volume = volume;

}

}

}

Приложение Б №17. Код класса dbAromas.cs

using System;

using System.Collections.Generic;

using System.Data.SqlClient;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using PerfumeSalon.Models;

namespace PerfumeSalon.Classes

{

class dbAromas

{

public static List<Aromas> LoadFromDB(int id)

{

SqlConnection connect = new SqlConnection(Connection.connectionString);

SqlCommand command = new SqlCommand("select A.[id],A.[Aroma],[Description],PT.Type,MIN([Price]), Min(V.Volume)" +

" from Aromas A join Products P on P.Aroma = A.id" +

" join ProductsTypes PT on PT.id = P.ProductType" +

" join AromasGroupsInAromas AGA on AGA.Aroma = A.id" +

" join Volumes V on V.id = P.Volume" +

" where AGA.AromaGroup = @id" +

" group by A.[id], A.[Aroma],[Description], PT.Type", connect);

List<Aromas> spisok = new List<Aromas>();

command.Parameters.AddWithValue("@id", id);

try

{

connect.Open();

SqlDataReader reader = command.ExecuteReader();

while (reader.Read())

{

spisok.Add(new Aromas(reader.GetInt32(0), reader.GetString(1),

reader.GetString(2), reader.GetString(3), reader.GetInt32(4), reader.GetInt32(5)));

}

connect.Close();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message);

}

return spisok;

}

public static Aromas LoadAromaById(int id)

{

SqlConnection connect = new SqlConnection(Connection.connectionString);

SqlCommand command = new SqlCommand("select A.[id],A.[Aroma],[Description],PT.Type,MIN([Price]), Min(V.Volume) " +

" from Aromas A join Products P on P.Aroma = A.id" +

" join ProductsTypes PT on PT.id = P.ProductType" +

" join AromasGroupsInAromas AGA on AGA.Aroma = A.id" +

" join Volumes V on V.id = P.Volume" +

" where AGA.AromaGroup = @id" +

" group by A.[id], A.[Aroma],[Description], PT.Type", connect);

Aromas spisok = new Aromas();

command.Parameters.AddWithValue("@id", id);

try

{

connect.Open();

SqlDataReader reader = command.ExecuteReader();

while (reader.Read())

{

spisok = new Aromas(reader.GetInt32(0), reader.GetString(1),

reader.GetString(2), reader.GetString(3), reader.GetInt32(4), reader.GetInt32(5));

}

connect.Close();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message);

}

return spisok;

}

public static List<Aromas> GetFavoriteAromas(int id)

{

SqlConnection connect = new SqlConnection(Connection.connectionString);

SqlCommand command = new SqlCommand("select \* from GetFavoriteAromas(@id)", connect);

List<Aromas> spisok = new List<Aromas>();

command.Parameters.AddWithValue("@id", id);

try

{

connect.Open();

SqlDataReader reader = command.ExecuteReader();

while (reader.Read())

{

spisok.Add(new Aromas(reader.GetInt32(0), reader.GetString(1),

reader.GetString(2), reader.GetString(3), reader.GetInt32(4), reader.GetInt32(5)));

}

connect.Close();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message);

}

return spisok;

}

public static List<Aromas> SearchAromas(int index, string aromasGroup, string type, string search)

{

if (aromasGroup == "" || aromasGroup == " " || aromasGroup == null)

{

aromasGroup = "%";

}

if (type == "" || type == " " || type == null)

{

type = "%";

}

if (search == "" || search == " " || search == null)

{

search = "%";

}

SqlConnection connect = new SqlConnection(Connection.connectionString);

SqlCommand command = new SqlCommand("select \* from SearchAromas(@aromasGroup,@type,@search)", connect);

switch (index)

{

case 0: command.CommandText += " order by Aroma asc"; break;

case 1: command.CommandText += " order by MinPrice asc"; break;

case 2: command.CommandText += " order by MinPrice desc"; break;

default: command.CommandText += " order by Aroma asc"; break;

}

List<Aromas> spisok = new List<Aromas>();

command.Parameters.AddWithValue("@aromasGroup", aromasGroup);

command.Parameters.AddWithValue("@type", type);

command.Parameters.AddWithValue("@search", search);

try

{

connect.Open();

SqlDataReader reader = command.ExecuteReader();

while (reader.Read())

{

spisok.Add(new Aromas(reader.GetInt32(0), reader.GetString(1),

reader.GetString(2), reader.GetString(3), reader.GetInt32(4), reader.GetInt32(5)));

}

//MessageBox.Show($"{index}\t{aromasGroup}\t{type}\t{search}");

connect.Close();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message);

}

return spisok;

}

}

}

Приложение Б №18. Код класса dbAromasGroups.cs

using PerfumeSalon.Models;

using System;

using System.Collections.Generic;

using System.Data.SqlClient;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Media.Imaging;

namespace PerfumeSalon.Classes

{

internal class dbAromasGroups

{

public static List<AromasGroups> LoadFromDB()

{

SqlConnection connect = new SqlConnection(Connection.connectionString);

SqlCommand command = new SqlCommand("select [id],AG.[AromasGroup],[Image],count([Aroma]) from [dbo].[AromasGroups] AG" +

" left join [dbo].[AromasGroupsInAromas] AGA on AGA.AromaGroup = AG.id" +

" group by [id],AG.[AromasGroup],[Image]", connect);

List<AromasGroups> spisok = new List<AromasGroups>();

try

{

connect.Open();

SqlDataReader reader = command.ExecuteReader();

while (reader.Read())

{

spisok.Add(new AromasGroups(reader.GetInt32(0), reader.GetString(1),

new Uri(reader.GetString(2), UriKind.Relative), reader.GetInt32(3)));

}

connect.Close();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message);

}

return spisok;

}

public static List<string> GetAromasGroupsByAromaId(int id)

{

SqlConnection connect = new SqlConnection(Connection.connectionString);

SqlCommand command = new SqlCommand("select [AromasGroup] from AromasGroups AG" +

" join AromasGroupsInAromas AGA on AGA.AromaGroup = AG.id" +

" join Aromas A on A.id = AGA.Aroma" +

" where A.id = @id", connect);

command.Parameters.AddWithValue("@id",id);

List<string> spisok = new List<string>();

try

{

connect.Open();

SqlDataReader reader = command.ExecuteReader();

while (reader.Read())

{

spisok.Add(reader.GetString(0));

}

connect.Close();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message);

}

return spisok;

}

}

}

Приложение Б №19. Код класса dbBaskets.cs

using System;

using System.Collections.Generic;

using System.Data.SqlClient;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using PerfumeSalon.Models;

namespace PerfumeSalon.Classes

{

public class dbBaskets

{

public static List<Products> GetProductsFromBasket(int id)

{

SqlConnection connect = new SqlConnection(Connection.connectionString);

SqlCommand command = new SqlCommand("select \* from GetProductsFromBasket(@id)", connect);

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

command.Parameters.AddWithValue("@id", id);

try

{

connect.Open();

SqlDataReader reader = command.ExecuteReader();

if (reader.HasRows)

{

while (reader.Read())

{

products.Add(new Products(reader.GetInt32(0), reader.GetString(1),

reader.GetString(2), reader.GetString(3), reader.GetInt32(4),

reader.GetInt32(5), reader.GetInt32(6)));

}

}

connect.Close();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message);

}

return products;

}

public static int GetAromaIdFromProduct(int id)

{

SqlConnection connect = new SqlConnection(Connection.connectionString);

SqlCommand command = new SqlCommand("select A.id from Aromas A" +

" join Products P on P.Aroma = A.id" +

" where P.id = @id", connect);

int aromaId = 0;

command.Parameters.AddWithValue("@id", id);

try

{

connect.Open();

SqlDataReader reader = command.ExecuteReader();

if (reader.HasRows)

{

while (reader.Read())

{

aromaId = reader.GetInt32(0);

}

}

connect.Close();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message);

}

return aromaId;

}

public static void AddProductToBasket(int userId, int productId)

{

SqlConnection connect = new SqlConnection(Connection.connectionString);

SqlCommand command = new SqlCommand("exec AddProductToBasket @userId, @productId", connect);

command.Parameters.AddWithValue("@userId", userId);

command.Parameters.AddWithValue("@productId", productId);

try

{

connect.Open();

command.ExecuteNonQuery();

MessageBox.Show("Продукт добавлен в корзину!");

connect.Close();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message);

}

}

public static void DeleteProductFromBasket(int userId, int productId)

{

SqlConnection connect = new SqlConnection(Connection.connectionString);

SqlCommand command = new SqlCommand("exec DeleteFromBasket @userId, @productId", connect);

command.Parameters.AddWithValue("@userId", userId);

command.Parameters.AddWithValue("@productId", productId);

try

{

connect.Open();

command.ExecuteNonQuery();

connect.Close();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message);

}

}

}

}

Приложение Б №20. Код класса dbNotes.cs

using System;

using System.Collections.Generic;

using System.Data.SqlClient;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using PerfumeSalon.Models;

namespace PerfumeSalon.Classes

{

class dbNotes

{

public static List<Notes> LoadFromDB(int id)

{

SqlConnection connect = new SqlConnection(Connection.connectionString);

SqlCommand command = new SqlCommand("select N.[id],N.[Note], NG.id" +

" from Notes N join[dbo].[NotesGroups] NG on NG.id = N.NoteGroup" +

" where NG.id = @id", connect);

List<Notes> spisok = new List<Notes>();

command.Parameters.AddWithValue("@id", id);

try

{

connect.Open();

SqlDataReader reader = command.ExecuteReader();

while (reader.Read())

{

spisok.Add(new Notes(reader.GetInt32(0), reader.GetString(1), reader.GetInt32(2)));

}

connect.Close();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message);

}

return spisok;

}

}

}

Приложение Б №21. Код класса dbNotesGroups.cs

using System;

using System.Collections.Generic;

using System.Data.SqlClient;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using PerfumeSalon.Models;

using System.Windows.Controls;

using System.Windows.Media.Imaging;

namespace PerfumeSalon.Classes

{

public class dbNotesGroups

{

public static List<NotesGroups> LoadFromDB()

{

SqlConnection connect = new SqlConnection(Connection.connectionString);

SqlCommand command = new SqlCommand("select NG.[id],NG.[NotesGroup],[Image], count(N.[id])" +

" from NotesGroups NG left join Notes N on N.NoteGroup = NG.id" +

" group by NG.[id], NG.[NotesGroup],[Image]" +

" order by count(N.[id]) desc", connect);

List<NotesGroups> spisok = new List<NotesGroups>();

try

{

connect.Open();

SqlDataReader reader = command.ExecuteReader();

while (reader.Read())

{

spisok.Add(new NotesGroups(reader.GetInt32(0), reader.GetString(1),

new Uri(@"C:\Users\I'm\Desktop\PerfumeSalon\PerfumeSalon" + reader.GetString(2), UriKind.Relative), reader.GetInt32(3)));

//spisok.Add(new NotesGroups(reader.GetInt32(0), reader.GetString(1),

// new Uri(@"C:\Users\stud\Desktop\PerfumeSalon\PerfumeSalon" + reader.GetString(2), UriKind.Relative), reader.GetInt32(3)));

//spisok.Add(new NotesGroups(reader.GetInt32(0), reader.GetString(1),

// new Uri(reader.GetString(2), UriKind.Relative), reader.GetInt32(3)));

}

connect.Close();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message);

}

return spisok;

}

}

}

Приложение Б №22. Код класса dbOrders.cs

using System;

using System.Collections.Generic;

using System.Data.SqlClient;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using PerfumeSalon.Models;

namespace PerfumeSalon.Classes

{

class dbOrders

{

public static void CreateOrder(int userId, DateTime date, string address)

{

SqlConnection connect = new SqlConnection(Connection.connectionString);

SqlCommand command = new SqlCommand("exec CreateOrder @userId, @date, @address", connect);

command.Parameters.AddWithValue("@userId", userId);

command.Parameters.AddWithValue("@date", date);

command.Parameters.AddWithValue("@address", address);

try

{

connect.Open();

command.ExecuteNonQuery();

connect.Close();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message);

}

}

public static int GetLastOrderId(int userId, string address)

{

SqlConnection connect = new SqlConnection(Connection.connectionString);

SqlCommand command = new SqlCommand("select Top(1) O.id from Orders O" +

" join Customers C on C.id = O.Customer" +

" join Users U on U.id = C.Login" +

" where U.id = @userId" +

" and O.Address = @address" +

" group by O.id" +

" order by O.id desc", connect);

command.Parameters.AddWithValue("@userId", userId);

command.Parameters.AddWithValue("@address", address);

int orderId = 0;

try

{

connect.Open();

SqlDataReader reader = command.ExecuteReader();

if (reader.HasRows)

{

while (reader.Read())

{

orderId = reader.GetInt32(0);

}

}

connect.Close();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message);

}

return orderId;

}

public static void CreateOrderInfo(int orderId, int productId, int quantity)

{

SqlConnection connect = new SqlConnection(Connection.connectionString);

SqlCommand command = new SqlCommand("exec CreateOrderInfo @orderId, @productId, @quantity", connect);

command.Parameters.AddWithValue("@orderId", orderId);

command.Parameters.AddWithValue("@productId", productId);

command.Parameters.AddWithValue("@quantity", quantity);

try

{

connect.Open();

command.ExecuteNonQuery();

connect.Close();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message);

}

}

public static List<Orders> ShowOrders(int userId)

{

SqlConnection connect = new SqlConnection(Connection.connectionString);

SqlCommand command = new SqlCommand("select \* from ShowOrders(@userId)", connect);

List<Orders> orders = new List<Orders>();

command.Parameters.AddWithValue("@userId", userId);

try

{

connect.Open();

SqlDataReader reader = command.ExecuteReader();

if (reader.HasRows)

{

while (reader.Read())

{

orders.Add(new Orders(reader.GetInt32(0), reader.GetDateTime(1), reader.GetString(2), reader.GetInt32(3)));

}

}

connect.Close();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message);

}

return orders;

}

public static List<Products> ShowOrderInfo(int orderId)

{

SqlConnection connect = new SqlConnection(Connection.connectionString);

SqlCommand command = new SqlCommand("select \* from ShowOrderInfo(@orderId)", connect);

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

command.Parameters.AddWithValue("@orderId", orderId);

try

{

connect.Open();

SqlDataReader reader = command.ExecuteReader();

if (reader.HasRows)

{

while (reader.Read())

{

products.Add(new Products(reader.GetInt32(0), reader.GetString(1),

reader.GetString(2), reader.GetString(3), reader.GetInt32(4),

reader.GetInt32(5), reader.GetInt32(6)));

}

}

connect.Close();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message);

}

return products;

}

}

}

Приложение Б №23. Код класса dbProducts.cs

using System;

using System.Collections.Generic;

using System.Data.SqlClient;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using PerfumeSalon.Models;

namespace PerfumeSalon.Classes

{

class dbProducts

{

public static List<Products> LoadProductsByAromaId(int id)

{

SqlConnection connect = new SqlConnection(Connection.connectionString);

SqlCommand command = new SqlCommand("select P.[id],[Type],[Category],A.[Aroma],V.[Volume],[Price]" +

" from Products P join Aromas A on A.id = P.Aroma" +

" join ProductsTypes PT on PT.id = P.ProductType" +

" join ProductsCategories PC on PC.id = P.ProductCategory" +

" join Volumes V on V.id = P.Volume" +

" where A.id = @id", connect);

List<Products> spisok = new List<Products>();

command.Parameters.AddWithValue("@id", id);

try

{

connect.Open();

SqlDataReader reader = command.ExecuteReader();

while (reader.Read())

{

spisok.Add(new Products(reader.GetInt32(0), reader.GetString(1),

reader.GetString(2), reader.GetString(3), reader.GetInt32(4), reader.GetInt32(5)));

}

connect.Close();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message);

}

return spisok;

}

public static List<Volumes> LoadVolumesByAromaId(int id)

{

SqlConnection connect = new SqlConnection(Connection.connectionString);

SqlCommand command = new SqlCommand("select V.[id],V.[Volume] from Volumes V" +

" join Products P on P.Volume = V.id" +

" join Aromas A on A.id = P.Aroma" +

" where A.id = @id", connect);

List<Volumes> spisok = new List<Volumes>();

command.Parameters.AddWithValue("@id", id);

try

{

connect.Open();

SqlDataReader reader = command.ExecuteReader();

while (reader.Read())

{

spisok.Add(new Volumes(reader.GetInt32(0), reader.GetInt32(1)));

}

connect.Close();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message);

}

return spisok;

}

public static List<NotesInAromas> LoadNotesByAromaId(int id)

{

SqlConnection connect = new SqlConnection(Connection.connectionString);

SqlCommand command = new SqlCommand("select NC.id, N.Note from [dbo].[NotesInAromas] NA" +

" join[dbo].[Aromas] A on A.id = NA.Aroma" +

" join[dbo].[NotesClasses] NC on NC.id = NA.NoteClass" +

" join Notes N on N.id = NA.Note" +

" where A.id = @id", connect);

List<NotesInAromas> spisok = new List<NotesInAromas>();

command.Parameters.AddWithValue("@id", id);

try

{

connect.Open();

SqlDataReader reader = command.ExecuteReader();

while (reader.Read())

{

spisok.Add(new NotesInAromas(reader.GetInt32(0), reader.GetString(1)));

}

connect.Close();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message);

}

return spisok;

}

public static int GetPriceByVolume(int id, int volume)

{

SqlConnection connect = new SqlConnection(Connection.connectionString);

SqlCommand command = new SqlCommand("select Price from Products P" +

" join Volumes V on V.id = P.Volume" +

" where Aroma = @id" +

" and V.Volume = @volume", connect);

int price = 0;

command.Parameters.AddWithValue("@id", id);

command.Parameters.AddWithValue("@volume", volume);

try

{

connect.Open();

SqlDataReader reader = command.ExecuteReader();

if (reader.HasRows)

{

while (reader.Read())

{

price = reader.GetInt32(0);

}

}

connect.Close();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message);

}

return price;

}

public static string GetDescriptionByAromaId(int id)

{

SqlConnection connect = new SqlConnection(Connection.connectionString);

SqlCommand command = new SqlCommand("select Description from Aromas" +

" where id = @id", connect);

string description = "";

command.Parameters.AddWithValue("@id", id);

try

{

connect.Open();

SqlDataReader reader = command.ExecuteReader();

if (reader.HasRows)

{

while (reader.Read())

{

description = reader.GetString(0);

}

}

connect.Close();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message);

}

return description;

}

public static List<int> GetProductIdFromFavorites(int id)

{

SqlConnection connect = new SqlConnection(Connection.connectionString);

SqlCommand command = new SqlCommand("select \* from GetProductIdFromFavorites(@id)", connect);

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

command.Parameters.AddWithValue("@id", id);

try

{

connect.Open();

SqlDataReader reader = command.ExecuteReader();

if (reader.HasRows)

{

while (reader.Read())

{

products.Add(reader.GetInt32(0));

}

}

connect.Close();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message);

}

return products;

}

public static List<string> GetAromaFromFavorites(int id)

{

SqlConnection connect = new SqlConnection(Connection.connectionString);

SqlCommand command = new SqlCommand("select \* from GetAromaFromFavorites(@id)", connect);

List<string> aromas = new List<string>();

command.Parameters.AddWithValue("@id", id);

try

{

connect.Open();

SqlDataReader reader = command.ExecuteReader();

if (reader.HasRows)

{

while (reader.Read())

{

aromas.Add(reader.GetString(0));

}

}

connect.Close();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message);

}

return aromas;

}

public static void ChangeMark(int userId, string aroma)

{

SqlConnection connect = new SqlConnection(Connection.connectionString);

SqlCommand command = new SqlCommand("exec MarkAsFavorite @userId, @aroma", connect);

command.Parameters.AddWithValue("@userId", userId);

command.Parameters.AddWithValue("@aroma", aroma);

try

{

connect.Open();

command.ExecuteNonQuery();

connect.Close();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message);

}

}

public static List<string> GetProductsTypes()

{

SqlConnection connect = new SqlConnection(Connection.connectionString);

SqlCommand command = new SqlCommand("select Type from ProductsTypes", connect);

List<string> types = new List<string>();

try

{

connect.Open();

SqlDataReader reader = command.ExecuteReader();

if (reader.HasRows)

{

while (reader.Read())

{

types.Add(reader.GetString(0));

}

}

connect.Close();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message);

}

return types;

}

}

}

Приложение Б №24. Код класса dbUsers.cs

using System;

using System.Collections.Generic;

using System.Data.SqlClient;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using PerfumeSalon.Models;

namespace PerfumeSalon.Classes

{

public class dbUsers

{

public static void RegisterNewUser(string login, string password, string f, string m, string l, string email, string phone)

{

SqlConnection connect = new SqlConnection(Connection.connectionString);

SqlCommand command = new SqlCommand("select \* from IsLoginTaken(@login)", connect);

command.Parameters.AddWithValue("@login", login);

try

{

connect.Open();

SqlDataReader reader = command.ExecuteReader();

if (!reader.HasRows)

{

reader.Close();

SqlCommand commandEmail = new SqlCommand("select \* from IsEmailTaken(@email)", connect);

commandEmail.Parameters.AddWithValue("@email",email);

SqlDataReader readerEmail = commandEmail.ExecuteReader();

if (!readerEmail.HasRows)

{

readerEmail.Close();

SqlCommand commandPhone = new SqlCommand("select \* from IsPhoneTaken(@phone)", connect);

commandPhone.Parameters.AddWithValue("@phone", phone);

SqlDataReader readerPhone = commandPhone.ExecuteReader();

if (!readerPhone.HasRows)

{

readerPhone.Close();

SqlCommand command1 = new SqlCommand("exec RegisterNewUser @login, @password," +

" @f, @m, @l, @email, @phone", connect);

command1.Parameters.AddWithValue("@login", login);

command1.Parameters.AddWithValue("@password", password);

command1.Parameters.AddWithValue("@f", f);

command1.Parameters.AddWithValue("@m", m);

command1.Parameters.AddWithValue("@l", l);

command1.Parameters.AddWithValue("@email", email);

command1.Parameters.AddWithValue("@phone", phone);

command1.ExecuteNonQuery();

MessageBox.Show("Регистрация прошла успешно");

}

else

{

MessageBox.Show("Этот номер телефона занят");

}

}

else

{

MessageBox.Show("Этот адрес эл. почты занят");

}

}

else

{

MessageBox.Show("Этот логин занят");

}

connect.Close();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message);

}

}

public static Users Login(string login, string password)

{

SqlConnection connect = new SqlConnection(Connection.connectionString);

SqlCommand command = new SqlCommand("select \* from Users" +

" where Login = @login", connect);

Users user = new Users();

command.Parameters.AddWithValue("@login", login);

try

{

connect.Open();

SqlDataReader reader = command.ExecuteReader();

if (reader.HasRows)

{

reader.Close();

SqlCommand command1 = new SqlCommand("select \* from Users" +

" where Login = @login and Password = @password", connect);

command1.Parameters.AddWithValue("@login", login);

command1.Parameters.AddWithValue("@password", password);

SqlDataReader reader1 = command1.ExecuteReader();

if (reader1.HasRows)

{

while (reader1.Read())

{

user = new Users(reader1.GetInt32(0), reader1.GetString(1), reader1.GetString(2), reader1.GetInt32(3));

}

MessageBox.Show("Авторизация прошла успешно");

}

else

{

MessageBox.Show("Неверный пароль");

}

}

else

{

MessageBox.Show("Неверный логин");

}

connect.Close();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message);

}

return user;

}

public static Profiles GetProfileByUserId(int userId)

{

SqlConnection connect = new SqlConnection(Connection.connectionString);

SqlCommand command = new SqlCommand("select \* from GetProfileByUserId(@userId)", connect);

Profiles profile = new Profiles();

command.Parameters.AddWithValue("@userId", userId);

try

{

connect.Open();

SqlDataReader reader = command.ExecuteReader();

while (reader.Read())

{

profile = new Profiles(reader.GetString(0), reader.GetString(1),

reader.GetInt32(2), reader.GetString(3), reader.GetString(4),

reader.GetString(5), reader.GetString(6), reader.GetString(7));

}

connect.Close();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message);

}

return profile;

}

public static void EditProfile(int id, string login, string password, string f, string m,

string l, string email, string phone)

{

SqlConnection connect = new SqlConnection(Connection.connectionString);

SqlCommand command = new SqlCommand("exec EditUser @id, @login, @password," +

" @f, @m, @l, @email, @phone"

, connect);

command.Parameters.AddWithValue("@id", id);

command.Parameters.AddWithValue("@login", login);

command.Parameters.AddWithValue("@password", password);

command.Parameters.AddWithValue("@f", f);

command.Parameters.AddWithValue("@m", m);

command.Parameters.AddWithValue("@l", l);

command.Parameters.AddWithValue("@email", email);

command.Parameters.AddWithValue("@phone", phone);

try

{

connect.Open();

command.ExecuteNonQuery();

MessageBox.Show("Данные изменены");

connect.Close();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message);

}

}

public static void EditProfileWithoutPassword(int id, string f, string m,

string l, string email, string phone)

{

SqlConnection connect = new SqlConnection(Connection.connectionString);

SqlCommand command = new SqlCommand("exec EditUserWithoutPassword @id," +

" @f, @m, @l, @email, @phone"

, connect);

command.Parameters.AddWithValue("@id", id);

command.Parameters.AddWithValue("@f", f);

command.Parameters.AddWithValue("@m", m);

command.Parameters.AddWithValue("@l", l);

command.Parameters.AddWithValue("@email", email);

command.Parameters.AddWithValue("@phone", phone);

try

{

connect.Open();

command.ExecuteNonQuery();

MessageBox.Show("Данные изменены");

connect.Close();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message);

}

}

}

}

Приложение Б №25. Код класса AromaInfoPage.xaml.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Navigation;

using System.Windows.Shapes;

using PerfumeSalon.Classes;

namespace PerfumeSalon.Pages

{

/// <summary>

/// Логика взаимодействия для AromaInfo.xaml

/// </summary>

///

/\*

<Button.Template>

<ControlTemplate>

<Image x:Name="imgFavorite" Source="/Resources/Favorite.png" Height="70" Width="70"/>

</ControlTemplate>

</Button.Template>

\*/

public partial class AromaInfo : Page

{

public static bool markedAsFavorite = false;

public int id;

public List<Products> products = new List<Products>();

public List<NotesInAromas> notes = new List<NotesInAromas>();

public List<Volumes> volumes = new List<Volumes>();

public List<string> aromasGroups = new List<string>();

public List<string> favoriteAromas = new List<string>();

public static string description;

public static int selectedVolume = 0;

public AromaInfo(int id)

{

InitializeComponent();

this.id = id;

}

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

{

if (UserId.userId == 0)

{

FrameClass.BodyFrame.Navigate(new Pages.LoginPage());

}

else

{

markedAsFavorite = !markedAsFavorite;

if (markedAsFavorite)

{

Uri uri = new Uri("/Resources/FavoriteMarked.png", UriKind.Relative);

BitmapImage bitmap = new BitmapImage(uri);

Image image = new Image();

image.Source = bitmap;

btnFavorite.Content = image;

}

else

{

Uri uri = new Uri("/Resources/Favorite.png", UriKind.Relative);

BitmapImage bitmap = new BitmapImage(uri);

Image image = new Image();

image.Source = bitmap;

btnFavorite.Content = image;

}

dbProducts.ChangeMark(UserId.userId, products[0].Aroma);

}

}

public void LoadImageToButton()

{

Uri uri = new Uri("/Resources/Favorite.png", UriKind.Relative);

BitmapImage bitmap = new BitmapImage(uri);

Image image = new Image();

image.Source = bitmap;

btnFavorite.Content = image;

}

private void Page\_Loaded(object sender, RoutedEventArgs e)

{

LoadImageToButton();

btnVolume5.Visibility = Visibility.Collapsed;

btnVolume15.Visibility = Visibility.Collapsed;

btnVolume30.Visibility = Visibility.Collapsed;

btnVolume50.Visibility = Visibility.Collapsed;

btnVolume100.Visibility = Visibility.Collapsed;

products = dbProducts.LoadProductsByAromaId(id);

tbAroma.Text = products[0].Aroma;

tbProductType.Text = products[0].Type;

tbCategory.Text = products[0].Category;

description = dbProducts.GetDescriptionByAromaId(id);

int minVolume = products[0].Volume;

selectedVolume = minVolume;

aromasGroups = dbAromasGroups.GetAromasGroupsByAromaId(id);

volumes = dbProducts.LoadVolumesByAromaId(id);

notes = dbProducts.LoadNotesByAromaId(id);

foreach (string group in aromasGroups)

{

tbAromasGroups.Text += $"{group}, ";

}

foreach (Volumes volume in volumes)

{

switch (volume.Volume)

{

case 5: btnVolume5.Visibility = Visibility.Visible; break;

case 15: btnVolume15.Visibility = Visibility.Visible; break;

case 30: btnVolume30.Visibility = Visibility.Visible; break;

case 50: btnVolume50.Visibility = Visibility.Visible; break;

case 100: btnVolume100.Visibility = Visibility.Visible; break;

}

}

foreach (NotesInAromas note in notes)

{

switch (note.NoteClass)

{

case 1:

tbTopNotes.Text += $"{note.Note}, ";

break;

case 2:

tbMidNotes.Text += $"{note.Note}, ";

break;

case 3:

tbBaseNotes.Text += $"{note.Note}, ";

break;

}

}

tbAromasGroups.Text = tbAromasGroups.Text.Remove(tbAromasGroups.Text.Length - 2);

tbTopNotes.Text = tbTopNotes.Text.Remove(tbTopNotes.Text.Length - 2);

tbMidNotes.Text = tbMidNotes.Text.Remove(tbMidNotes.Text.Length - 2);

tbBaseNotes.Text = tbBaseNotes.Text.Remove(tbBaseNotes.Text.Length - 2);

if (description.Length > 1)

{

tbDescription.Visibility = Visibility.Visible;

tbDescription.Text = description;

}

if (UserId.userId != 0)

{

favoriteAromas = dbProducts.GetAromaFromFavorites(UserId.userId);

if (favoriteAromas.Count > 0)

{

foreach (string a in favoriteAromas)

{

if (a == tbAroma.Text)

{

Uri uri = new Uri("/Resources/FavoriteMarked.png", UriKind.Relative);

BitmapImage bitmap = new BitmapImage(uri);

Image image = new Image();

image.Source = bitmap;

btnFavorite.Content = image;

markedAsFavorite = true;

}

}

}

}

SelectedVolume(minVolume);

}

private void btnVolume5\_Click(object sender, RoutedEventArgs e)

{

SelectedVolume(5);

}

private void btnVolume15\_Click(object sender, RoutedEventArgs e)

{

SelectedVolume(15);

}

private void btnVolume30\_Click(object sender, RoutedEventArgs e)

{

SelectedVolume(30);

}

private void btnVolume50\_Click(object sender, RoutedEventArgs e)

{

SelectedVolume(50);

}

private void btnVolume100\_Click(object sender, RoutedEventArgs e)

{

SelectedVolume(100);

}

private void SelectedVolume(int vol)

{

//#FF5F5F5F

//Brush grey = new Brush();

btnVolume5.Foreground = Brushes.LightGray;

btnVolume5.BorderBrush = Brushes.LightGray;

btnVolume15.Foreground = Brushes.LightGray;

btnVolume15.BorderBrush = Brushes.LightGray;

btnVolume30.Foreground = Brushes.LightGray;

btnVolume30.BorderBrush = Brushes.LightGray;

btnVolume50.Foreground = Brushes.LightGray;

btnVolume50.BorderBrush = Brushes.LightGray;

btnVolume100.Foreground = Brushes.LightGray;

btnVolume100.BorderBrush = Brushes.LightGray;

Uri uri;

BitmapImage bitmap;

switch (vol)

{

case 5:

selectedVolume = 5;

uri = new Uri("/Resources/Volumes/5.jpg", UriKind.Relative);

bitmap = new BitmapImage(uri);

imgAromaPic.Source = bitmap;

btnVolume5.Foreground = Brushes.Black;

btnVolume5.BorderBrush = Brushes.Black;

tbPrice.Text = dbProducts.GetPriceByVolume(id,5).ToString();

break;

case 15:

selectedVolume = 15;

uri = new Uri("/Resources/Volumes/15.jpg", UriKind.Relative);

bitmap = new BitmapImage(uri);

imgAromaPic.Source = bitmap;

btnVolume15.Foreground = Brushes.Black;

btnVolume15.BorderBrush = Brushes.Black;

tbPrice.Text = dbProducts.GetPriceByVolume(id, 15).ToString();

break;

case 30:

selectedVolume = 30;

uri = new Uri("/Resources/Volumes/30.jpg", UriKind.Relative);

bitmap = new BitmapImage(uri);

imgAromaPic.Source = bitmap;

btnVolume30.Foreground = Brushes.Black;

btnVolume30.BorderBrush = Brushes.Black;

tbPrice.Text = dbProducts.GetPriceByVolume(id, 30).ToString();

break;

case 50:

selectedVolume = 50;

uri = new Uri("/Resources/Volumes/50.jpg", UriKind.Relative);

bitmap = new BitmapImage(uri);

imgAromaPic.Source = bitmap;

btnVolume50.Foreground = Brushes.Black;

btnVolume50.BorderBrush = Brushes.Black;

tbPrice.Text = dbProducts.GetPriceByVolume(id, 50).ToString();

break;

case 100:

selectedVolume = 100;

uri = new Uri("/Resources/Volumes/100.jpg", UriKind.Relative);

bitmap = new BitmapImage(uri);

imgAromaPic.Source = bitmap;

btnVolume100.Foreground = Brushes.Black;

btnVolume100.BorderBrush = Brushes.Black;

tbPrice.Text = dbProducts.GetPriceByVolume(id, 100).ToString();

break;

}

}

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

{

if (UserId.userId == 0)

{

FrameClass.BodyFrame.Navigate(new Pages.LoginPage());

}

else

{

int productId = 0;

foreach (Products p in products)

{

if(p.Volume == selectedVolume) { productId = p.Id; }

}

dbBaskets.AddProductToBasket(UserId.userId, productId);

}

}

}

}

Приложение Б №26. Код класса AromaPickPage.xaml.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Navigation;

using System.Windows.Shapes;

using PerfumeSalon.Classes;

namespace PerfumeSalon.Pages

{

/// <summary>

/// Логика взаимодействия для AromaPick.xaml

/// </summary>

public partial class AromaPickPage : Page

{

public int id;

public static List<Aromas> aromas = new List<Aromas>();

public AromaPickPage(int id = 0)

{

InitializeComponent();

this.id = id;

}

private void Page\_Loaded(object sender, RoutedEventArgs e)

{

aromas = dbAromas.LoadFromDB(id);

ListViewAromas.ItemsSource = null;

ListViewAromas.Items.Clear();

ListViewAromas.ItemsSource = aromas;

}

private void ListViewAromas\_SelectionChanged(object sender, SelectionChangedEventArgs e)

{

if (ListViewAromas.Items.Count > 0)

{

int index = ListViewAromas.SelectedIndex;

int id = aromas[index].Id;

FrameClass.BodyFrame.Navigate(new Pages.AromaInfo(id));

}

}

}

}

Приложение Б №27. Код класса AromasGroupPage.xaml.cs

using PerfumeSalon.Classes;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Security.Policy;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Navigation;

using System.Windows.Shapes;

namespace PerfumeSalon.Pages

{

/// <summary>

/// Логика взаимодействия для AromasGroupPage.xaml

/// </summary>

public partial class AromasGroupPage : Page

{

List<AromasGroups> aromasGroups = new List<AromasGroups>();

public AromasGroupPage()

{

InitializeComponent();

aromasGroups = dbAromasGroups.LoadFromDB();

ListViewGroups.Items.Clear();

ListViewGroups.ItemsSource = aromasGroups;

}

private void ListViewGroups\_SelectionChanged(object sender, SelectionChangedEventArgs e)

{

int index = ListViewGroups.SelectedIndex;

//string group = aromasGroups[index].AromasGroup;

//MessageBox.Show(group);

int id = aromasGroups[index].Id;

FrameClass.BodyFrame.Navigate(new Pages.AromaPickPage(id));

}

}

}

Приложение Б №28. Код класса BasketsPage.xaml.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Navigation;

using System.Windows.Shapes;

using PerfumeSalon.Classes;

namespace PerfumeSalon.Pages

{

/// <summary>

/// Логика взаимодействия для BasketPage.xaml

/// </summary>

public partial class BasketPage : Page

{

public static List<Products> products = new List<Products>();

public static ListView lvPlaceholder = new ListView();

public BasketPage()

{

InitializeComponent();

DataContext = new cmdDelete();

}

private void Page\_Loaded(object sender, RoutedEventArgs e)

{

LoadProducts();

if (products.Count > 0)

{

gridEmpty.Visibility = Visibility.Collapsed;

}

else

{

gridEmpty.Visibility = Visibility.Visible;

gridList.Visibility = Visibility.Collapsed;

}

}

private void ListViewAromas\_SelectionChanged(object sender, SelectionChangedEventArgs e)

{

if (ListViewAromas.Items.Count > 0)

{

int index = ListViewAromas.SelectedIndex;

int id = products[index].Id;

int aromaId = dbBaskets.GetAromaIdFromProduct(id);

FrameClass.BodyFrame.Navigate(new Pages.AromaInfo(aromaId));

}

}

public void LoadProducts()

{

products = dbBaskets.GetProductsFromBasket(UserId.userId);

//UpdateProducts();

ListViewAromas.ItemsSource = null;

ListViewAromas.Items.Clear();

ListViewAromas.ItemsSource = products;

//ListViewProducts.lvPlaceholder = ListViewAromas;

}

public static void UpdateProducts()

{

ListViewProducts.lvPlaceholder.ItemsSource = null;

ListViewProducts.lvPlaceholder.Items.Clear();

ListViewProducts.lvPlaceholder.ItemsSource = products;

}

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

{

LoadProducts();

}

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

{

if (ListViewAromas.Items.Count > 0)

{

FrameClass.BodyFrame.Navigate(new Pages.OrderPage(products));

}

else

{

MessageBox.Show("Корзина пуста");

}

}

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

{

FrameClass.BodyFrame.Navigate(new Pages.UsersOrdersPage());

}

}

}

Приложение Б №29. Код класса ConstructorPage.xaml.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Navigation;

using System.Windows.Shapes;

namespace PerfumeSalon.Pages

{

/// <summary>

/// Логика взаимодействия для ConstructorPage.xaml

/// </summary>

public partial class ConstructorPage : Page

{

public ConstructorPage()

{

InitializeComponent();

}

}

}

Приложение Б №30. Код класса FavoriteAromasPage.xaml.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Navigation;

using System.Windows.Shapes;

using PerfumeSalon.Classes;

namespace PerfumeSalon.Pages

{

/// <summary>

/// Логика взаимодействия для FavoriteProductsPage.xaml

/// </summary>

public partial class FavoriteAromasPage : Page

{

public static List<Aromas> aromas = new List<Aromas>();

public FavoriteAromasPage()

{

InitializeComponent();

}

private void Page\_Loaded(object sender, RoutedEventArgs e)

{

aromas = dbAromas.GetFavoriteAromas(UserId.userId);

ListViewAromas.ItemsSource = null;

ListViewAromas.Items.Clear();

if (aromas.Count > 0)

{

ListViewAromas.ItemsSource = aromas;

gridEmpty.Visibility = Visibility.Collapsed;

gridList.Margin = new Thickness(0,0,0,0);

}

else

{

gridEmpty.Visibility = Visibility.Visible;

gridList.Margin = new Thickness(0, 40, 0, 0);

ListViewAromas.Visibility = Visibility.Collapsed;

}

}

private void ListViewAromas\_SelectionChanged(object sender, SelectionChangedEventArgs e)

{

if (ListViewAromas.Items.Count > 0)

{

int index = ListViewAromas.SelectedIndex;

int id = aromas[index].Id;

FrameClass.BodyFrame.Navigate(new Pages.AromaInfo(id));

}

}

}

}

Приложение Б №31. Код класса LoginPage.xaml.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Navigation;

using System.Windows.Shapes;

using PerfumeSalon.Classes;

namespace PerfumeSalon.Pages

{

/// <summary>

/// Логика взаимодействия для LoginPage.xaml

/// </summary>

public partial class LoginPage : Page

{

Users user = new Users();

public LoginPage()

{

InitializeComponent();

}

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

{

FrameClass.BodyFrame.Navigate(new Pages.RegistrationPage());

}

private void Page\_KeyDown(object sender, KeyEventArgs e)

{

if (e.Key == Key.Enter)

{

btnLogin\_Click(sender, e);

}

}

public void btnLogin\_Click(object sender, RoutedEventArgs e)

{

user = dbUsers.Login(tbxLogin.Text, tbxPassword.Text);

UserId.userId = user.Id;

UserId.roleId = user.Role;

if (UserId.userId != 0)

{

FrameClass.BodyFrame.Navigate(new Pages.UserProfilePage());

}

}

}

}

Приложение Б №32. Код класса NotePickPage.xaml.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Navigation;

using System.Windows.Shapes;

using PerfumeSalon.Classes;

namespace PerfumeSalon.Pages

{

/// <summary>

/// Логика взаимодействия для NotePick.xaml

/// </summary>

public partial class NotePickPage : Page

{

public int id;

public List<Notes> notes = new List<Notes>();

public NotePickPage(int id)

{

InitializeComponent();

this.id = id;

}

private void Page\_Loaded(object sender, RoutedEventArgs e)

{

notes = dbNotes.LoadFromDB(id);

ListViewNotes.Items.Clear();

ListViewNotes.ItemsSource = notes;

}

private void ListViewNotes\_SelectionChanged(object sender, SelectionChangedEventArgs e)

{

}

}

}

Приложение Б №33. Код класса NoteGroupPage.xaml.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Navigation;

using System.Windows.Shapes;

using PerfumeSalon.Classes;

namespace PerfumeSalon.Pages

{

/// <summary>

/// Логика взаимодействия для NotesGroupPage.xaml

/// </summary>

public partial class NotesGroupPage : Page

{

List<NotesGroups> notesGroups = new List<NotesGroups>();

public NotesGroupPage()

{

InitializeComponent();

notesGroups = dbNotesGroups.LoadFromDB();

ListViewGroups.Items.Clear();

ListViewGroups.ItemsSource = notesGroups;

}

private void ListViewGroups\_SelectionChanged(object sender, SelectionChangedEventArgs e)

{

int index = ListViewGroups.SelectedIndex;

//string group = aromasGroups[index].AromasGroup;

//MessageBox.Show(group);

int id = notesGroups[index].Id;

FrameClass.BodyFrame.Navigate(new Pages.NotePickPage(id));

}

}

}

Приложение Б №34. Код класса OrderInfoPage.xaml.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Navigation;

using System.Windows.Shapes;

using PerfumeSalon.Classes;

namespace PerfumeSalon.Pages

{

/// <summary>

/// Логика взаимодействия для OrderInfoPage.xaml

/// </summary>

public partial class OrderInfoPage : Page

{

public List<Products> products = new List<Products>();

public int orderId = 0;

public OrderInfoPage(int orderId)

{

InitializeComponent();

this.orderId = orderId;

}

private void ListViewAromas\_SelectionChanged(object sender, SelectionChangedEventArgs e)

{

if (ListViewAromas.Items.Count > 0)

{

int index = ListViewAromas.SelectedIndex;

int id = products[index].Id;

int aromaId = dbBaskets.GetAromaIdFromProduct(id);

FrameClass.BodyFrame.Navigate(new Pages.AromaInfo(aromaId));

}

}

private void Page\_Loaded(object sender, RoutedEventArgs e)

{

products = dbOrders.ShowOrderInfo(orderId);

ListViewAromas.ItemsSource = null;

ListViewAromas.Items.Clear();

ListViewAromas.ItemsSource = products;

}

}

}

Приложение Б №35. Код класса OrderPage.xaml.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Navigation;

using System.Windows.Shapes;

using PerfumeSalon.Classes;

namespace PerfumeSalon.Pages

{

/// <summary>

/// Логика взаимодействия для OrderPage.xaml

/// </summary>

public partial class OrderPage : Page

{

public List<Products> products = new List<Products>();

public OrderPage(List<Products> products)

{

InitializeComponent();

this.products = products;

}

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

{

FrameClass.BodyFrame.Navigate(new Pages.BasketPage());

}

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

{

dbOrders.CreateOrder(UserId.userId, DateTime.Now, tbxAddress.Text);

int orderId = dbOrders.GetLastOrderId(UserId.userId, tbxAddress.Text);

foreach (Products p in products)

{

dbOrders.CreateOrderInfo(orderId, p.Id, p.Quantity);

}

MessageBox.Show("Заказ создан!");

FrameClass.BodyFrame.Navigate(new Pages.BasketPage());

}

}

}

Приложение Б №36. Код класса PerfumePickPage.xaml.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Navigation;

using System.Windows.Shapes;

using PerfumeSalon.Classes;

namespace PerfumeSalon.Pages

{

/// <summary>

/// Логика взаимодействия для PerfumePickPage.xaml

/// </summary>

public partial class PerfumePickPage : Page

{

public static List<Aromas> aromas = new List<Aromas>();

public static List<AromasGroups> aromasGroups = new List<AromasGroups>();

public static List<string> productsTypes = new List<string>();

public PerfumePickPage()

{

InitializeComponent();

}

private void Page\_Loaded(object sender, RoutedEventArgs e)

{

cbSortList.Items.Clear();

cbAromasGroups.Items.Clear();

cbProductType.Items.Clear();

aromasGroups = dbAromasGroups.LoadFromDB();

foreach (AromasGroups ag in aromasGroups)

{

cbAromasGroups.Items.Add(ag.AromasGroup);

}

productsTypes = dbProducts.GetProductsTypes();

foreach (string pt in productsTypes)

{

cbProductType.Items.Add(pt);

}

cbAromasGroups.Items.Add("");

cbProductType.Items.Add("");

cbAromasGroups.Text = "";

cbProductType.Text = "";

cbSortList.Items.Add("по названию");

cbSortList.Items.Add("по возрастанию цены");

cbSortList.Items.Add("по убыванию цены");

cbSortList.SelectedIndex = 0;

cbAromasGroups.SelectedIndex = cbAromasGroups.Items.Count-1;

cbProductType.SelectedIndex = cbProductType.Items.Count - 1;

UpdateList(0,"%","%","%");

}

public void UpdateList(int index, string aromasGroup, string type, string search)

{

aromas = dbAromas.SearchAromas(index, aromasGroup, type, search);

//cbSortList.SelectedIndex, cbAromasGroups.Text,

//cbProductType.Text, tbxSearch.Text

ListViewAromas.ItemsSource = null;

ListViewAromas.Items.Clear();

ListViewAromas.ItemsSource = aromas;

}

private void ListViewAromas\_SelectionChanged(object sender, SelectionChangedEventArgs e)

{

if (ListViewAromas.Items.Count > 0)

{

int index = ListViewAromas.SelectedIndex;

int id = aromas[index].Id;

FrameClass.BodyFrame.Navigate(new Pages.AromaInfo(id));

}

}

private void tbxSearch\_TextChanged(object sender, TextChangedEventArgs e)

{

UpdateList(cbSortList.SelectedIndex, cbAromasGroups.Text,

cbProductType.Text, tbxSearch.Text);

}

private void cbSortList\_SelectionChanged(object sender, SelectionChangedEventArgs e)

{

if (ListViewAromas.Items.Count > 0 && cbSortList.Items.Count > 0)

{

int selectedComboBoxIndex = cbSortList.SelectedIndex;

UpdateList(selectedComboBoxIndex, cbAromasGroups.Text,

cbProductType.Text, tbxSearch.Text);

}

}

private void cbAromasGroups\_SelectionChanged(object sender, SelectionChangedEventArgs e)

{

if (ListViewAromas.Items.Count > 0 && cbAromasGroups.Items.Count > 0)

{

string selectedComboBoxItem = cbAromasGroups.SelectedItem.ToString();

UpdateList(cbSortList.SelectedIndex, selectedComboBoxItem.ToString(),

cbProductType.Text, tbxSearch.Text);

}

}

private void cbProductType\_SelectionChanged(object sender, SelectionChangedEventArgs e)

{

if (ListViewAromas.Items.Count > 0 && cbProductType.Items.Count > 0)

{

string selectedComboBoxItem = cbProductType.SelectedItem.ToString();

UpdateList(cbSortList.SelectedIndex, cbAromasGroups.Text,

selectedComboBoxItem.ToString(), tbxSearch.Text);

}

}

}

}

Приложение Б №37. Код класса RegistrationPage.xaml.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Navigation;

using System.Windows.Shapes;

using PerfumeSalon.Classes;

namespace PerfumeSalon.Pages

{

/// <summary>

/// Логика взаимодействия для RegistrationPAge.xaml

/// </summary>

public partial class RegistrationPage : Page

{

public RegistrationPage()

{

InitializeComponent();

}

public void btnRegistration\_Click(object sender, RoutedEventArgs e)

{

if (tbxPassword.Text == tbxPasswordConfirm.Text)

{

if (tbxLogin.Text == "" ||

tbxPassword.Text == "" ||

tbxPasswordConfirm.Text == "" ||

tbxFName.Text == "" ||

tbxMName.Text == "" ||

tbxLName.Text == "" ||

tbxEmail.Text == "" ||

tbxPhone.Text == "")

{

MessageBox.Show("Заполните все поля");

}

else

{

dbUsers.RegisterNewUser(tbxLogin.Text, tbxPassword.Text, tbxFName.Text, tbxMName.Text,

tbxLName.Text, tbxEmail.Text, tbxPhone.Text);

}

}

else

{

MessageBox.Show("Ошибка подтверждения пароля");

}

}

private void Page\_KeyDown(object sender, KeyEventArgs e)

{

if (e.Key == Key.Enter)

{

btnRegistration\_Click(sender, e);

}

}

}

}

Приложение Б №38. Код класса UserProfilePage.xaml.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Navigation;

using System.Windows.Shapes;

using PerfumeSalon.Classes;

namespace PerfumeSalon.Pages

{

/// <summary>

/// Логика взаимодействия для UserProfilePage.xaml

/// </summary>

public partial class UserProfilePage : Page

{

public static Profiles profile = new Profiles();

public UserProfilePage()

{

InitializeComponent();

}

private void Page\_Loaded(object sender, RoutedEventArgs e)

{

tbxLogin.IsEnabled = false;

profile = dbUsers.GetProfileByUserId(UserId.userId);

tbxLogin.Text = profile.Login;

tbxPassword.Text = "●●●●●●●●";

tbxFName.Text = profile.FirstName;

tbxMName.Text = profile.MiddleName;

tbxLName.Text = profile.LastName;

tbxEmail.Text = profile.Email;

tbxPhone.Text = profile.Phone;

}

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

{

if (tbxPassword.Text == "●●●●●●●●")

{

dbUsers.EditProfileWithoutPassword(profile.Id, tbxFName.Text, tbxMName.Text,

tbxLName.Text, tbxEmail.Text, tbxPhone.Text);

}

else

{

dbUsers.EditProfile(profile.Id, profile.Login, tbxPassword.Text,

tbxFName.Text, tbxMName.Text,

tbxLName.Text, tbxEmail.Text, tbxPhone.Text);

}

}

}

}

Приложение Б №39. Код класса UsersOrderPage.xaml.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Navigation;

using System.Windows.Shapes;

using PerfumeSalon.Classes;

namespace PerfumeSalon.Pages

{

/// <summary>

/// Логика взаимодействия для UsersOrdersPage.xaml

/// </summary>

public partial class UsersOrdersPage : Page

{

List<Orders> orders = new List<Orders>();

public UsersOrdersPage()

{

InitializeComponent();

}

private void ListViewOrders\_SelectionChanged(object sender, SelectionChangedEventArgs e)

{

if (ListViewOrders.Items.Count > 0)

{

int index = ListViewOrders.SelectedIndex;

int orderId = orders[index].Id;

FrameClass.BodyFrame.Navigate(new Pages.OrderInfoPage(orderId));

}

}

private void Page\_Loaded(object sender, RoutedEventArgs e)

{

orders = dbOrders.ShowOrders(UserId.userId);

ListViewOrders.ItemsSource = null;

ListViewOrders.Items.Clear();

ListViewOrders.ItemsSource = orders;

}

}

}

Приложение Б №40. Код класса MainWindow.xaml.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Navigation;

using System.Windows.Shapes;

using PerfumeSalon.Classes;

namespace PerfumeSalon

{

/// <summary>

/// Логика взаимодействия для MainWindow.xaml

/// </summary>

public partial class MainWindow : Window

{

public MainWindow()

{

InitializeComponent();

FrameClass.BodyFrame = FrameBody;

//Button btn = new Button();

//btn.Style = btnOpenAromas.Style;

//btnOpenNotes.Style = btn.Style;

//btnOpenConstructor.Style = btn.Style;

//btnOpenPerfumes.Style = btn.Style;

FrameClass.BodyFrame.Navigate(new Pages.PerfumePickPage());

btnOpenConstructor.Visibility = Visibility.Hidden;

}

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

{

//FrameClass.BodyFrame.Navigate(new Pages.AromaPickPage());

//FrameClass.BodyFrame.Navigate(new Pages.AromaInfo());

FrameClass.BodyFrame.Navigate(new Pages.AromasGroupPage());

}

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

{

FrameClass.BodyFrame.Navigate(new Pages.NotesGroupPage());

}

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

{

FrameClass.BodyFrame.Navigate(new Pages.PerfumePickPage());

}

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

{

FrameClass.BodyFrame.Navigate(new Pages.ConstructorPage());

}

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

{

if (UserId.userId == 0)

{

FrameClass.BodyFrame.Navigate(new Pages.LoginPage());

}

else

{

FrameClass.BodyFrame.Navigate(new Pages.UserProfilePage());

}

}

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

{

FrameClass.BodyFrame.Navigate(new Pages.FavoriteAromasPage());

}

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

{

FrameClass.BodyFrame.Navigate(new Pages.BasketPage());

}

}

}

Приложение Б №41. Код App.xaml

<Application x:Class="PerfumeSalon.App"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:local="clr-namespace:PerfumeSalon"

StartupUri="MainWindow.xaml">

<Application.Resources>

<Style TargetType="Button" x:Key="btnMouseOverStyle">

<Setter Property="Cursor" Value="Hand"/>

<Setter Property="Template">

<Setter.Value>

<ControlTemplate TargetType="{x:Type Button}">

<Border Background="{TemplateBinding Background}">

<ContentPresenter HorizontalAlignment="Center" VerticalAlignment="Center"/>

</Border>

</ControlTemplate>

</Setter.Value>

</Setter>

<Style.Triggers>

<Trigger Property="IsMouseOver" Value="True">

<Setter Property="Background" Value="Red"/>

</Trigger>

</Style.Triggers>

</Style>

<Style TargetType="TextBox" x:Key="tbxPlaceholderStyle">

<Setter Property="FontSize" Value="22"/>

<Setter Property="FontFamily" Value="New Courier"/>

<Setter Property="FontWeight" Value="Light"/>

<Setter Property="Foreground" Value="Black"/>

</Style>

<Style TargetType="TextBox" x:Key="tbxPlaceholderLoginStyle" BasedOn="{StaticResource tbxPlaceholderStyle}">

<Setter Property="Template">

<Setter.Value>

<ControlTemplate TargetType="TextBox">

<Grid>

<Border Background="{TemplateBinding Background}"

BorderBrush="{TemplateBinding BorderBrush}"

BorderThickness="{TemplateBinding BorderThickness}" />

<ScrollViewer x:Name="PART\_ContentHost" />

<TextBlock x:Name="PlaceholderText" Text="логин"

Foreground="Gray"

Visibility="Collapsed"

FontSize="22"

FontFamily="New Courier"

FontWeight="Light"

IsHitTestVisible="False"/>

</Grid>

<ControlTemplate.Triggers>

<Trigger Property="Text" Value="">

<Setter TargetName="PlaceholderText" Property="Visibility" Value="Visible"/>

</Trigger>

</ControlTemplate.Triggers>

</ControlTemplate>

</Setter.Value>

</Setter>

</Style>

<Style TargetType="TextBox" x:Key="tbxPlaceholderPasswordStyle" BasedOn="{StaticResource tbxPlaceholderStyle}">

<Setter Property="Template">

<Setter.Value>

<ControlTemplate TargetType="TextBox">

<Grid>

<Border Background="{TemplateBinding Background}"

BorderBrush="{TemplateBinding BorderBrush}"

BorderThickness="{TemplateBinding BorderThickness}" />

<ScrollViewer x:Name="PART\_ContentHost" />

<TextBlock x:Name="PlaceholderText" Text="пароль"

Foreground="Gray"

Visibility="Collapsed"

FontSize="22"

FontFamily="New Courier"

FontWeight="Light"

IsHitTestVisible="False"/>

</Grid>

<ControlTemplate.Triggers>

<Trigger Property="Text" Value="">

<Setter TargetName="PlaceholderText" Property="Visibility" Value="Visible"/>

</Trigger>

</ControlTemplate.Triggers>

</ControlTemplate>

</Setter.Value>

</Setter>

</Style>

<Style TargetType="TextBox" x:Key="tbxPlaceholderPasswordSubmitStyle" BasedOn="{StaticResource tbxPlaceholderStyle}">

<Setter Property="Template">

<Setter.Value>

<ControlTemplate TargetType="TextBox">

<Grid>

<Border Background="{TemplateBinding Background}"

BorderBrush="{TemplateBinding BorderBrush}"

BorderThickness="{TemplateBinding BorderThickness}" />

<ScrollViewer x:Name="PART\_ContentHost" />

<TextBlock x:Name="PlaceholderText" Text="подтверждение пароля"

Foreground="Gray"

Visibility="Collapsed"

FontSize="22"

FontFamily="New Courier"

FontWeight="Light"

IsHitTestVisible="False"/>

</Grid>

<ControlTemplate.Triggers>

<Trigger Property="Text" Value="">

<Setter TargetName="PlaceholderText" Property="Visibility" Value="Visible"/>

</Trigger>

</ControlTemplate.Triggers>

</ControlTemplate>

</Setter.Value>

</Setter>

</Style>

<Style TargetType="TextBox" x:Key="tbxPlaceholderFirstNameStyle" BasedOn="{StaticResource tbxPlaceholderStyle}">

<Setter Property="Template">

<Setter.Value>

<ControlTemplate TargetType="TextBox">

<Grid>

<Border Background="{TemplateBinding Background}"

BorderBrush="{TemplateBinding BorderBrush}"

BorderThickness="{TemplateBinding BorderThickness}" />

<ScrollViewer x:Name="PART\_ContentHost" />

<TextBlock x:Name="PlaceholderText" Text="имя"

Foreground="Gray"

Visibility="Collapsed"

FontSize="22"

FontFamily="New Courier"

FontWeight="Light"

IsHitTestVisible="False"/>

</Grid>

<ControlTemplate.Triggers>

<Trigger Property="Text" Value="">

<Setter TargetName="PlaceholderText" Property="Visibility" Value="Visible"/>

</Trigger>

</ControlTemplate.Triggers>

</ControlTemplate>

</Setter.Value>

</Setter>

</Style>

<Style TargetType="TextBox" x:Key="tbxPlaceholderMiddleNameStyle" BasedOn="{StaticResource tbxPlaceholderStyle}">

<Setter Property="Template">

<Setter.Value>

<ControlTemplate TargetType="TextBox">

<Grid>

<Border Background="{TemplateBinding Background}"

BorderBrush="{TemplateBinding BorderBrush}"

BorderThickness="{TemplateBinding BorderThickness}" />

<ScrollViewer x:Name="PART\_ContentHost" />

<TextBlock x:Name="PlaceholderText" Text="отчество"

Foreground="Gray"

Visibility="Collapsed"

FontSize="22"

FontFamily="New Courier"

FontWeight="Light"

IsHitTestVisible="False"/>

</Grid>

<ControlTemplate.Triggers>

<Trigger Property="Text" Value="">

<Setter TargetName="PlaceholderText" Property="Visibility" Value="Visible"/>

</Trigger>

</ControlTemplate.Triggers>

</ControlTemplate>

</Setter.Value>

</Setter>

</Style>

<Style TargetType="TextBox" x:Key="tbxPlaceholderLastNameStyle" BasedOn="{StaticResource tbxPlaceholderStyle}">

<Setter Property="Template">

<Setter.Value>

<ControlTemplate TargetType="TextBox">

<Grid>

<Border Background="{TemplateBinding Background}"

BorderBrush="{TemplateBinding BorderBrush}"

BorderThickness="{TemplateBinding BorderThickness}" />

<ScrollViewer x:Name="PART\_ContentHost" />

<TextBlock x:Name="PlaceholderText" Text="фамилия"

Foreground="Gray"

Visibility="Collapsed"

FontSize="22"

FontFamily="New Courier"

FontWeight="Light"

IsHitTestVisible="False"/>

</Grid>

<ControlTemplate.Triggers>

<Trigger Property="Text" Value="">

<Setter TargetName="PlaceholderText" Property="Visibility" Value="Visible"/>

</Trigger>

</ControlTemplate.Triggers>

</ControlTemplate>

</Setter.Value>

</Setter>

</Style>

<Style TargetType="TextBox" x:Key="tbxPlaceholderEmailStyle" BasedOn="{StaticResource tbxPlaceholderStyle}">

<Setter Property="Template">

<Setter.Value>

<ControlTemplate TargetType="TextBox">

<Grid>

<Border Background="{TemplateBinding Background}"

BorderBrush="{TemplateBinding BorderBrush}"

BorderThickness="{TemplateBinding BorderThickness}" />

<ScrollViewer x:Name="PART\_ContentHost" />

<TextBlock x:Name="PlaceholderText" Text="эл. почта"

Foreground="Gray"

Visibility="Collapsed"

FontSize="22"

FontFamily="New Courier"

FontWeight="Light"

IsHitTestVisible="False"/>

</Grid>

<ControlTemplate.Triggers>

<Trigger Property="Text" Value="">

<Setter TargetName="PlaceholderText" Property="Visibility" Value="Visible"/>

</Trigger>

</ControlTemplate.Triggers>

</ControlTemplate>

</Setter.Value>

</Setter>

</Style>

<Style TargetType="TextBox" x:Key="tbxPlaceholderPhoneStyle" BasedOn="{StaticResource tbxPlaceholderStyle}">

<Setter Property="Template">

<Setter.Value>

<ControlTemplate TargetType="TextBox">

<Grid>

<Border Background="{TemplateBinding Background}"

BorderBrush="{TemplateBinding BorderBrush}"

BorderThickness="{TemplateBinding BorderThickness}" />

<ScrollViewer x:Name="PART\_ContentHost" />

<TextBlock x:Name="PlaceholderText" Text="телефон"

Foreground="Gray"

Visibility="Collapsed"

FontSize="22"

FontFamily="New Courier"

FontWeight="Light"

IsHitTestVisible="False"/>

</Grid>

<ControlTemplate.Triggers>

<Trigger Property="Text" Value="">

<Setter TargetName="PlaceholderText" Property="Visibility" Value="Visible"/>

</Trigger>

</ControlTemplate.Triggers>

</ControlTemplate>

</Setter.Value>

</Setter>

</Style>

<Style TargetType="Button" x:Key="btnRoundedStyle">

<Setter Property="Cursor" Value="Hand"/>

<Setter Property="Foreground" Value="White"/>

<Setter Property="FontFamily" Value="Consolas"/>

<Setter Property="FontSize" Value="22"/>

<Setter Property="Template">

<Setter.Value>

<ControlTemplate TargetType="{x:Type Button}">

<Grid>

<Border Background="{TemplateBinding Background}" CornerRadius="10">

<ContentPresenter HorizontalAlignment="Center" VerticalAlignment="Center"/>

</Border>

<Image Source="/Resources/Trash2.png" Margin="0,3"/>

</Grid>

</ControlTemplate>

</Setter.Value>

</Setter>

<Style.Triggers>

<Trigger Property="IsMouseOver" Value="True">

<Setter Property="Background" Value="Red"/>

</Trigger>

</Style.Triggers>

</Style>

</Application.Resources>

</Application>