МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

Учреждение образования «БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»

Факультет Информационных технологий

Кафедра Программной инженерии

Специальность 1-40 01 01 Программное обеспечение информационных технологий

**ПОЯСНИТЕЛЬНАЯ ЗАПИСКА  
КУРСОВОГО ПРОЕКТА:**

по дисциплине «Объектно-ориентированные технологии программирования и стандарты проектирования»

Тема «Интернет-магазин сладостей»

Исполнитель

студент 2 курса группы 4        Кравченко К.В.

(Ф.И.О.)

Руководитель работы            ассистент Северинчик Н.А.

(учен. степень, звание, должность, подпись, Ф.И.О.)

Курсовой проект защищен с оценкой

Председатель Пацей Н.В. (подпись)

Минск 2021

МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

Учреждение образования   
«БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»

Факультет информационных технологий   
Кафедра программной инженерии

Утверждаю

Заведующий кафедрой ПИ

\_\_\_\_\_\_\_\_\_\_\_\_ Н.В. Пацей

подпись инициалы и фамилия

“\_\_\_” \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_2021г.

**ЗАДАНИЕ**

**к курсовому проектированию**

**по дисциплине** «Объектно-ориентированное программирование»

|  |  |
| --- | --- |
| Специальность: 1-40 01 01 Программное обеспечение информационных технологий Группа: 4  Студент: Кравченко К.В. |  |
| **Тема:** Программное средство «Интернет-магазин сладостей» | |

**1. Срок сдачи студентом законченной работы**: "29 мая 2021 г."

**2. Исходные данные к проекту:**

**2.1**. Функционально ПС поддерживает:

* Функции администратора:
  + Добавлять товары;
  + Отслеживать товары в магазине (их название, стоимость, количество);
  + Осуществлять сортировку товаров (по названию, стоимости (поддерживается поиск по диапазону) и количеству);
  + Удалять товары;
  + Отслеживать заказы;
  + Поддерживать работу c базой данных.
* Функции клиента:
  + Выполнять регистрацию и авторизацию (с помощью имя пользователя и пароля);
  + Отслеживать товары в магазине (их название, стоимость, количество, изображение);
  + Осуществлять покупку товара;
  + Отслеживать и сортировать купленные товары в корзине (по названию, стоимости и количеству).

**2.2.** При выполнении курсового проекта необходимо использовать принципы проектирования ООП. Приложение разрабатывается под ОС Windows и представляет собой настольное приложение (desktop). Отображение, бизнес-логика должны быть максимально независимы друг от друга для возможности расширения. Диаграммы вариантов использования, классов реализации задачи, взаимодействия разработать на основе UML. Язык разработки проекта – C#. Управление программой должно быть интуитивно понятным и удобным. При разработке использовать несколько наиболее подходящих шаблонов проектирования ПО.

**3. Содержание расчетно-пояснительной записки**

(перечень вопросов, подлежащих разработке)

* Введение
* Постановка задачи и обзор литературы (алгоритмы решения, обзор прототипов, актуальность задачи)
* Проектирование архитектуры проекта (структура модулей, классов)
* Разработка функциональной модели и модели данных ПС (выполняемые функции)
* Тестирование
* Заключение
* Список используемых источников
* Приложения

**4. Форма представления выполненной курсовой работы:**

* + Теоретическая часть курсового проекта должны быть представлены в формате docx. Оформление записки должно быть согласно выданным правилам.
  + Листинги программы представляются частично в приложении.
  + Пояснительную записку, листинги, проект (инсталляцию проекта) необходимо загрузить на диск, указанный преподавателем.

#### **Календарный план**

|  |  |  |  |
| --- | --- | --- | --- |
| №  п/п | Наименование этапов курсового проекта | Срок выполнения этапов проекта | Примечание |
| 1 | Введение | 19.02.2021 |  |
| 2 | Аналитический обзор литературы по теме проекта. Изучение требований, определение вариантов использования | 12.03.2021 |  |
| 3 | Анализ и проектирование архитектуры приложения (построение диаграмм, проектирование бизнес-слоя, представления и данных) | 26.03.2021 |  |
| 4 | Проектирование структуры базы данных. Разработка дизайна пользовательского интерфейса | 02.04.2021 |  |
| 5 | Кодирование программного средства | 23.04.2021 |  |
| 6 | Тестирования и отладка программного средства | 30.04.2021 |  |
| 7 | Оформление пояснительной записки | 07.05.2021 |  |
| 8 | Сдача проекта | 29.05.2021 |  |

**5. Дата выдачи задания** 21.02.2021

Руководитель\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ *Н.А. Северинчик*

(подпись)

Задание принял к исполнению

(дата и подпись студента)

Содержание

[Введение 3](file:///C:\Users\Asus\Downloads\Poyasnitelnaya_Zapiska_Kravchenko_N_V__kopia.docx#_Toc72886572)

[1. Постановка задачи и анализ прототипов 4](file:///C:\Users\Asus\Downloads\Poyasnitelnaya_Zapiska_Kravchenko_N_V__kopia.docx#_Toc72886577)

[1.2 Средства разработки 4](file:///C:\Users\Asus\Downloads\Poyasnitelnaya_Zapiska_Kravchenko_N_V__kopia.docx#_Toc72886578)

[1.3 Обзор прототипов и аналогов 5](file:///C:\Users\Asus\Downloads\Poyasnitelnaya_Zapiska_Kravchenko_N_V__kopia.docx#_Toc72886579)

[2. Разработка функциональных требований проекта 10](file:///C:\Users\Asus\Downloads\Poyasnitelnaya_Zapiska_Kravchenko_N_V__kopia.docx#_Toc72886580)

[1.4 Обобщенная структура 10](file:///C:\Users\Asus\Downloads\Poyasnitelnaya_Zapiska_Kravchenko_N_V__kopia.docx#_Toc72886581)

[1.5 Диаграмма UML 11](file:///C:\Users\Asus\Downloads\Poyasnitelnaya_Zapiska_Kravchenko_N_V__kopia.docx#_Toc72886582)

[1.6 Диаграмма классов 12](file:///C:\Users\Asus\Downloads\Poyasnitelnaya_Zapiska_Kravchenko_N_V__kopia.docx#_Toc72886583)

[1.7 Проектирование базы данных 16](file:///C:\Users\Asus\Downloads\Poyasnitelnaya_Zapiska_Kravchenko_N_V__kopia.docx#_Toc72886584)

[3. Проектирование программного средства 19](file:///C:\Users\Asus\Downloads\Poyasnitelnaya_Zapiska_Kravchenko_N_V__kopia.docx#_Toc72886585)

3[.1 Описание функций приложения и MVVM 19](file:///C:\Users\Asus\Downloads\Poyasnitelnaya_Zapiska_Kravchenko_N_V__kopia.docx#_Toc72886586)

[3.2 Основные окна приложения 21](file:///C:\Users\Asus\Downloads\Poyasnitelnaya_Zapiska_Kravchenko_N_V__kopia.docx#_Toc72886587)

[3.2.1 Главное окно приложения 21](file:///C:\Users\Asus\Downloads\Poyasnitelnaya_Zapiska_Kravchenko_N_V__kopia.docx#_Toc72886588)

[3.3 Основные страницы приложения 21](file:///C:\Users\Asus\Downloads\Poyasnitelnaya_Zapiska_Kravchenko_N_V__kopia.docx#_Toc72886589)

[3.3.1 Страница авторизации 21](file:///C:\Users\Asus\Downloads\Poyasnitelnaya_Zapiska_Kravchenko_N_V__kopia.docx#_Toc72886590)

3[.3.2 Страница регистрации 22](file:///C:\Users\Asus\Downloads\Poyasnitelnaya_Zapiska_Kravchenko_N_V__kopia.docx#_Toc72886591)

[3.3.3 Страница клиентов 23](file:///C:\Users\Asus\Downloads\Poyasnitelnaya_Zapiska_Kravchenko_N_V__kopia.docx#_Toc72886592)

[3.3.4 Страница товаров 23](file:///C:\Users\Asus\Downloads\Poyasnitelnaya_Zapiska_Kravchenko_N_V__kopia.docx#_Toc72886593)

[3.3.5 Страница заказов 24](file:///C:\Users\Asus\Downloads\Poyasnitelnaya_Zapiska_Kravchenko_N_V__kopia.docx#_Toc72886594)

[3.3.6 Страница дополнительной информации о заказе 25](file:///C:\Users\Asus\Downloads\Poyasnitelnaya_Zapiska_Kravchenko_N_V__kopia.docx#_Toc72886595)

[4. Создание(реализация) программного средства 26](file:///C:\Users\Asus\Downloads\Poyasnitelnaya_Zapiska_Kravchenko_N_V__kopia.docx#_Toc72886596)

[5. Тестирование, проверка работоспособности и анализ полученных результатов 27](file:///C:\Users\Asus\Downloads\Poyasnitelnaya_Zapiska_Kravchenko_N_V__kopia.docx#_Toc72886597)

[6. Руководство по установке и использования 36](file:///C:\Users\Asus\Downloads\Poyasnitelnaya_Zapiska_Kravchenko_N_V__kopia.docx#_Toc72886598)

[Заключение 38](file:///C:\Users\Asus\Downloads\Poyasnitelnaya_Zapiska_Kravchenko_N_V__kopia.docx#_Toc72886599)

[Список используемых источников 39](file:///C:\Users\Asus\Downloads\Poyasnitelnaya_Zapiska_Kravchenko_N_V__kopia.docx#_Toc72886600)

Приложение А…………………………………………………………....41

Приложение Б……………………………...……………………………..41

Приложение В…………………………………………………………….43

Приложение Г…………………………………………………………….46

Приложение Д…………………………………………………………….48

ВВЕДЕНИЕ

Мы живём в эпоху современных технологий и интернета. С каждым днём покупка товаров в интернете (не выходя из дома) выходит на новый уровень. Поэтому реализация программного средства “Интернет-магазин” будет как нельзя кстати.

Моё приложение будет процессом организации структуры управления интернет-магазином на основе администратора и пользователя.

Приложение позволяет администратору отслеживать товары, заказы, управлять ими.

Пользователю же позволяется отслеживать товары в магазине, осуществлять их покупку.

Целью курсового проекта является: развитие практических навыков вёрстки приложений на базе WPF с использованием языка разметки — XAML, и языка программирования — С#, показать навыки разработки дизайна, макета для приложения и умение грамотно разделить информацию.

# 1. Анализ требований к программному средству и разработка функциональных требований

**1.1 Описание функций приложения**

Основной задачей курсового проекта является разработка приложения с возможностью управлять товарами и их движением, заказами на их покупку. Это поможет пользователям сделать многие из этих процессов автоматическими.

Функционально данное программное средство должно выполнять следующие задачи:

* выполнять регистрацию и авторизацию;
* поддерживать работу с базой данных;
* добавлять товары;
* отслеживать и управлять товарами в магазине;
* удалять товары;
* обновлять товары и их данные;
* осуществлять покупку товара
* отслеживать заказы

## 1.2 Средства разработки

При разработке приложения были использованы:

* среда разработки Microsoft Visual Studio 2019;
* технология WPF;
* объектно-ориентированная технология Entity Framework на базе фреймворка .NET;
* система управления Microsoft SQL Server.

Microsoft Visual Studio — это стартовая площадка для написания, отладки и сборки кода, а также последующей публикации приложений. Интегрированная среда разработки (IDE) представляет собой многофункциональную программу, которую можно использовать для различных аспектов разработки программного обеспечения. 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.

Windows Presentation Foundation (WPF) — это система следующего поколения для построения клиентских приложений Windows с визуально привлекательными возможностями взаимодействия с пользователем. C WPF можно создавать широкий спектр как автономных приложений, так и приложений, размещенных в веб-обозревателе. в основе WPF лежит векторная система визуализации, не зависящая от разрешения и созданная с расчетом на возможности современного графического оборудования. WPF расширяет базовую систему полным набором функций разработки приложений, элементами управления, привязкой данных, макетом, графикой, анимацией, стилями, шаблонами, документами, мультимедиа, текстом и оформлением. Именно использование WPF позволило гибко управлять дизайном интерфейса, также стало возможным подключение различных сторонних пакетов и использование паттернов.

Entity Framework представляет специальную объектно-ориентированную технологию на базе фреймворка .NET для работы с данными. Если традиционные средства ADO.NET позволяют создавать подключения, команды и прочие объекты для взаимодействия с базами данных, то Entity Framework представляет собой более высокий уровень абстракции, который позволяет абстрагироваться от самой базы данных и работать с данными независимо от типа хранилища. Если на физическом уровне мы оперируем таблицами, индексами, первичными и внешними ключами, но на концептуальном уровне, который нам предлагает Entity Framework, мы уже работает с объектами.

Microsoft SQL Server — система управления реляционными БД. SQL Server Management Studio – утилита из Microsoft SQL Server, необходимая для управления и администрирования всех компонентов Microsoft SQL Server. Она в себя включает скриптовый редактор и графическую программу, которая работает с объектами и настройками сервера. SQL является, прежде всего, информационно-логическим языком, предназначенным для описания, изменения и извлечения данных, хранимых в реляционных базах данных.

# 2. Проектирование программного средства

**2.1 Обобщённая структура**

Решение представлено проектом «OnlineStore» с имеющей структурой (рисунок 2.1).

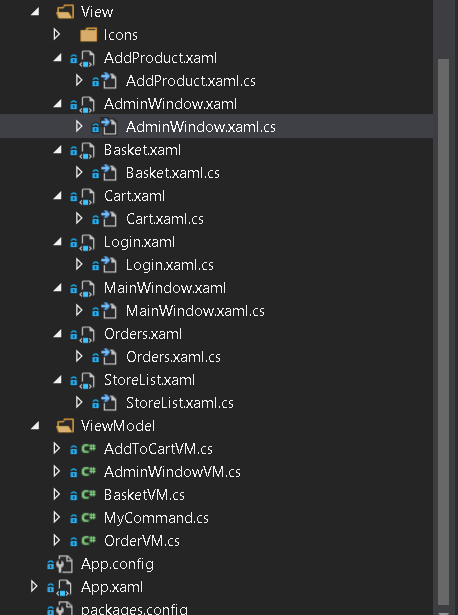
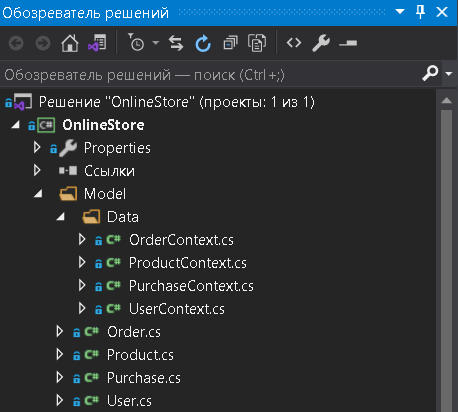


Рисунок 2.1 – Обобщённая структура проекта

Описание обобщенной структуры проекта «OnlineStore» представлено в таблице 2.1.

Таблица 2.1 – Обобщенная структура проекта

|  |  |
| --- | --- |
| Объект | Описание |
| Properties | Cвойства проекта (содержит информацию о сборке, используемых ресурсах и настройках) |
| Ссылки | Перечень сборок, используемых в проекте |
| Папка Model | Описывает используемые в приложении данные |
| Папка View | Определяет визуальный интерфейс, через который пользователь взаимодействует с приложением |
| Папка Icons | Cодержит основные картинки проекта |
| Папка ViewModel | Связывает модель и представление, определяет логику по обновлению данных в модели |
| App.config | Файл с параметрами проекта |
| App.xaml | Класс Application |
| packages.config | Файл для поддержки списка пакетов, на которые ссылается проект |

## 2.2 Диаграммы UML

UML — унифицированный язык моделирования (Unified Modeling Language) — язык графического описания для объектного моделирования в области разработки программного обеспечения, для моделирования системного проектирования и отображения организационных структур.

UML является языком широкого профиля, это — [открытый стандарт](https://ru.wikipedia.org/wiki/%D0%9E%D1%82%D0%BA%D1%80%D1%8B%D1%82%D1%8B%D0%B9_%D1%81%D1%82%D0%B0%D0%BD%D0%B4%D0%B0%D1%80%D1%82), использующий графические обозначения для создания [абстрактной модели](https://ru.wikipedia.org/wiki/%D0%90%D0%B1%D1%81%D1%82%D1%80%D0%B0%D0%BA%D1%82%D0%BD%D0%B0%D1%8F_%D0%BC%D0%BE%D0%B4%D0%B5%D0%BB%D1%8C) [системы](https://ru.wikipedia.org/wiki/%D0%A1%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0), называемой UML-моделью. UML был создан для определения, визуализации, проектирования и документирования, в основном, программных систем.

**2.2.1 Диаграмма вариантов использования**

В данной программе существует две разновидности ролей пользователя. На диаграмме вариантов использования, представлены роли администратора и пользователя (рисунок 2.2).

Администратору предоставляется:

* + Добавлять товары;
  + Отслеживать товары в магазине (их название, стоимость, количество);
  + Осуществлять сортировку товаров (по названию, стоимости (поддерживается поиск по диапазону) и количеству);
  + Удалять товары;
  + Отслеживать заказы;
  + Поддерживать работу c базой данных.

Пользователю предоставляется:

* + Выполнять регистрацию и авторизацию (с помощью имя пользователя и пароля);
  + Отслеживать товары в магазине (их название, стоимость, количество, изображение);
  + Осуществлять покупку товара;
  + Отслеживать и сортировать купленные товары в корзине (по названию, стоимости и количеству).

Добавление товаров

Отслеживание товаров в магазине

Осуществление сортировки товаров

Удаление товаров

Отслеживание заказов

Поддержка работы c базой данных

Выполнение регистрации и авторизации

Отслеживание товаров в магазине

Осуществление покупки товара

Отслеживание и сортировка купленных товаров в корзине

Пользователь

Администратор

Рисунок 2.2 – Диаграмма вариантов использования

## 2.2.2 Диаграмма классов

Структурная диаграмма языка моделирования UML, демонстрирующая общую структуру иерархии классов системы, их коопераций, атрибутов, методов, интерфейсов и взаимосвязей между ними. Широко применяется не только для документирования и визуализации, но также для конструирования посредством прямого или обратного проектирования.

На данной диаграмме классов изображены классы программы, методы, поля и свойства классов. Диаграмма классов служит для представления статической структуры модели системы в терминологии классов объектно-ориентированного программирования. Диаграмма классов может отражать различные взаимосвязи между отдельными сущностями предметной области, такими как объекты и подсистемы, а также описывает их внутреннюю структуру (поля, методы) и типы отношений (наследование, реализация интерфейсов).

Диаграмма классов моделей (описывает используемые в приложении данные) представлена на рисунке 2.3.

Рисунок 2.3 – Диаграмма классов моделей

Диаграмма классов представления (определяют визуальный интерфейс) представлена на рисунке 2.4.

Рисунок 2.4 – Диаграмма классов представления

Диаграмма классов моделей представления (связывает модель и представление через механизм привязки данных) представлена на рисунке 2.5.

Рисунок 2.5 – Диаграмма классов модели представления

## 2.3 Проектирование базы данных

Данные для работы будут храниться в базе данных. Её создание происходит с помощью Microsoft SQL Server — это система управления реляционными базами данных, которая подключается к приложению.

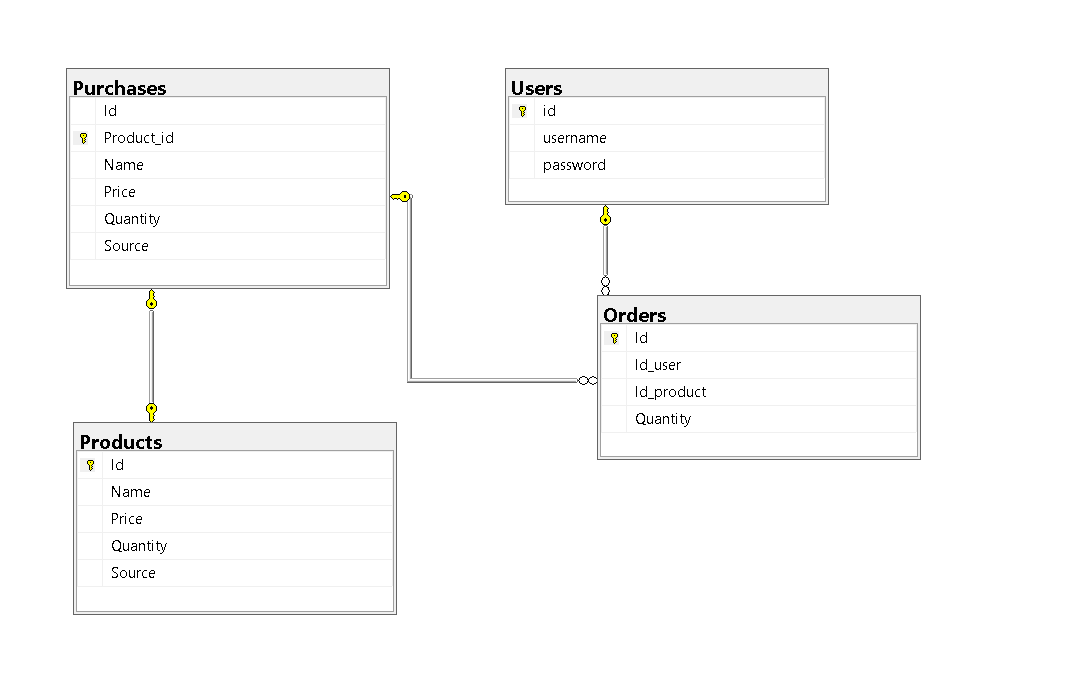
 При разработке приложения сначала разрабатывалась база данных, которая должна содержать всю нужную информацию для приложения, после программное обеспечение. База данных состоит из 4 таблиц, между которыми установлены связи один ко многим (рисунок 2.9).

Рисунок 2.9 – Диаграмма базы данных

Таблица Users предназначена для хранения информации о пользователях, состоит из 3 столбцов:

­– id - уникальный идентификатор пользователя;

– username – имя пользователя;

– password - пароль пользователя.

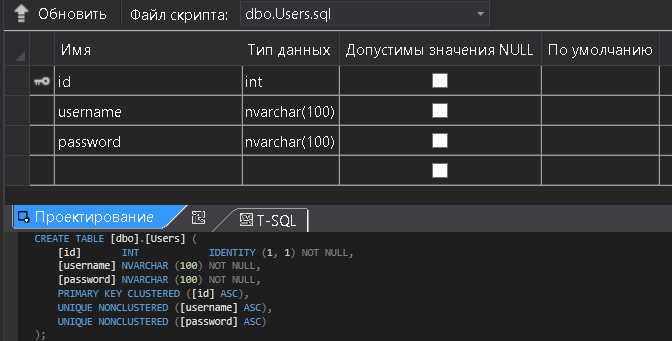


Таблица Products предназначена для хранения информации о добавленных товарах, состоит из 5 столбцов:

­– Id - уникальный идентификатор товара;

– Name - название товара;

– Price - цена товара;

– Quantity - количество товара;

– Source – изображение товара;

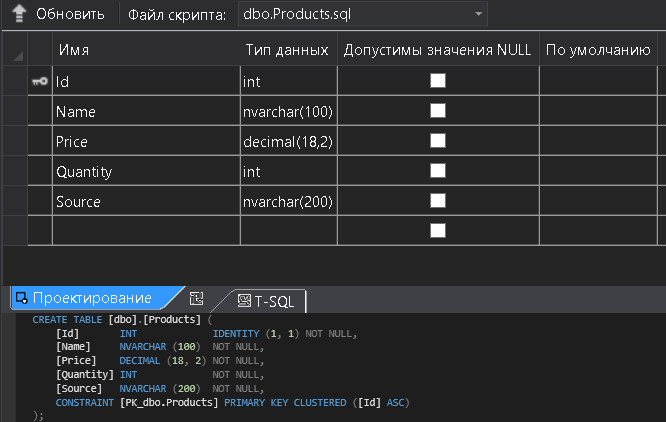


Таблица Purchases предназначена для хранения информации о покупках, состоит из 6 столбцов:

­– Id - уникальный идентификатор покупки;

– Product\_id - уникальный идентификатор товара;

– Name - название товара;

– Price - цена товара;

– Quantity - количество товара;

– Source – изображение товара;

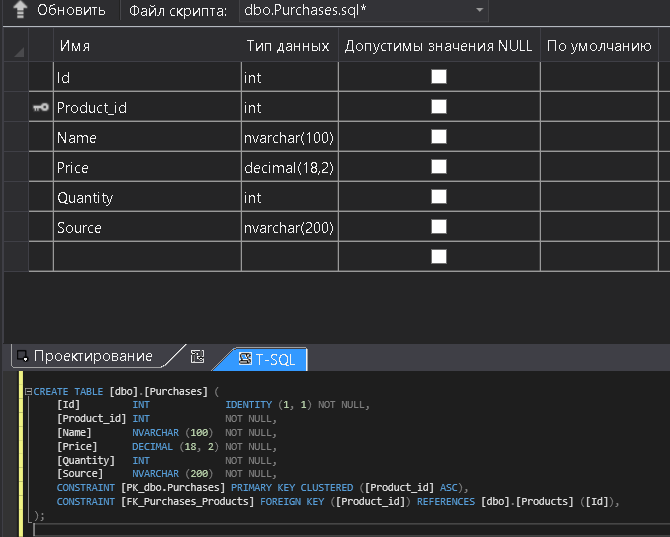


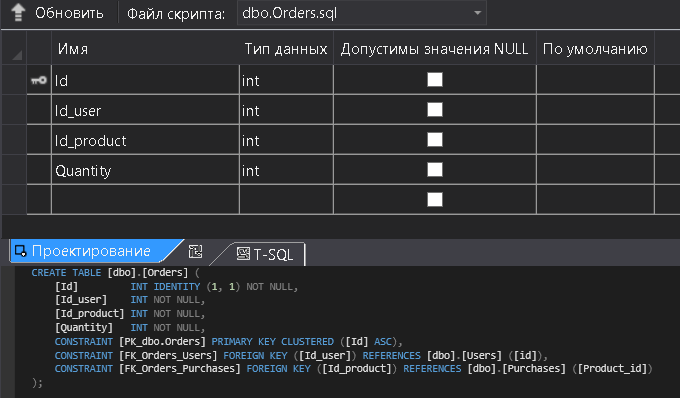
Таблица Orders предназначена для хранения информации о заказах, состоит из 4 столбцов:

­– Id - уникальный идентификатор заказа;

– Id\_user - уникальный идентификатор пользователя;

– Id\_product - уникальный идентификатор товара;

– Quantity - количество товара;



Также в каждой таблице для поля Id прописан атрибут Identity (1, 1), который позволяет сделать столбец идентификатором, значение которого будет автоматически увеличиваться на единицу для каждой следующей строки.

## 2.4 MVVM

В процессе создания приложения был реализован паттерн MVVM. Паттерн MVVM (Model-View-ViewModel) позволяет отделить логику приложения от визуальной части (представления). Данный паттерн является архитектурным, то есть он задает общую архитектуру приложения.

MVVM состоит из трех компонентов: модели (Model), модели представления (ViewModel) и представления (View).

Паттерн MVVM в WPF:



Рисунок 3.1 – схема паттерна MVVM

**Model**

Модель описывает используемые в приложении данные. Модели могут содержать логику, непосредственно связанную этими данными, например, логику валидации свойств модели. В то же время модель не должна содержать никакой логики, связанной с отображением данных и взаимодействием с визуальными элементами управления.

**View**

View или представление определяет визуальный интерфейс, через который пользователь взаимодействует с приложением. Применительно к WPF представление – это код в xaml, который определяет интерфейс в виде кнопок, текстовых полей и прочих визуальных элементов.

Хотя окно (класс Window) в WPF может содержать как интерфейс в xaml, так и привязанный к нему код C#, однако в идеале код C# не должен содержать какой-то логики, кроме разве что конструктора, который вызывает метод InitializeComponent и выполняет начальную инициализацию окна. Вся же основная логика приложения выносится в компонент ViewModel.

Однако иногда в файле связанного кода все может находиться некоторая логика, которую трудно реализовать в рамках паттерна MVVM во ViewModel.

Представление не обрабатывает события за редким исключением, а выполняет действия в основном посредством команд.

**ViewModel**

ViewModel или модель представления связывает модель и представление через механизм привязки данных. Если в модели изменяются значения свойств, при реализации моделью интерфейса INotifyPropertyChanged автоматически идет изменение отображаемых данных в представлении, хотя напрямую модель и представление не связаны.

ViewModel также содержит логику по получению данных из модели, которые потом передаются в представление. И также VewModel определяет логику по обновлению данных в модели.

Поскольку элементы представления, то есть визуальные компоненты типа кнопок, не используют события, то представление взаимодействует с ViewModel посредством команд.

Например, пользователь хочет сохранить введенные в текстовое поле данные. Он нажимает на кнопку и тем самым отправляет команду во ViewModel. А ViewModel уже получает переданные данные и в соответствии с ними обновляет модель.

Итогом применения паттерна MVVM является функциональное разделение приложения на три компонента, которые проще разрабатывать и тестировать, а также в дальнейшем модифицировать и поддерживать.

**3. Создание (реализация) программного средства**

**3.1 Основные окна**

**3.1.1 Окно авторизации**

Окно авторизации содержит поля *ИМЯ ПОЛЬЗОВАТЕЛЯ* и *ПАРОЛЬ*. Вход не осуществиться до тех пор, пока не будут введены корректные данные в поля.

Если при авторизации введены данные админа, то мы попадаем в окно админа AdminWindow (рисунок 3.3).

Иначе же мы можем перейти в окно регистрации пользователя MainWindow (рисунок 3.2).

После успешной регистрации мы можем ввести данные и перейти в главное окно магазина StoreList (рисунок 3.6).

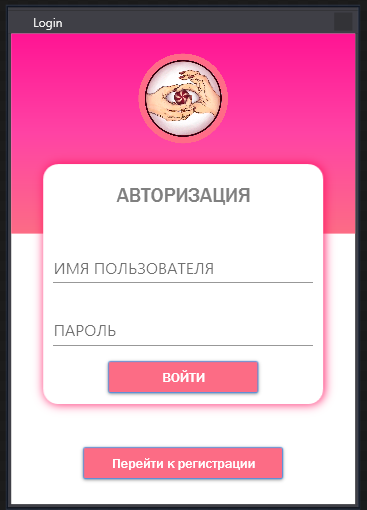
****

Рисунок 3.1 – Окно авторизации Login

**3.1.2 Окно регистрации**

Окно регистрации содержит поля *ИМЯ ПОЛЬЗОВАТЕЛЯ*, *ПАРОЛЬ* и *ПОДТВЕРЖДЕНИЕ ПАРОЛЯ*. Регистрация не осуществиться до тех пор, пока не будут введены корректные данные в поля.

После успешной регистрации нас автоматически перенаправляет в окно авторизации Login (рисунок 3.1).

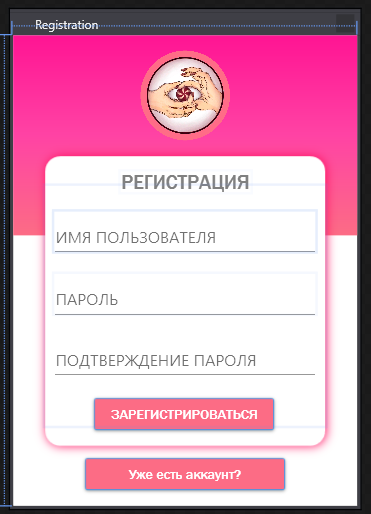
****

Рисунок 3.2 – Окно регистрации MainWindow

**3.1.3 Окно админа**

В окне админа мы можем наблюдать за товарами в нашем магазине, сортировать их, удалять.

Из окна админа мы можем перейти в окно добавления товара AddProduct (рисунок 3.4).

Так же мы можем перейти в окно наблюдения за заказами Orders (рисунок 3.5) и выйти в окно авторизации Login (рисунок 3.1).

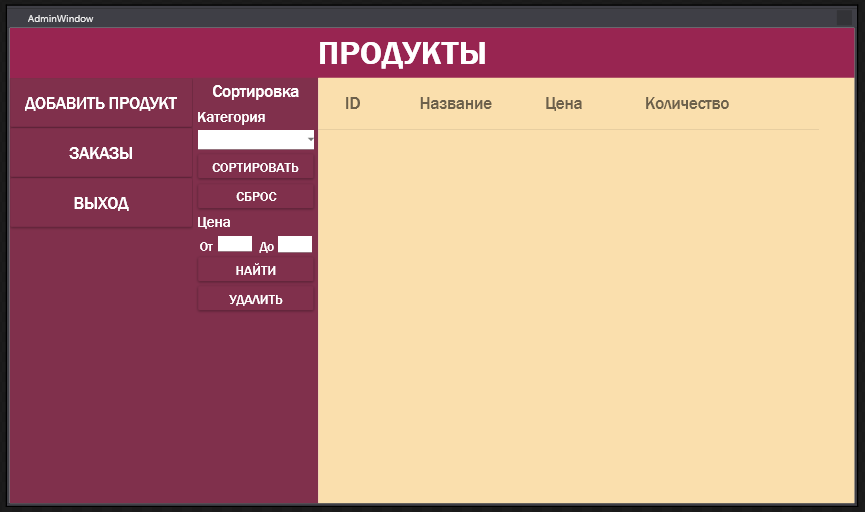
****

Рисунок 3.3 – Окно админа AdminWindow

**3.1.4 Окно добавления товара**

В окне добавления товара мы можем добавлять товары в наш магазин.

Окно добавления товара содержит поля *НАЗВАНИЕ ПРОДУКТА*, *ЦЕНА* ПРОДУКТА, КОЛИЧЕСТВО ПРОДУКТАикнопку добавления *ИЗОБРАЖЕНИЯ ПРОДУКТА*. Добавление товара не осуществиться до тех пор, пока не будут введены корректные данные в поля.

Из окна добавления товара мы можем перейти в окно админа AdminWindow (рисунок 3.3).

Так же мы можем выйти в окно авторизации Login (рисунок 3.1).

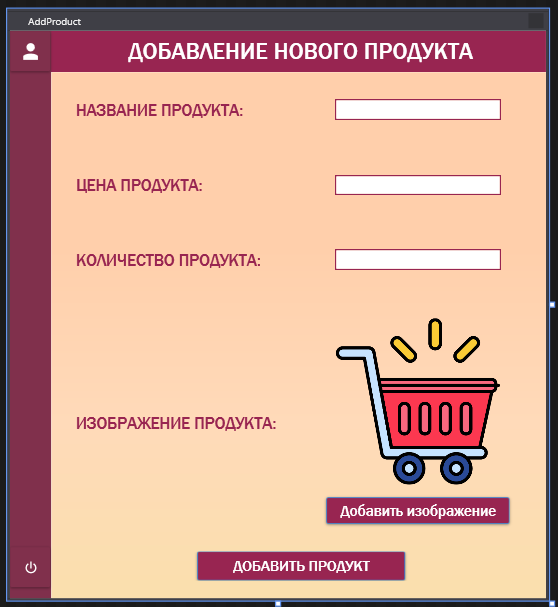


Рисунок 3.4 – Окно добавления товара AddProduct

**3.1.5 Окно наблюдения за заказами**

В окне наблюдения за заказами мы можем наблюдать за заказами, которые были осуществлены.

Из окна наблюдения за заказами мы можем перейти в окно админа AdminWindow (рисунок 3.3).

Так же мы можем выйти в окно авторизации Login (рисунок 3.1).

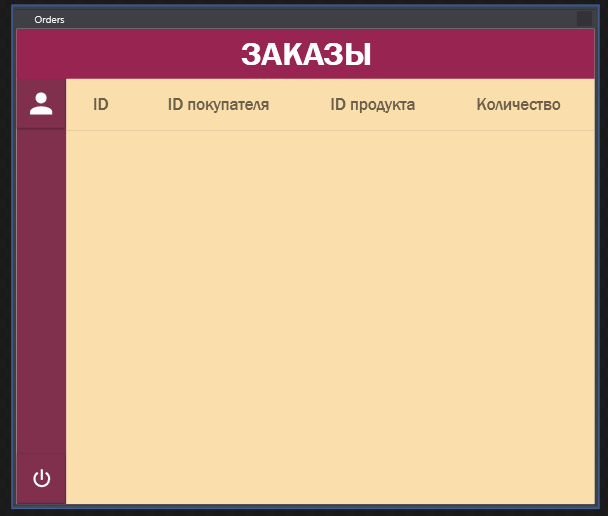
****

Рисунок 3.5 – Окно наблюдения за заказами Orders

**3.1.6 Главное окно магазина**

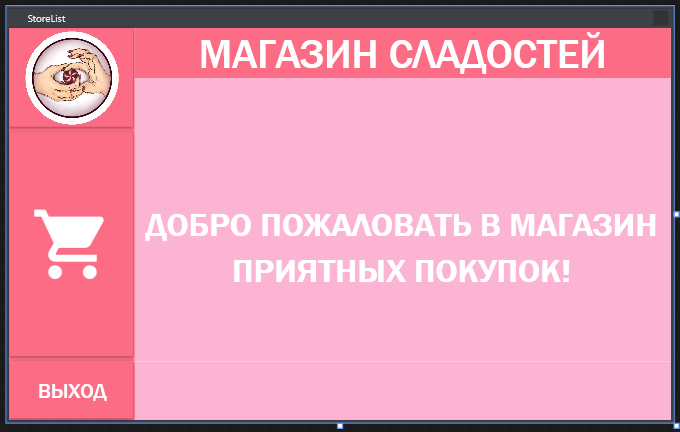
Из главного окна магазина мы можем перейти в окно покупки Cart (рисунок 3.7) или же выйти в окно авторизации Login (рисунок 3.1).

Рисунок 3.6 – Главное окно магазина StoreList

**3.1.7 Окно покупки**

В окне покупки мы можем осуществлять покупку товаров, которые представлены.

После этого мы можем перейти в окно корзины Basket (рисунок 3.8) или же перейти в главное окно магазина StoreList (рисунок 3.6).

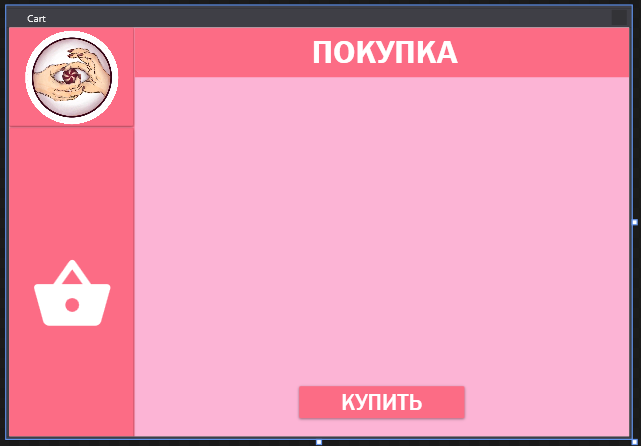
****

Рисунок 3.7 – Окно покупки Cart

**3.1.8 Окно корзины**

В окне корзины мы можем наблюдать за товарами, которые мы приобрели в магазине, а также сортировать их.

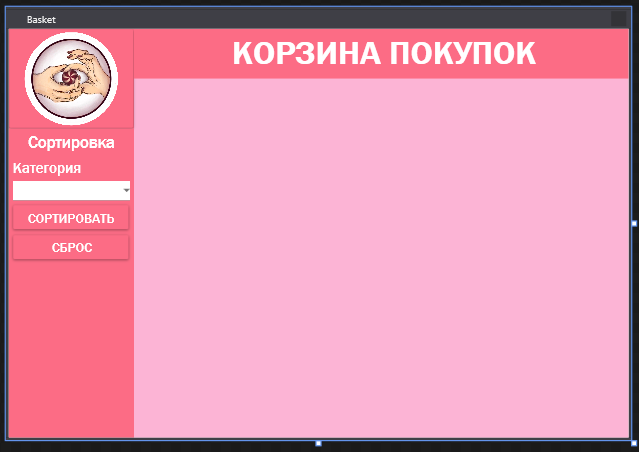
****Из окна корзины мы можем перейти в окно покупки Cart (рисунок 3.7)

Рисунок 3.8 – Окно корзины Basket

**4. Тестирование, проверка работоспособности и анализ полученных результатов**

**4.1 Тест авторизации**

Ошибка авторизации, если введены данные пользователя, которого не существует (рисунок 4.1).

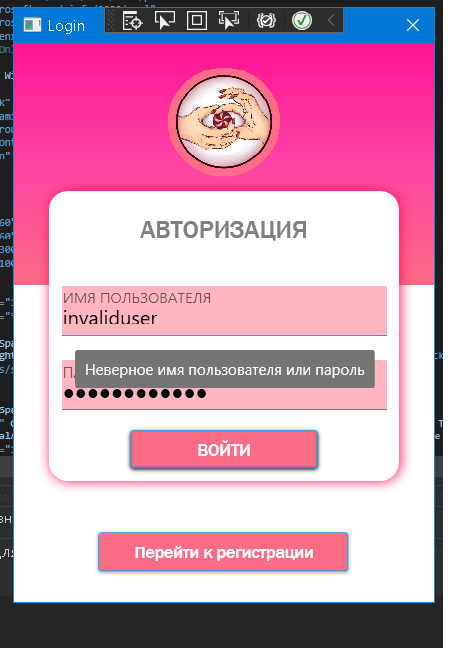
****

Рисунок 4.1 – Тест авторизации

**4.2 Тест регистрации**

Ошибка регистрации, если введённое поле *ИМЯ ПОЛЬЗОВАТЕЛЯ* содержит меньше 4 символов (рисунок 4.2).

Ошибка регистрации, если введённое поле *ПАРОЛЬ* содержит меньше 4 символов (рисунок 4.3).

Ошибка регистрации, если введённое поле *ПАРОЛЬ* не соответствует полю *ПОДТВЕРЖДЕНИЕ* *ПАРОЛЯ* (рисунок 4.4).

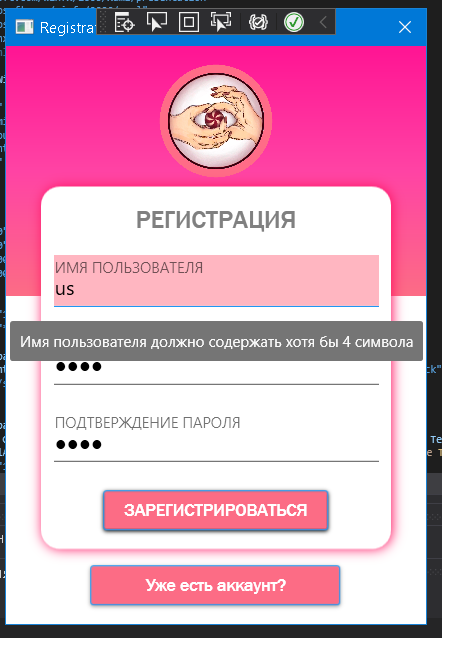
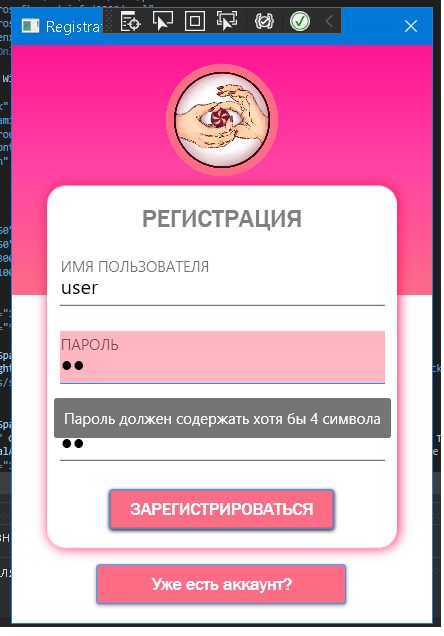


Рисунок 4.2 – Тест регистрации Рисунок 4.3 – Тест регистрации

на пароль на имя пользователя

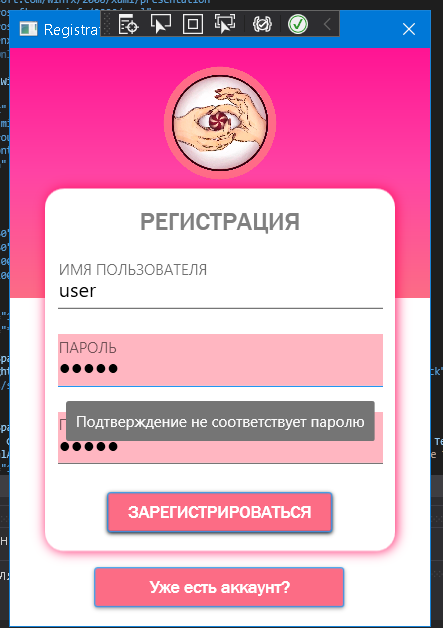


Рисунок 4.4 – Тест регистрации

на совпадение паролей

**4.3 Тест добавления товара**

Ошибка добавления товара, если существуют пустые поля (рисунок 4.5).

Ошибка добавления товара, если поля содержат 0 (рисунок 4.6).

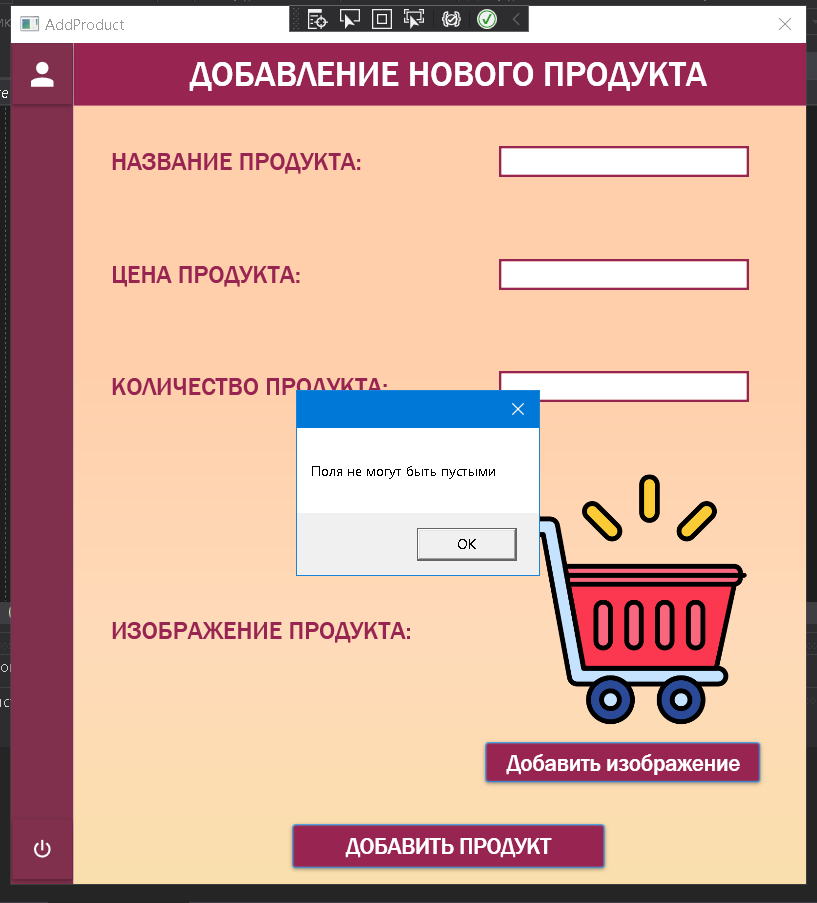
****

Рисунок 4.5 – Тест добавления товара на пустые поля

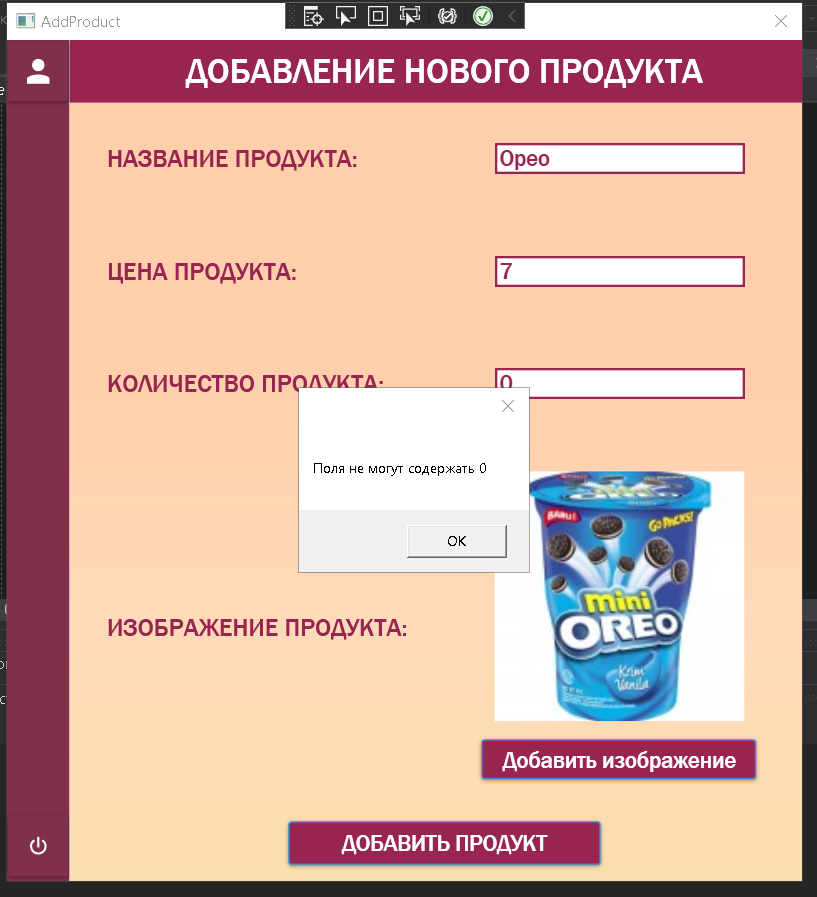
****

Рисунок 4.6 – Тест добавления товара на поля,содержащие 0

Ошибка добавления товара, если поле *ЦЕНА ПРОДУКТА* содержит что-нибудь кроме цифр и запятой (рисунок 4.7).

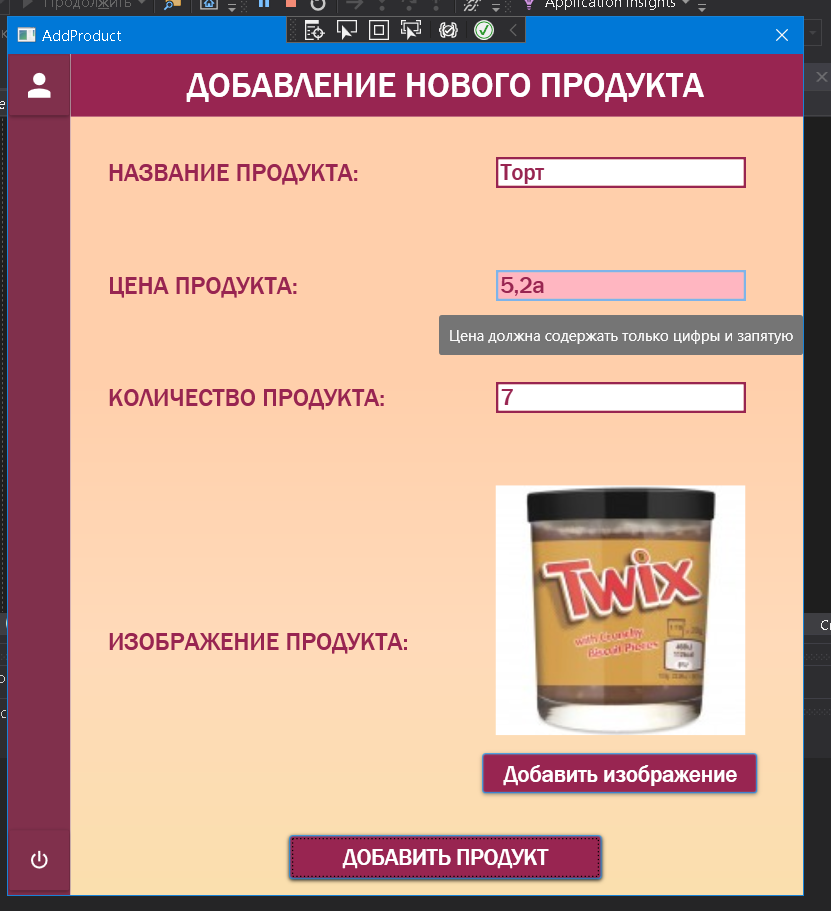


Рисунок 4.7 – Тест добавления товара на буквы в поле цена

Ошибка добавления товара, если поле *КОЛИЧЕСТВО ПРОДУКТА* содержит что-нибудь кроме цифр (рисунок 4.8).

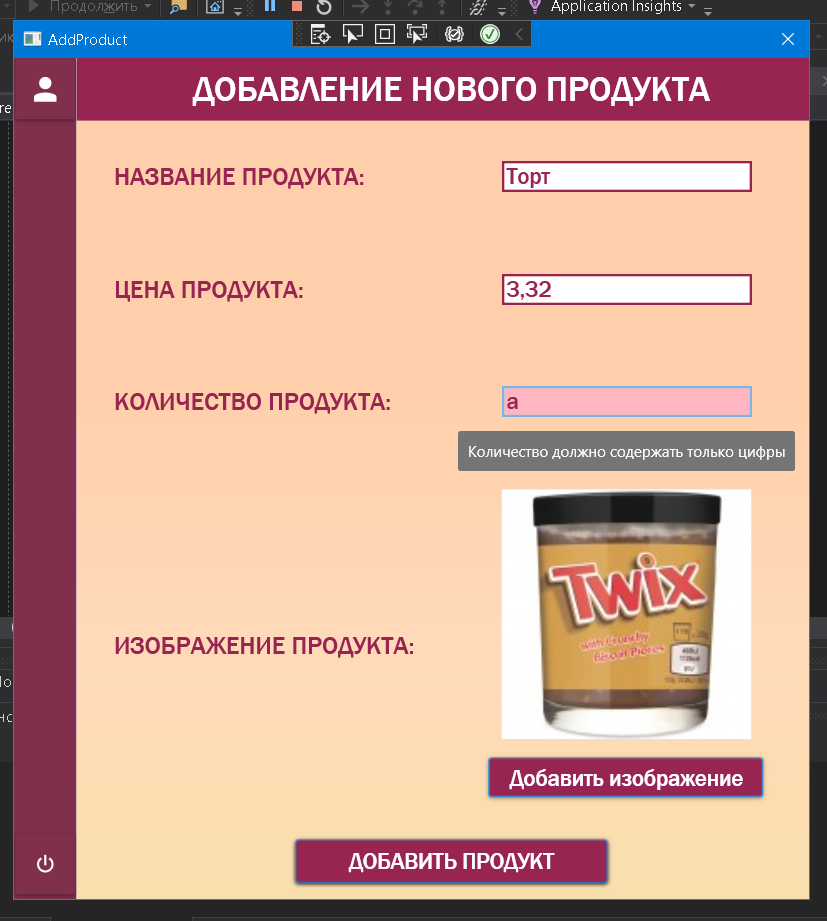


Рисунок 4.8 – Тест добавления товара на буквы в поле количество

Ошибка добавления товара, если поле *НАЗВАНИЕ ПРОДУКТА* содержит цифры (рисунок 4.9).

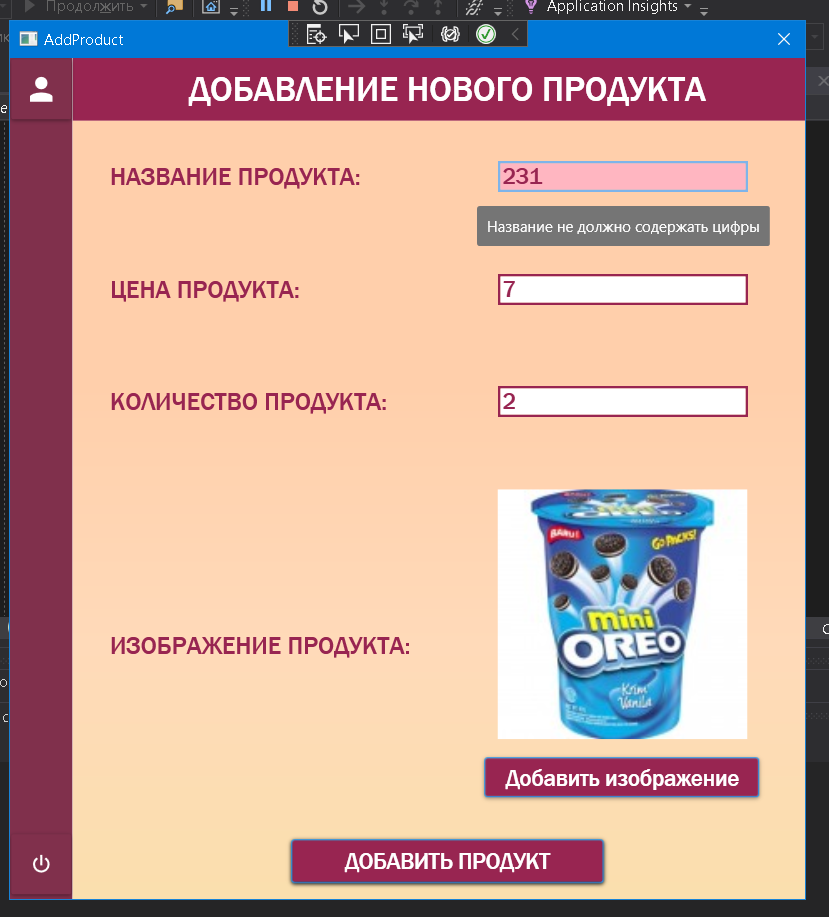


Рисунок 4.9 – Тест добавления товара на цифры в поле название

**4.4 Тест покупки товара**

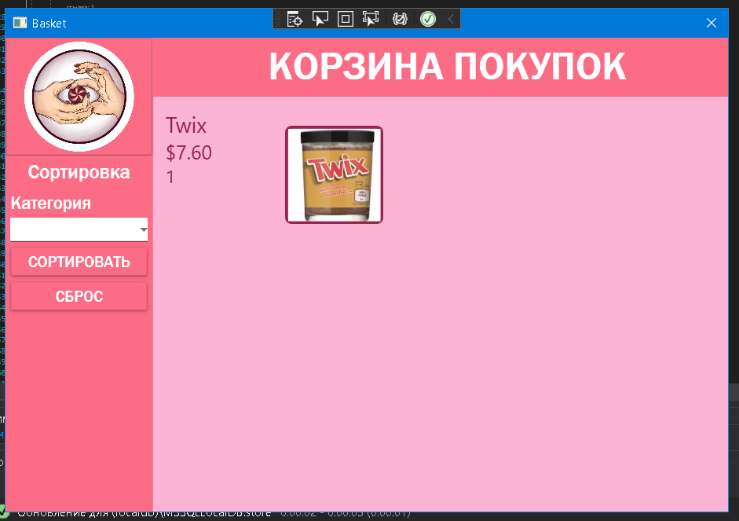
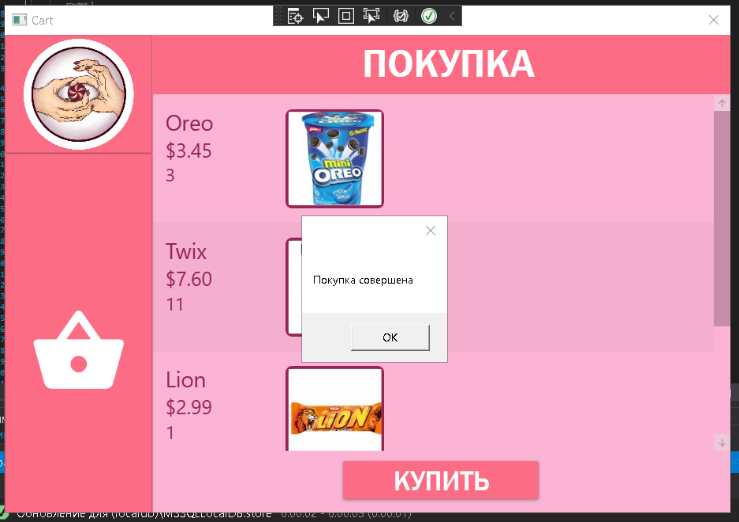
****

Рисунок 4.10 – Тест покупки товара

**4.5 Тест наблюдения за заказами**

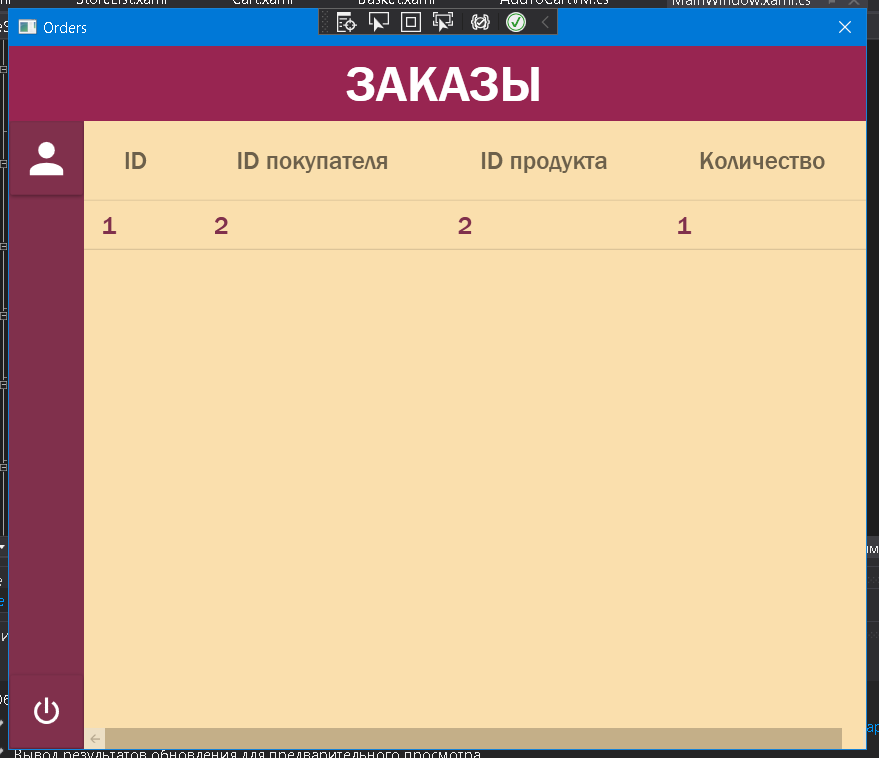
****

Рисунок 4.11 – Тест наблюдения за заказами

**4.6 Тест сортировки товаров**

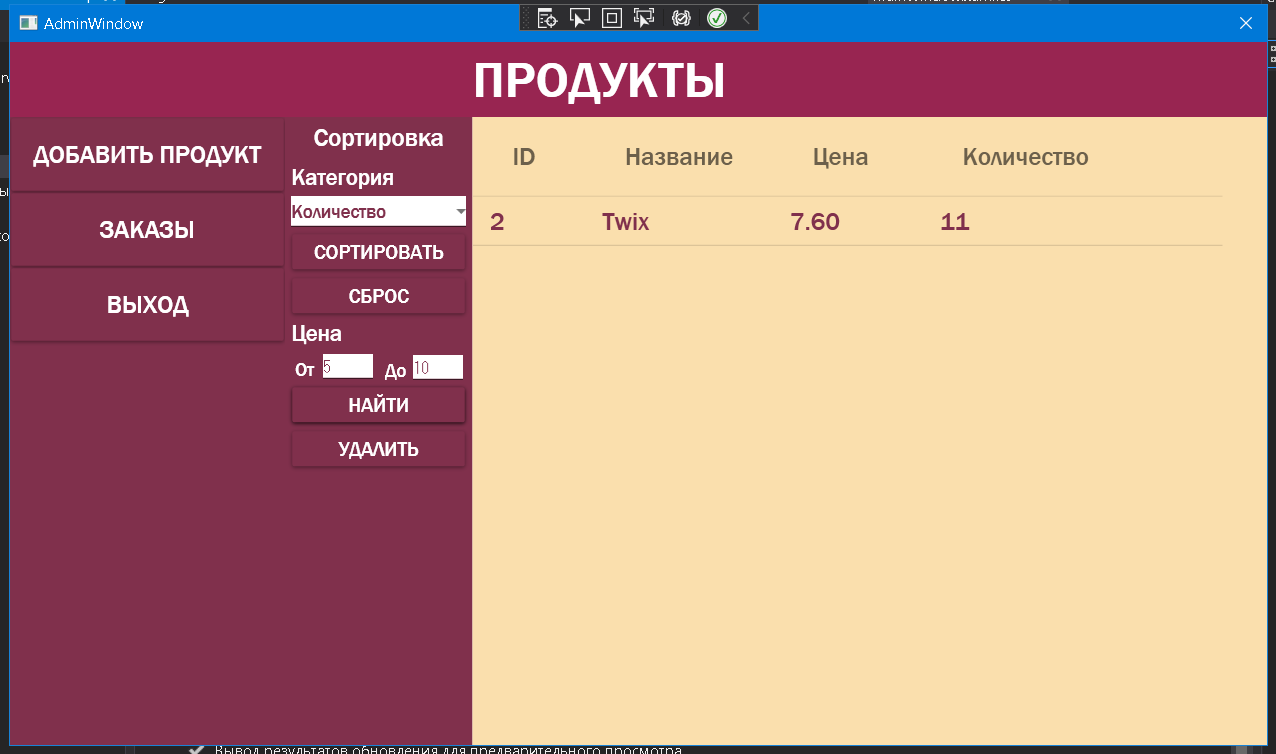
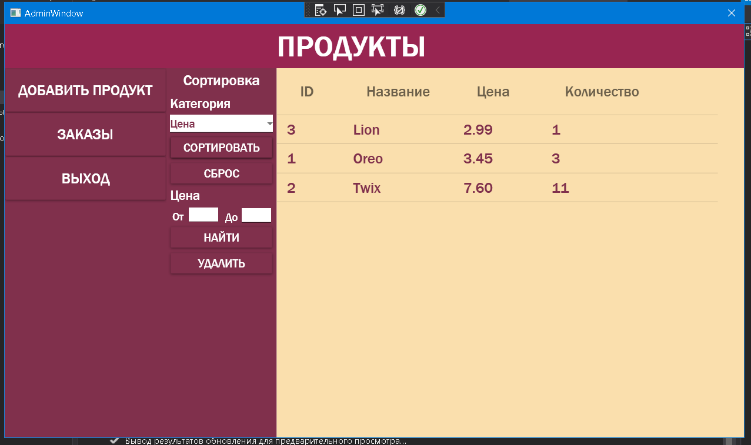
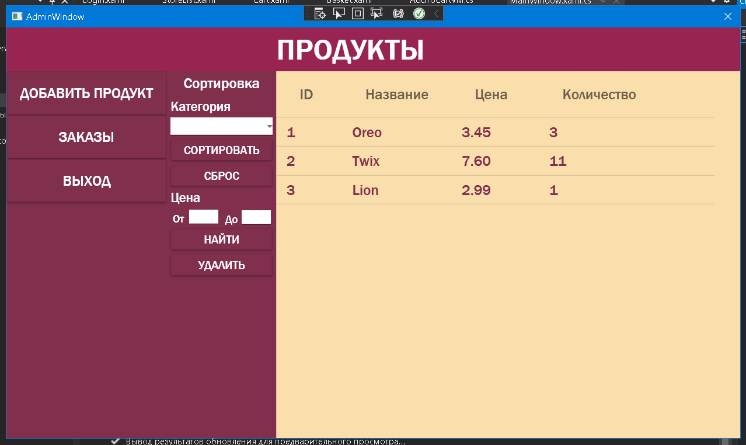
****

Рисунок 4.12 – Тест сортировки товаров (администратор)

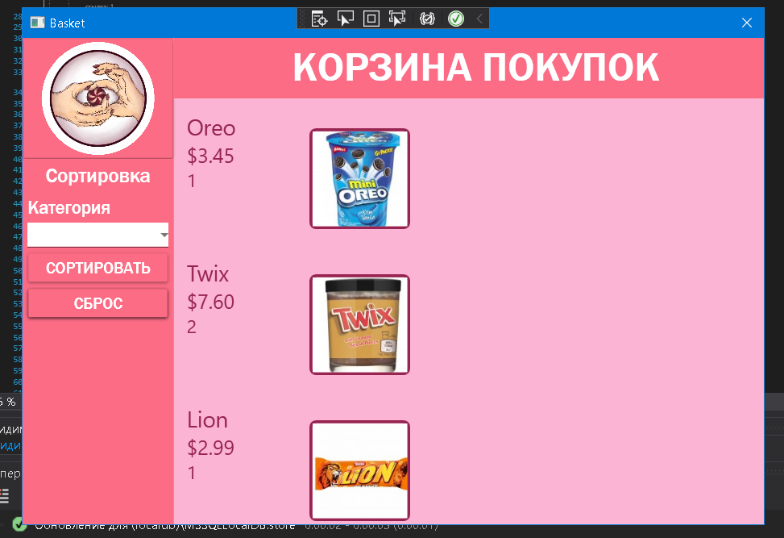
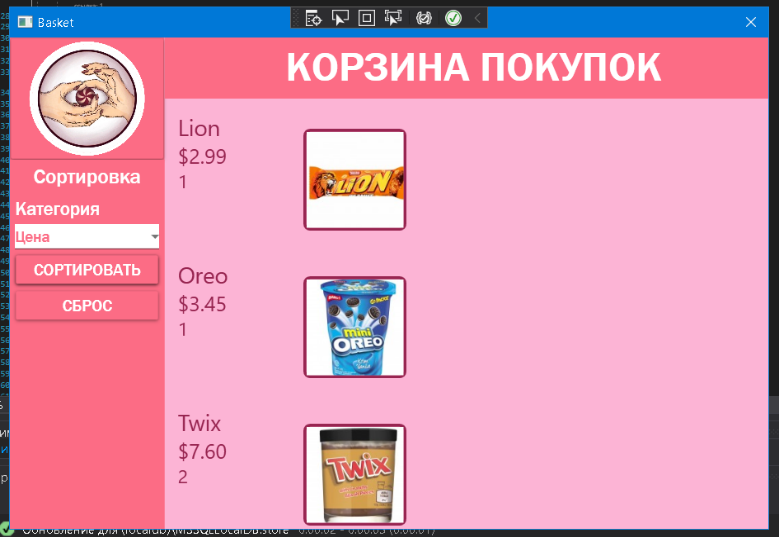


Рисунок 4.13 – Тест сортировки товаров (пользователь)

# 5. Руководство по установке и использования

На начальной странице необходимо ввести учётные данные (или же зарегистрироваться), для дальнейшего входа в программу (рисунок 5.1). В зависимости от введённых данных открывается либо окно администрирования, где администратор может добавлять, удалять, сортировать товары и наблюдать за заказами (рисунок 5.2), либо окно магазина, где пользователь может осуществлять покупку товаров, а также наблюдение за ними в корзине и их сортировку (рисунок 5.3).

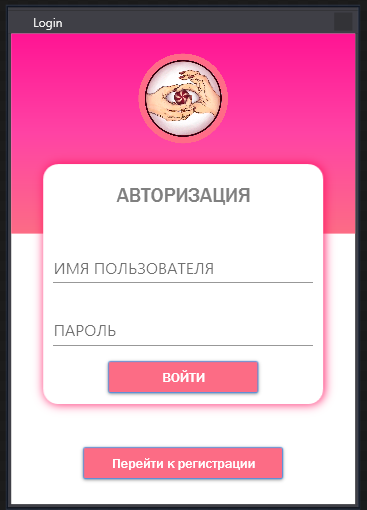
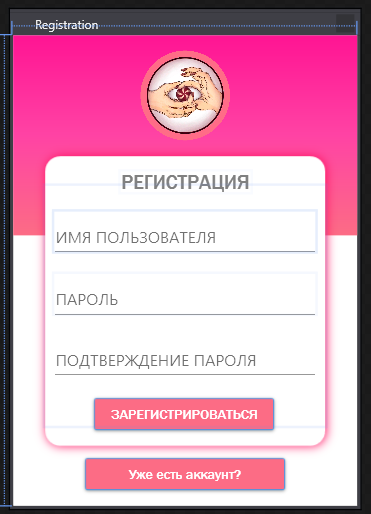
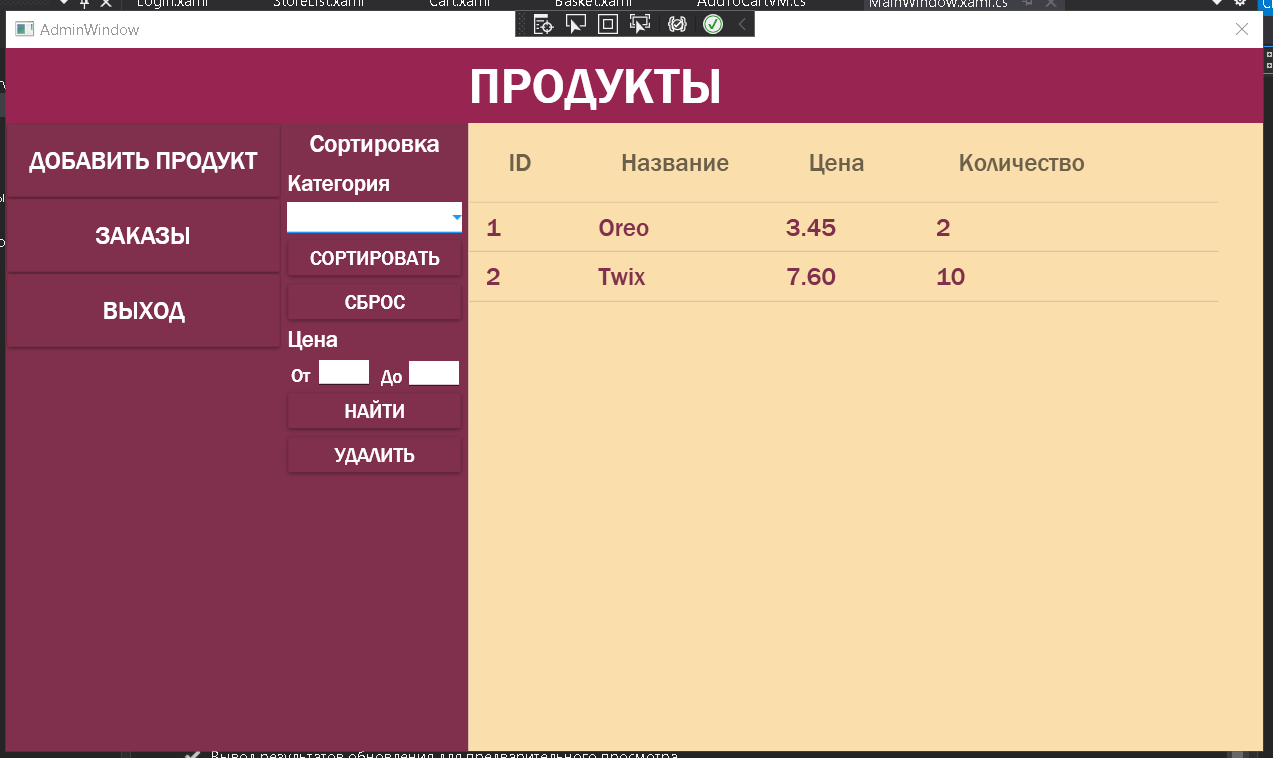


Рисунок 5.1 – Окна авторизации и регистрации

Рисунок 5.2 – Окно администрирования

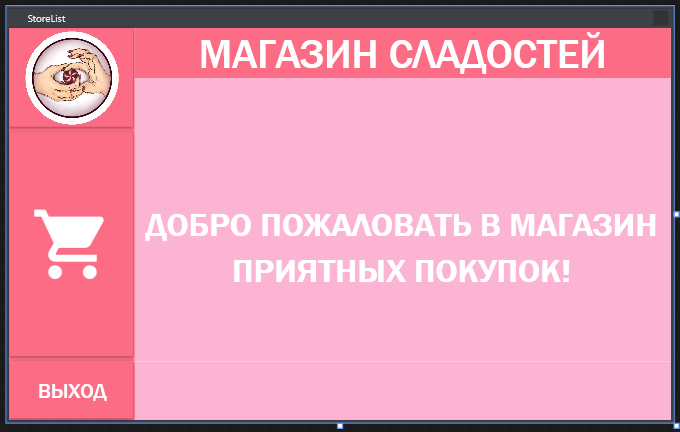


Рисунок 5.3 – Окно магазина

# ЗАКЛЮЧЕНИЕ

Итогом курсового проекта является рабочее приложение типа «Интернет-магазин» с возможностью разделения на администратора и пользователя, где администратор осуществляет добавление, наблюдение, сортировку товара и также наблюдение за заказами, а пользователь осуществляет покупку товаров, наблюдение за ними и сортировку.

В ходе выполнения курсового проекта было выполнено следующее:

* создание базы данных для хранения всей информации;
* разработка архитектуры приложения;
* разработка функциональной части приложения;
* разработка пользовательского интерфейса;
* написание исходного кода приложения;
* тестирование приложения.

В соответствии с полученным результатом работы программы можно сделать вывод, что требования технического задания выполнены, интерфейс получился достаточно простой и понятный, в нём сможет разобраться любой, даже далёкий от информационных технологий пользователь.

За время выполнения курсового проекта по созданию приложения были изучены способы работы с «SQL Server», принципы создания приложения с использованием .NET и WPF, изучен новый язык верстки XAML для построения интерфейса и работы с ним. Использовался Entity Framework для работы с базой данный, а само приложение написано на языке С#.

# Список используемых источников

1. Введение в WPF, XAML, Паттерн MVVM [Электронный ресурс]. / Режим доступа <https://metanit.com/>
2. Учебник по языку SQL [Электронный ресурс]. / Режим доступа: <https://habr.com/ru/post/255361/>