МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

Учреждение образования   
«БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»

Факультет информационных технологий   
Кафедра программной инженерии

Утверждаю

Заведующий кафедрой

\_\_\_\_\_\_\_\_\_\_\_\_ \_\_ Н.В. Пацей

подпись инициалы и фамилия

“\_\_\_”\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_2020г.

**ЗАДАНИЕ**

**к курсовому проектированию**

**по дисциплине** "Объектно-ориентированное программирование"

|  |  |
| --- | --- |
| Специальность: 1-40 05 01 Информационные системы и технологии  Студент: Хамичёнок И.А. | Группа:\_2\_\_ |
| **Тема: Программное средство «Интернет магазин электронной техники»** | |

**1. Срок сдачи студентом законченной работы**: "20 мая 2020 г."

**2. Исходные данные к проекту:**

**2.1**. Функционально ПС поддерживает:

* Выполнять функции администратора производства:
  + Поддерживать работу c базой данных;
  + Оповещать клиента о выполнении заказа;
  + Выполнять поисковые запросы.
* Выполнять функции клиента для пользователя:
  + Выполнять регистрацию и авторизацию;
  + Заполнять форму заказа по заданным параметрам;
  + Просматривать информацию о статусе заказа

**2.2.** При выполнении курсового проекта необходимо использовать принципы и приемы 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 диск, указанный преподавателем.

#### Календарный план

|  |  |  |  |
| --- | --- | --- | --- |
| №  п/п | Наименование этапов курсового проекта | Срок выполнения этапов проекта | Примечание |
| 1 | Введение | 19.02.2020 |  |
| 2 | Аналитический обзор литературы по теме проекта. Изучение требований, определение вариантов использования | 12.03.2020 |  |
| 3 | Анализ и проектирование архитектуры приложения (построение диаграмм, проектирование бизнес-слоя, представления и данных) | 26.03.2020 |  |
| 4 | Проектирование структуры базы данных. Разработка дизайна пользовательского интерфейса | 2.04.2020 |  |
| 5 | Кодирование программного средства | 23.04.2020 |  |
| 6 | Тестирования и отладка программного средства | 30.04.2020 |  |
| 7 | Оформление пояснительной записки | 7.05.2020 |  |
| 9 | Сдача проекта | 20.05.2020 |  |

**5. Дата выдачи задания \_\_\_\_**12.02.2020**\_\_\_\_**

Руководитель\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ *Д.А. Радиванович*

(подпись)

Задание принял к исполнению \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(дата и подпись студента)

**СОДЕРЖАНИЕ**

[ВВЕДЕНИЕ 4](#_Toc41617230)

[1 Аналитический обзор литературы 5](#_Toc41617231)

[2 Анализ требований к программному средству и разработка функциональных требований 7](#_Toc41617232)

[2.1 Постановка требований 7](#_Toc41617233)

[2.2 Диаграмма вариантов использования 7](#_Toc41617234)

[2.3 Анализ прототипов и аналогов 9](#_Toc41617235)

[3 Разработка архитектуры проекта 12](#_Toc41617236)

[3.1 Обобщенные структуры 12](#_Toc41617237)

[3.2 Диаграммы UML 13](#_Toc41617238)

[3.3 Диаграмма классов 14](#_Toc41617239)

[3.4 Схема базы данных 14](#_Toc41617240)

[4 Разработка функциональной модели 19](#_Toc41617241)

[4.1 Реализация авторизации и регистрации пользователя 19](#_Toc41617242)

[4.2 Реализация функцию добавления товара в корзину и оформления заказа. 20](#_Toc41617243)

[4.3 Реализация функций работы администратора 23](#_Toc41617244)

[4.4 Функции для работы с базой данных 26](#_Toc41617245)

[5 Тестирование 26](#_Toc41617246)

[6 Руководство по использованию 29](#_Toc41617247)

[6.1 Регистрация и авторизация 29](#_Toc41617248)

[6.2 Использование приложения обычным пользователем 31](#_Toc41617249)

[6.3 Использование приложения администратором 34](#_Toc41617250)

[ЗАКЛЮЧЕНИЕ 37](#_Toc41617251)

[СПИСОК ИСПОЛЬЗУЕМОЙ ЛИТЕРАТУРЫ 38](#_Toc41617252)

# ВВЕДЕНИЕ

Интернет-магазин — сайт, торгующий товарами посредством сети Интернет. Позволяет пользователям онлайн, в своём браузере или через мобильное приложение, сформировать заказ на покупку, выбрать способ оплаты и доставки заказа, оплатить заказ.

Предоставленная цифровым пространством возможность любому человеку в любой точке земного шара найти и купить любой товар стирает границы территорий, нивелирует национальную самобытность, размывает все возможные барьеры, так или иначе противопоставляющие одних людей другим, в чем бы это ни выражалось — будь то языковые, религиозные, расовые разграничения, предубеждения или неприязнь между народами.

Можно предположить, что поколение next перестанет рассматривать покупку как необходимость перемещения куда-либо в реальном пространстве. Опустение торговых центров, сокращение количества горожан, спешащих на рынки и в магазины, а также снижение нагрузки на транспортную инфраструктуру— важные последствия киберторговли, которые влекут за собой изменение городской среды.

Цель моего курсового проекта разработать интернет магазин для покупки электротехники, с возможностью оформления заказа, доставка этого заказа на определенную дату и адрес проживания, уведомление о принятии заказа.

При использовании данного приложения, каждый пользователь компьютера сможет просматривать каталог электротехники, добавлять товары в корзину, совершать заказ в режиме реального времени. Язык разработки проекта – C#. При выполнении курсового проекта будут использованы принципы и приемы ООП. Также будут использоваться технологии Windows Presentation Foundation(WPF).

# 1 Аналитический обзор литературы

При создании проекта использовались две среды разработки: одна — для работы с базами данных, другая — для создания самого приложения.

Microsoft SQL Server — система управления реляционными БД. Основной используемый язык запросов – Transact-SQL — реализован на структурированном языке запросов (SQL) с расширениями. SQL Server Management Studio — утилита из Microsoft SQL Server 2008 и более поздних версий для конфигурирования, управления и администрирования всех компонентов Microsoft SQL Server. Утилита 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.

Microsoft Visual Studio 2017 — это набор инструментов для создания программного обеспечения: от планирования до разработки пользовательского интерфейса, написания кода, тестирования, отладки, анализа качества кода и производительности, развертывания в средах клиентов и сбора данных по использованию. Эти инструменты предназначены для максимально эффективной совместной работы; все они доступны в интегрированной среде разработки (IDE) Visual Studio. Visual Studio можно использовать для создания различных типов приложений: от простых приложений для магазина и игр для мобильных клиентов до больших и сложных систем, обслуживающих предприятия и центры обработки данных.

В разработке приложения были использованы нижеперечисленные технологии.

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, в том числе Extensible Application Markup Language (XAML), элементами управления, привязкой данных, макетом, 2-D- и 3-D-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 6 представляет собой 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.

Центральной концепцией Entity Framework является понятие сущности или entity. Сущность представляет набор данных, ассоциированных с определенным объектом. Поэтому данная технология предполагает работу не с таблицами, а с объектами и их наборами.

# 2 Анализ требований к программному средству и разработка функциональных требований

## 2.1 Постановка требований

Задачей данной курсовой работы является разработка приложения, которое будет представлять собой интернет магазин. Приложение является расширяемым, так как поддерживает функции администратора, который может изменять базу данных.

Функционально ПС поддерживает:

* Выполнять функции администратора производства:
  + Поддерживать работу c базой данных;
  + Оповещать клиента о выполнении заказа;
  + Выполнять поисковые запросы.
* Выполнять функции клиента для пользователя:
  + Выполнять регистрацию и авторизацию;
  + Заполнять форму заказа по заданным параметрам;

## 2.2 Диаграмма вариантов использования

В диаграмме использования проектируемая система представляется в виде множества сущностей или актеров, взаимодействующих с системой с помощью, так называемых прецедентов. При этом актером или действующим лицом называется любая сущность, взаимодействующая с системой извне. Другими словами, каждый вариант использования определяет некоторый набор действий, совершаемый системой при диалоге с актером. При этом ничего не говорится о том, каким образом будет реализовано взаимодействие актеров с системой. Диаграмма вариантов использования для обычного пользователя приведена на рисунке 2.3.

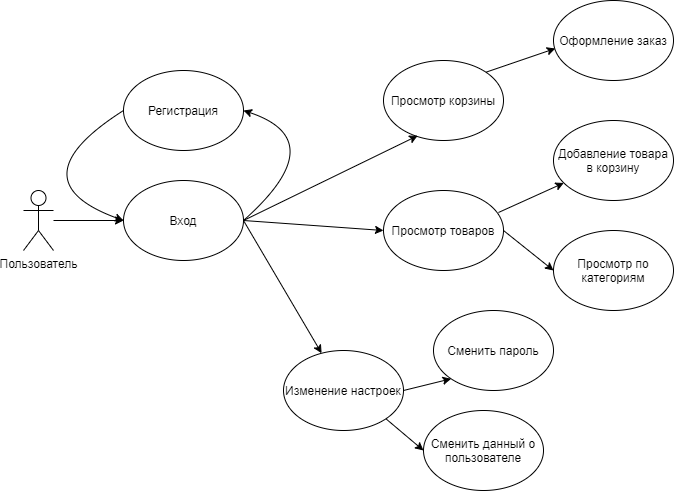


Рис.2.3 Диаграмма вариантов использования для обычного пользователя.

На данной диаграмме можно увидеть основные действия, который доступны пользователю: просмотр товаров и сопутствующие действия: добавление товара в корзину, просмотр корзины, оформление заказа, редактирование профиля, изменение пароля.

Диаграмма вариантов использования для администратора приложения приведена на рисунке 2.4.

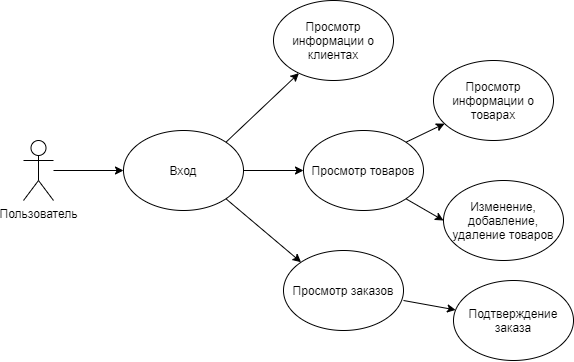


Рисунок 2.4. – Диаграмма вариантов использования для администратора.

На данной диаграмме можно увидеть основные возможности, которые доступны администратору, администратор имеет все возможности для добавления, изменения, удаления товаров. Он может изменять информацию о товарах, а также добавлять новые. Так же администратор может просматривать информацию о пользователях, заказах, и подтверждать заказы.

## 2.3 Анализ прототипов и аналогов

На сегодняшний день существует большое количество программ, предназначенных для покупки различных товаров в интернете. Одной из самых популярных является Onliner.

В качестве примера рассмотрим онлайн магазин Onliner. Интерфейс данного онлайн магазина представлен на рисунке 2.5.

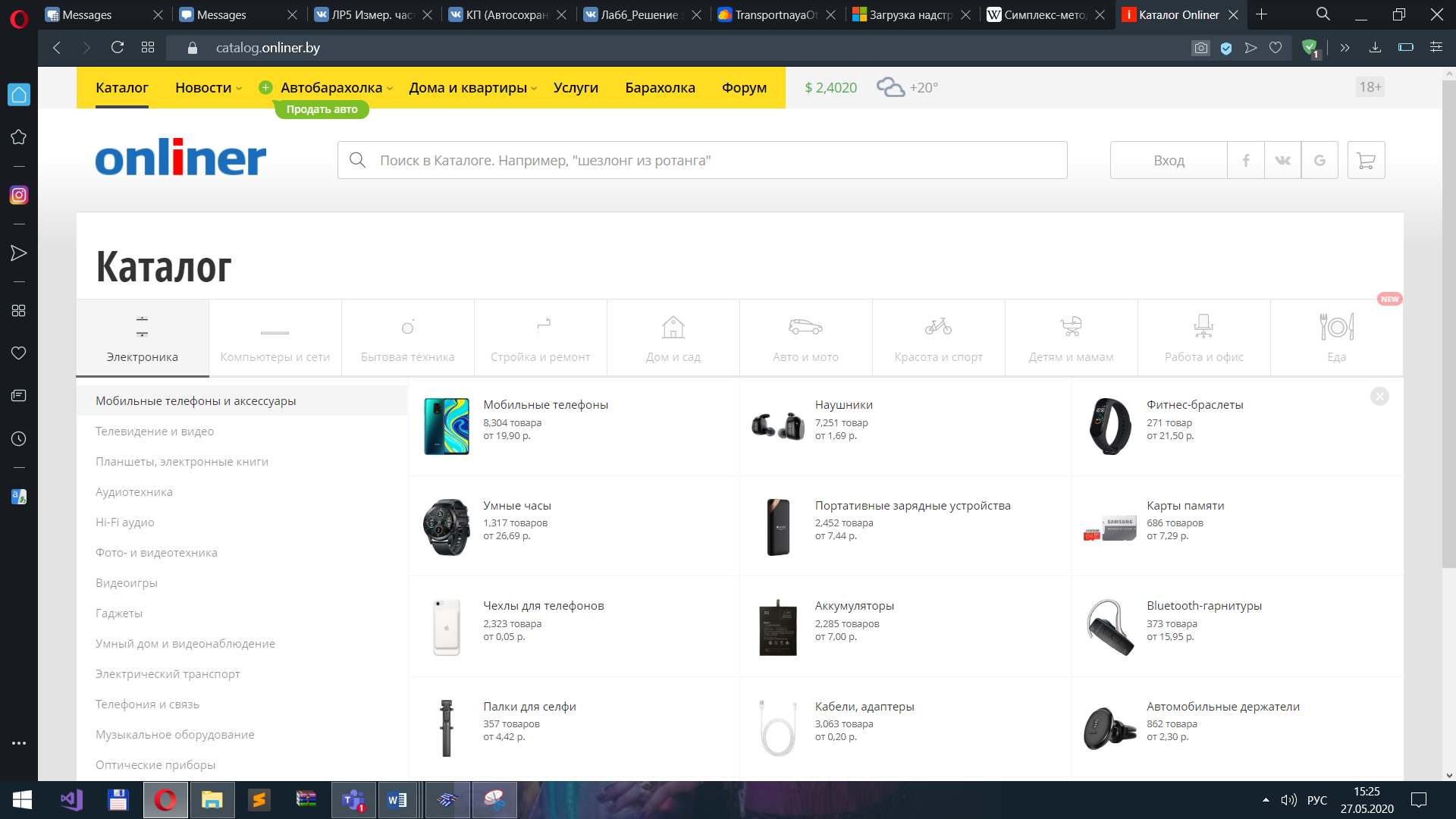


Рисунок 2.5 – Интерфейс интернет магазина Onliner.

Визуально интерфейс Onliner разбит на две визуальные части. Левая часть отведена на отображение подкатегорий товаров. А сами же категории расположились над отображением товаров. Т.е. выбрав категорию мы имеем возможность выбрать еще и под категорию товаров. Это очень упрощает поиск нужного товара, делает интерфейс более юзабельным. Функционал позволяет выбирать товар, добавлять его в карзину и оттуда совершать заказ.

Следующий крупный интернет магазин электротехники это “5 Элемент”. Интерфейс данного интернет магазина представлен на рисунке 2.6. Данный онлайн магазин имеет точно такую же структуру интерфейса как и аналог описанный ранее. Отличается только цветовая палитра данного мессенджера и небольшие различия в элементах управления. А так же наличием рекламы и т.д.

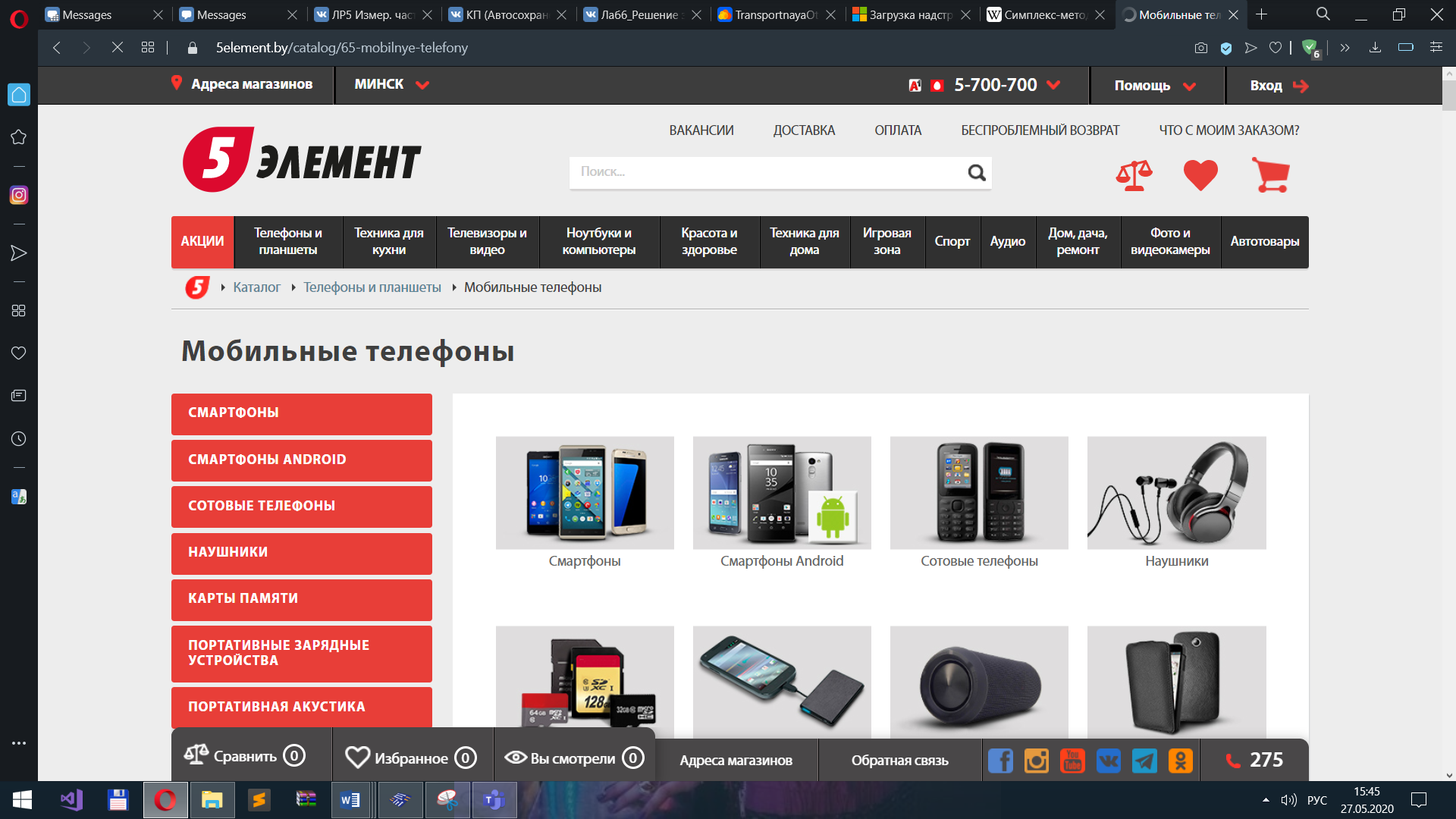


Рисунок 2.6 – Интерфейс интернет магазина “5 Элемент”.

Проанализировав прочие популярные интернет магазины, можно сделать вывод что, все они схожи в функционале и архитектуре интерфейса:

* Поле для отображения и выбора категорий;
* Добавление в корзину;
* Информация о товаре;
* Сама корзина;
* Оформление заказа.

# 3 Разработка архитектуры проекта

# 3.1 Обобщенные структуры

Решение представлено одним проектом «Курсовой», имеющим структуру, представленную на рисунке 2.1. Данный курсовой проект основывается на реали­зации паттерна MVVM (Model-View-ViewModel), который позволяет отделить логику приложения от визуальной части, разберем архитектуру данного паттерна:

* Pages — папка содержащая страницы меняя которые можно просто и удобно перемещать по категориям товаров;
* Model — папка с моделями описывающая используемые в приложении данные;
* View – папка содержащая представления определяющие визуальный интерфейс приложения;
* ViewModel – папка содержащая модель представления связывающая модель и представления через механизм привязки данных;

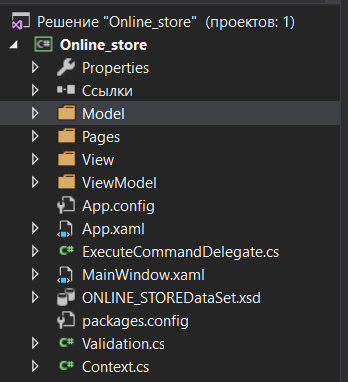


Рисунок 3.1. – Обобщенная структура проекта.

## 3.2 Диаграммы UML

UML — унифицированный язык моделирования (Unified Modeling Language) — это система обозначений, которую можно применять для объектно-ориентированного анализа и проектирования.

Его можно использовать для визуализации, спецификации, конструирова­ния и документирования программных систем.

Диаграмма — это графическое представление набора элементов, чаще всего изображенного в виде связного графа вершин (сущностей) и путей (связей).

Диаграммы последовательностей используются для уточнения диаграмм прецедентов, более детального описания логики сценариев использования. Это отличное средство документирования проекта с точки зрения сценариев использования.

Диаграмма последовательности приведена в приложении А.

Диаграмма деятельности, как и диаграмма состояний, отражает динамические аспекты поведения системы. По существу, эта диаграмма представляет собой блок-схему, которая наглядно показывает, как поток управления переходит от одной деятельности к другой.

Диаграммы деятельности используются при моделировании бизнес-процессов, технологических процессов, последовательных и параллельных вычислений.

Часть диаграммы деятельности данного приложения приведена в приложении Б, данная диаграмма приведена для части программы связанной со входом пользователя в приложение. Она разбита на 2 части: авторизация и регистрация, для каждой из этих частей приведен свой алгоритм, осуществляющий все проверки и этапы обработки информации необходимые для корректной работы данной части приложения.

Диаграмма компонентов — элемент языка моделирования UML, статическая структурная диаграмма, которая показывает разбиение программной системы на структурные компоненты и связи (зависимости) между компонентами. В качестве физических компонентов могут выступать файлы, библиотеки, модули, исполняемые файлы, пакеты и т. п.

Диаграмма развертывания показывает топологию системы и распределение компонентов системы по ее узлам, а также соединения — маршруты передачи информации между аппаратными узлами. Это единственная диаграмма, на которой применяются “трехмерные” обозначения: узлы системы обозначаются кубиками. Все остальные обозначения в UML — плоские фигуры.

Диаграммы компонентов приведена в приложении В. Так как приложение построено на архитектуре паттерна MVVM, то присутствует 3 основных блока, View, ViewModel и Model. В Model входят классы, созданные для проектирование БД. View представляет собой внешний вид приложения, а два эти компонента связаны между собой благодаря ViewModel. Данный паттерн позволяет легко извлекать информацию из базы данных и передать ее сначала во ViewModel, а затем уже отобразить во View.

## 3.3 Диаграмма классов

Диаграмма классов служит для представления статической структуры мо­дели системы в терминологии классов объектно-ориентированного программирования. Диаграмма классов может отражать, в частности, 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). На данной диаграмме не указывается информация о временных аспектах функционирования системы. С этой точки зрения диаграмма классов является дальнейшим развитием концептуальной модели проектируемой системы.

Диаграмма классов представлена в приложении Г

Классы Client, BascketProduct, Order, Product, User созданы для работы с базой данных.

Класс Context создан для создания и работы с базой данных, реализует паттерн Singleton.

Класс Validation создан для проверки на правильность вводимых данных.

В классах папки Pages я реализовал логику работы со страницами.

Класс RelayCommand реализует интерфейс Icommand.

В классе BaseViewModel реализовал интерфейс NotifyPropertyChanged.

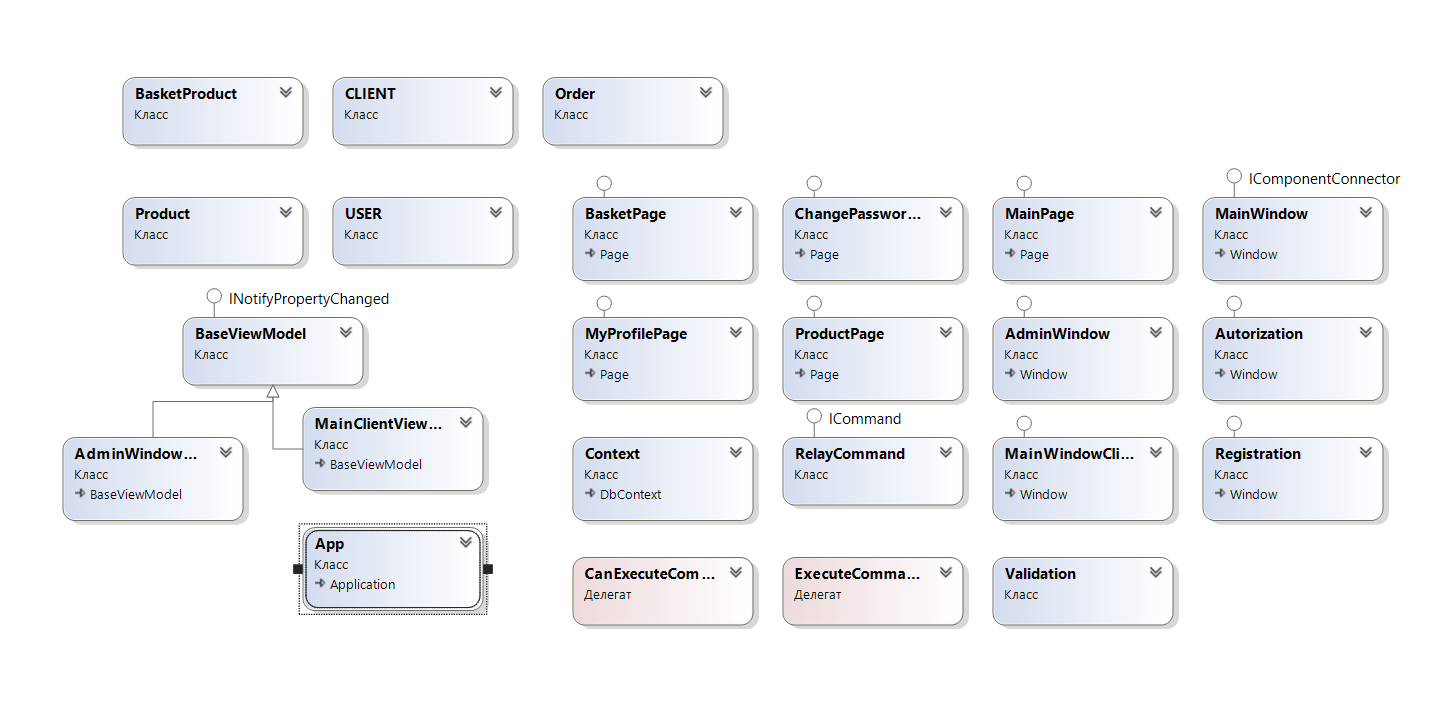


Рисунок 3.2. – Диаграмма всех классов

## 3.4 Схема базы данных

В реляционной модели, в отличие от иерархической или сетевой, не суще­ствует физических отношений. Вся информация хранится в виде таблиц (отношений), состоящих из рядов и столбцов. А данные двух таблиц связаны общими столбцами, а не физическими ссылками или указателями. Для манипуляций с рядами данных существуют специальные операторы.

В отличие от двух других типов СУБД, в реляционных моделях данных нет необходимости просматривать все указатели, что облегчает выполнение запросов на выборку информации по сравнению с сетевыми и иерархическими СУБД. Это одна из основных причин, почему реляционная модель оказалась более удобна. В реляционной модели, как объекты, так и их отношения представлены только таблицами, и ничем более.

Для реализации приложения была разработана база данных Online\_store. Для создания базы данных был выбран Microsoft SQL Server 2018.

Модель базы данных — тип модели данных, которая определяет логическую структуру базы данных и принципиально определяет, каким образом данные могут быть сохранены, организованы и обработаны.

Текущая база данных состоит из пяти таблиц, ее структура представлена на рисунке 3.3.

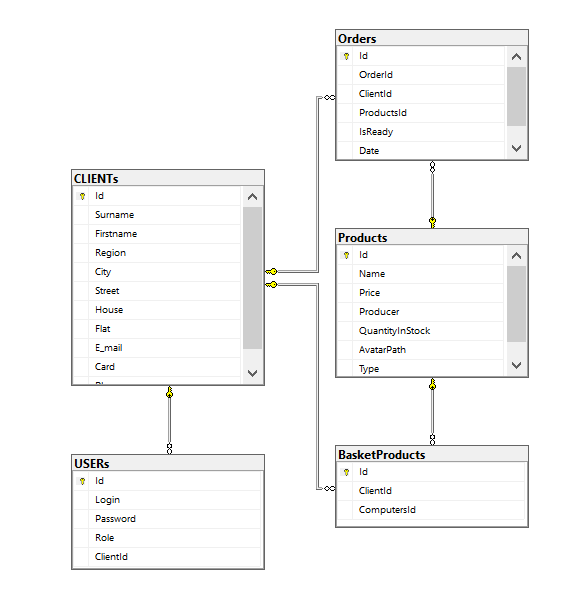


Рисунок 3.3. – Схема базы данных

Основной таблицей выступает таблица под названием СLIENTs – она содержит в себе личные данные о пользователе. Таблица СLIENTs содержит следующие столбцы:

* Id (int identity(1,1) primary key) – первичный ключ таблицы – Id пользователя;
* Surname (nvarchar(20)) –фамилия пользователя;
* Firstname (nvarchar(20)) – имя пользователя;
* Regiom (nvarchar(20)) – область проживания;
* City (nvarchar(20)) – город проживания;
* Street (nvarchar(20)) – улица проживания;
* House (nvarchar(20)) – дом проживания;
* Flat (nvarchar(3)) – квартира проживания;
* E\_mail (nvarchar(20)) – Email пользователя;
* Card (nvarchar(16)) – номер банковской карты;
* Phone (nvarchar(13)) – номер телефона.

С таблицей CLIENTs связана таблица USERs, которая хранит данные о паролях, логинах, а также роли пользователя в приложении (админ или клиент).

Включены следующие столбцы:

* Id (int identity(1,1) primary key) – первичный ключ таблицы – Id пользователя;
* Login (nvarchar(20)) – логин по которому пользователь может авторизоваться;
* Password (nvarchar(20)) – пароль которой пользователь вводит для того что бы авторизоваться(пароль хешированный);
* Role (nvarchar(20)) – роль, которая определяет кто пользователь(админ или клиент);
* ClientId(int) – внешний ключ таблицы – Id пользователя.

Products – содержит информацию о товарах. Включены следующие столбцы:

* Id (int identity (1,1)) – первичный ключ таблицы – Id товара;
* Name (nvarchar(20)) – наименование товара;
* Price (decimal(18,2)) – цена товара;
* Poducer (nvarchar(20)) – производитель товара;
* QuantitryInStock (int) – колличество на складе(штук);
* AvatarPath (nvarchar(MAX)) – путь к фотографии товара;
* Type (nvarchar(20)) –тип товара, нужен для сортировки товаров;

BasketProducts – содержит информацию о товарах, которые добавлены в карзину. Включены следующие столбцы:

* Id (int identity (1,1)) – первичный ключ таблицы – Id товара;
* ClientId(int) – внешний ключ таблицы – Id пользователя;
* ComputersId(int) – внешний ключ таблицы – Id продукта;

Orders – содержит информацию о заказах. Включены следующие столбцы:

* Id (int identity (1,1)) – первичный ключ таблицы ;
* OrderId(int) – Id заказа.
* ClientId(int) – внешний ключ таблицы – Id пользователя;
* ProductsId(int) – внешний ключ таблицы – Id продукта;
* IsReady(bit) – хранит значение, подтвержден ли заказ;
* Date(datetime) – хранит дату доставки заказа.

# 4 Разработка функциональной модели

## 4.1 Реализация авторизации и регистрации пользователя

Для того чтобы пользоваться приложением каждый пользователь должен войти в систему, предварительно зарегистрировавшись.

На рисунке 4.1 находится код, реализующий регистрацию.

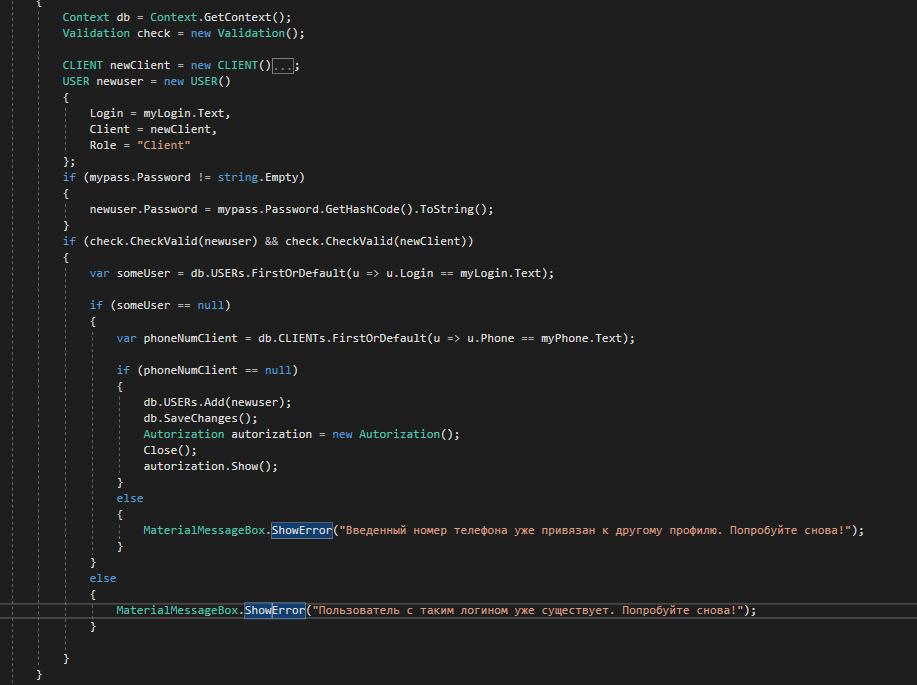


Рисунок 4.1. – Команда, реализующая регистрацию

В данном фрагменте кода присутствует валидация данных, для проверки корректности введенного email используется регулярное выражение. Для безопас­ности хранения информации используется хэширование, в данном случае применяется встроенный метод GetHashCode(). При регистрации пользователю задается тип аккаунта User, изменить этот статус может только администратор. Если все данные были введены верно и логин пользователя еще незанят, то осуществляется регистрация и переход на новую страницу.

Код обеспечивающий авторизацию приведен на рисунке 4.2.

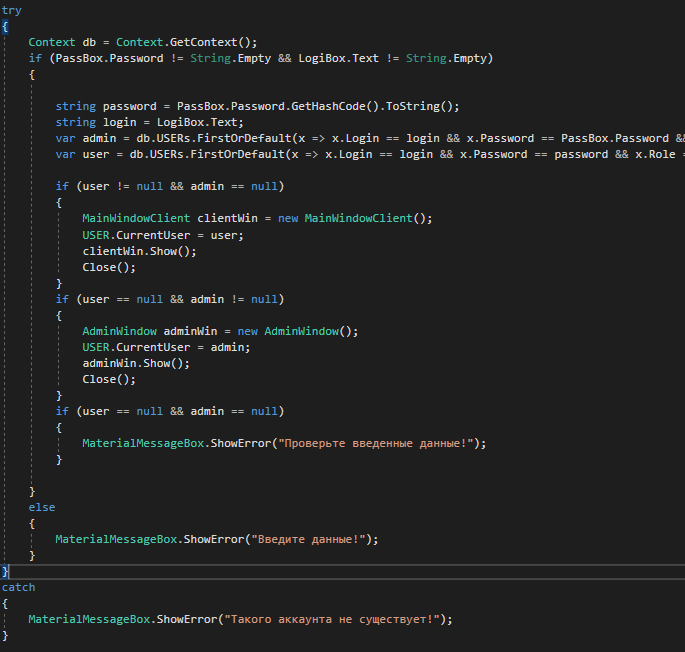


Рисунок 4.2. – Команда, реализующая авторизацию

Здесь проверяется тип аккаунта, в зависимости от которого будет открыто одно из окон: окно пользователя или окно администратора. При авторизации проверяется введены ли данные, существует ли пользователь с таким логином, а затем проверяется совпадение значений хэш-функций паролей. В случае неудовлетворения данных появляется окно с ошибкой. Если осуществляется вход, то в классе CurrentUser сохраняется данные текущего пользователя.

## 4.2 Реализация функцию добавления товара в корзину и оформления заказа.

Поскольку я использовал Page для удобного интерфейса. Я реализовал логику в Page используя события WPF.

Алгоритм добавление товара в корзину приведен на рисунке 4.3.

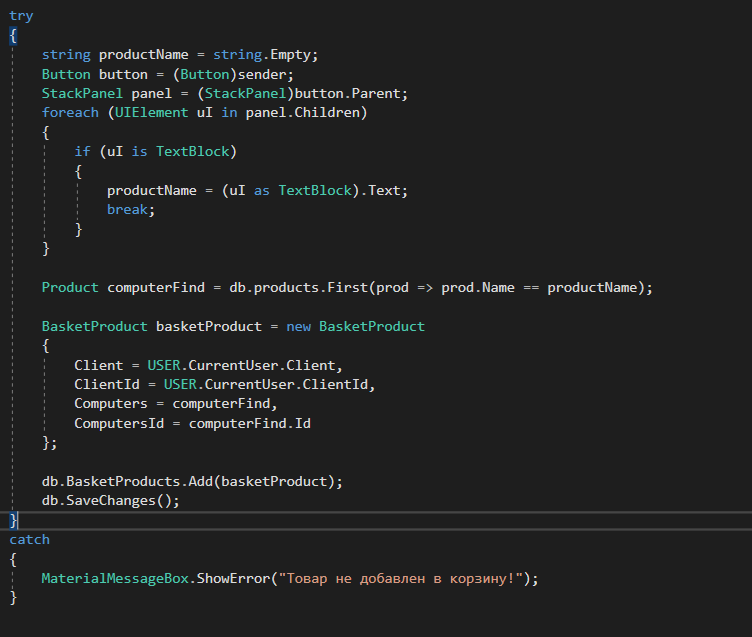


Рисунок 4.3. – Реализация добавление товара в корзину

На рисунке 4.4-4.5 реализованы обработчики событий в checkbox один из которых добавляет товар во временную коллекцию, а второй удаляет. Так же считается общая сумма временного заказа и количество товаров.

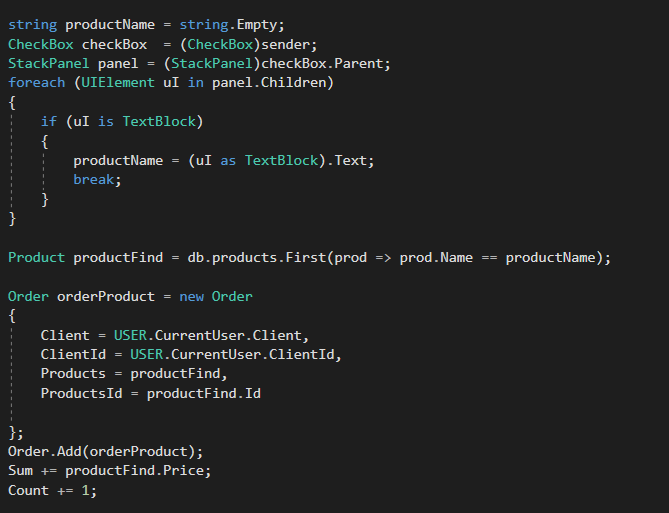


Рисунок 4.4. – Добавление товара во временную коллекцию.

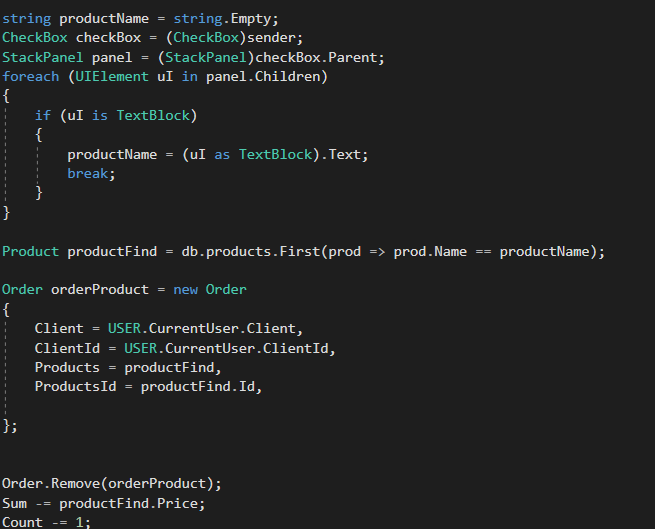


Рисунок 4.5. – удаление товара из временной коллекции.

Далее у меня есть кнопка оформления заказа, которая добавляет заказ в таблицу базы данных, с этим заказом потом будет работать администратор. А также после оформления заказа товары будут удаляться из корзины. Что бы оформить заказ пользователь должен обязательно заполнить поле с датой доставки. Так как в одном заказе может быть много товаров было решено добавить поле OrderId, которое будет заполнено случайным числом, что бы мы смогли отследить по номеру заказа все добавленные в таблицу Orders товары.

Добавление заказа в таблицу базы данных представлено на рисунке 4.6.

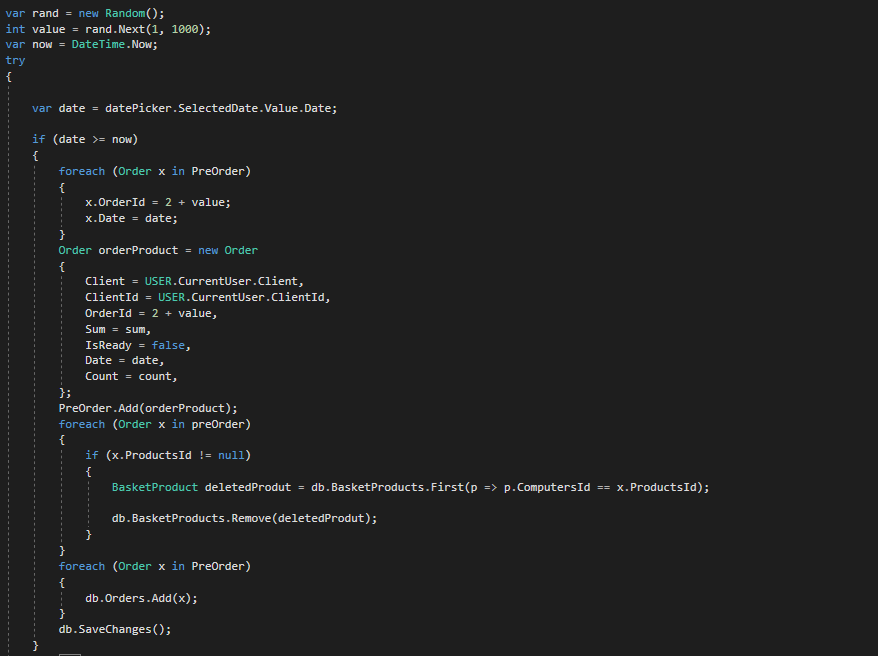


Рисунок 4.6. – Реализация добавления выбранных товаров в базу данных

## 4.3 Реализация функций работы администратора

Администратор может просматривать товары, добавлять, редактировать и удалять товары из базы данных. Администраторские функции были реализованы через MVVM. Для того что бы было удобнее просматривать информацию о товарах было решено добавить ComboBox. При выборе того или иного элемента которого выводиться из таблицы те или иные типы товаров. При добавлении в ту или иную категорию товаров, нужно было отследить какой Textblock активен, это было сделано с помощью свойства SelectedIndex. И в зависимости какой по TextBlock мы знаем какие товары выводить на экран, какие удалять, редактировать, добавлять. При добавлении товара мы добавляли картинку с помощью метода OpenFileDialog(). Который возвращает путь выбранной картинки. Этот путь мы и будем хранить в базе данных. Реализация добавления товара показана на рисунке 4.7.

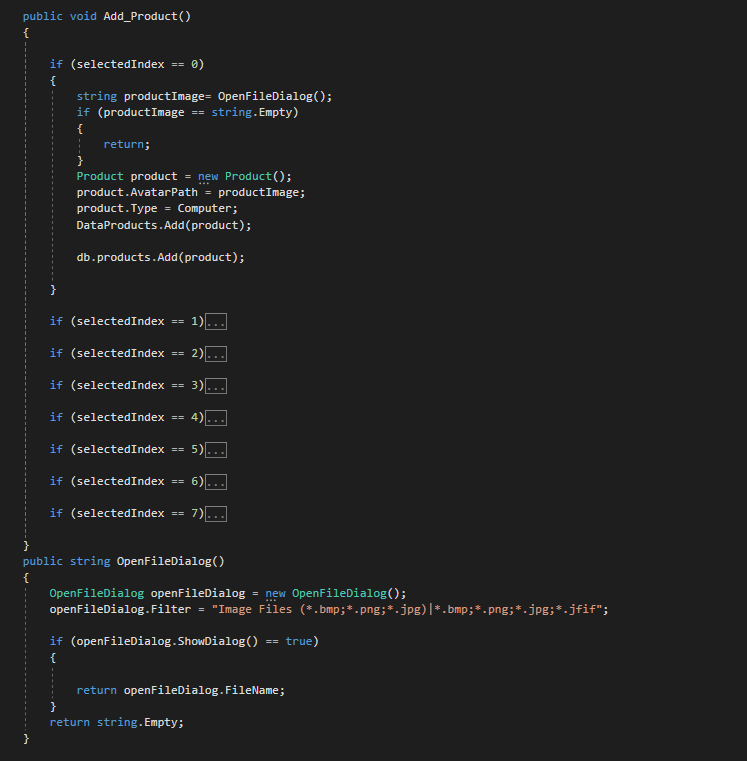


Рисунок 4.7. – Реализация добавления товара.

Реализация удаления товара представлена на рисунке 4.8

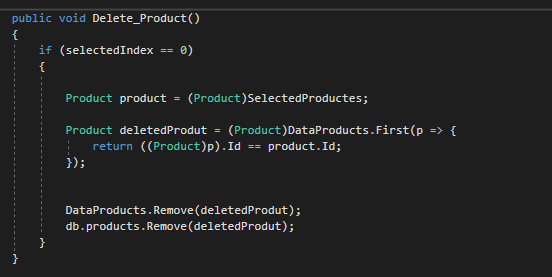


Рисунок 4.8. – Реализация удаление товара.

Редактирование происходит в datagrid, изменения автоматически отслеживаются. После того как мы отредактировали, добавили или удалили товар мы должны просто сохранить изменения используя метод SaveChanges().

Так же администратор может просматривать список клиентов, а так же галочкой в datagrid подтверждать выполнение заказа. В таблице заказов, которую видит администратор не указаны конкретные товары, для этого ему нужно выделить заказ, который он хочет посмотреть, и нажать кнопку “Показать заказ”. Для реализации этого, я отследил выделение строки в datagrid и вывел из базы данных все товары соответствующего заказа. Реализация кода просмотра конкретных товаров заказа, представлена на рисунке 4.9.

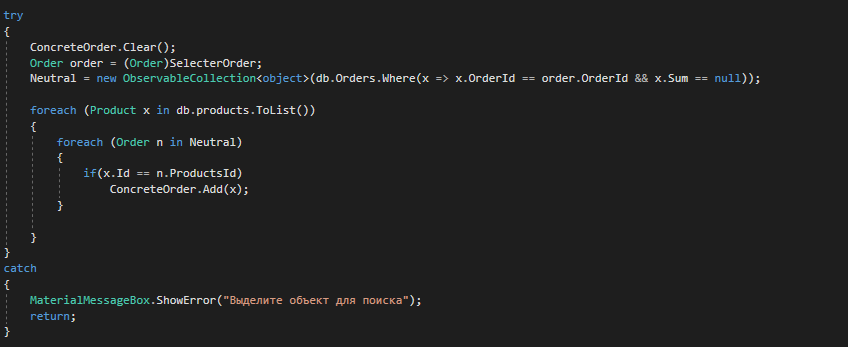


Рисунок 4.9. – Реализация просмотра товаров в заказе.

## 4.4 Функции для работы с базой данных

Работа с базой данных реализована через класс Context и реализацией паттерна Singleton.

На рисунке 4.10. приведен класс Context.



Рисунок 4.10. – Класс для работы с базой данных.

# 5 Тестирование

В курсовом проекте задействуется обработка исключений, таким образом, что пользователь будет уведомлен о неудачном выполнении запроса в базе дан­ных. Присутствуют различные всплывающие окна, выводящие сообщение об ошибке.

Ниже на картинках 5.1, 5.2, 5.3 представлены результаты обработки исключений при неверно введенных данных при авторизации и регистрации.

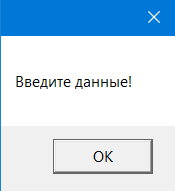


Рисунок 5.1 – Не введен логин и пароль

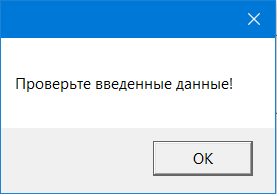


Рисунок 5.2 – Не верно введен логин или пароль

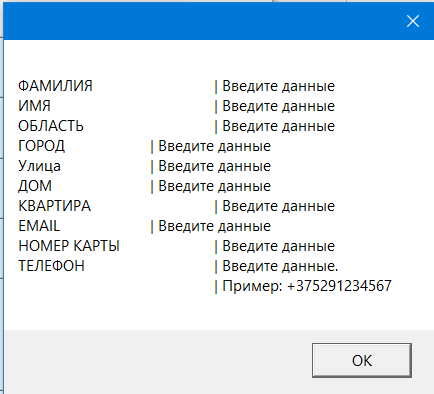


Рисунок 5.3 – Не верно введен логин или пароль

Во всех остальных окнах где присутствуют входные данные имеется такая же система обработки ошибок. Помимо этого, приложение постоянно уведомляет пользователя о событиях, произошедших в системе: удаления объекта, сохранение проекта, сохранение данных. Так же в случае случайного нажатия на кнопку уда­ления, появляется окно, которое проверяет действительно ли пользователь хочет удалить данные.

# 6 Руководство по использованию

## 6.1 Регистрация и авторизация

После запуска приложения пользователь видит главное окно приложения, на котором он может авторизоваться и получить возможность просматривать товары и совершать заказы, а если не авторизован то зарегестрироваться.

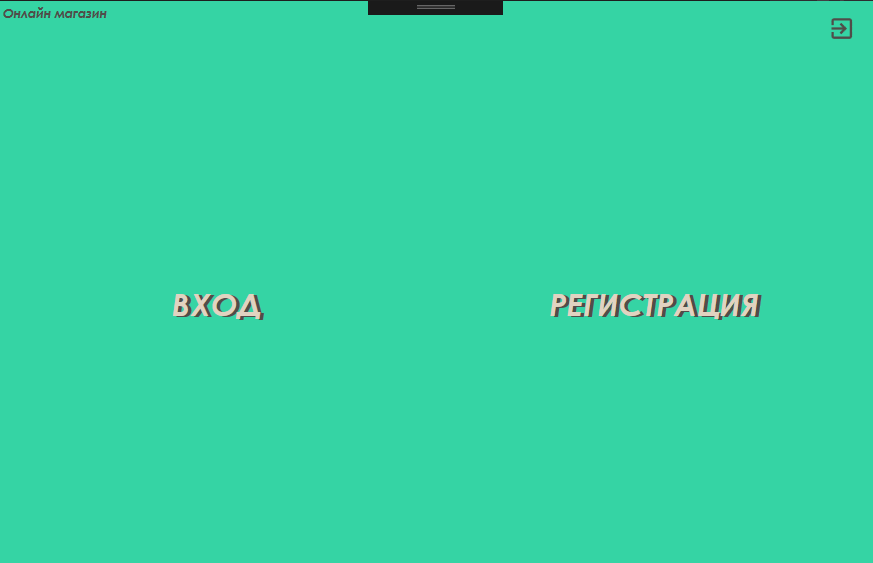


Рисунок 6.1. – Начальное окно.

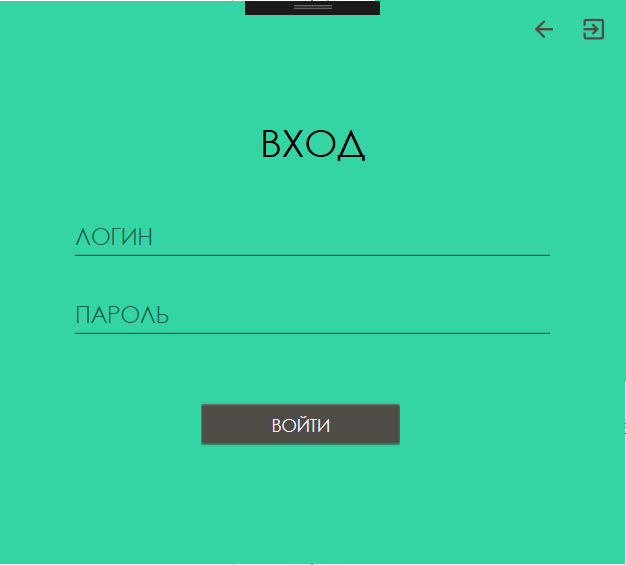


Рисунок 6.2. – Окно авторизации.

Если нет аккаунта, клиент может зарегистрироваться введя некоторые данные и придумая логин и пароль.

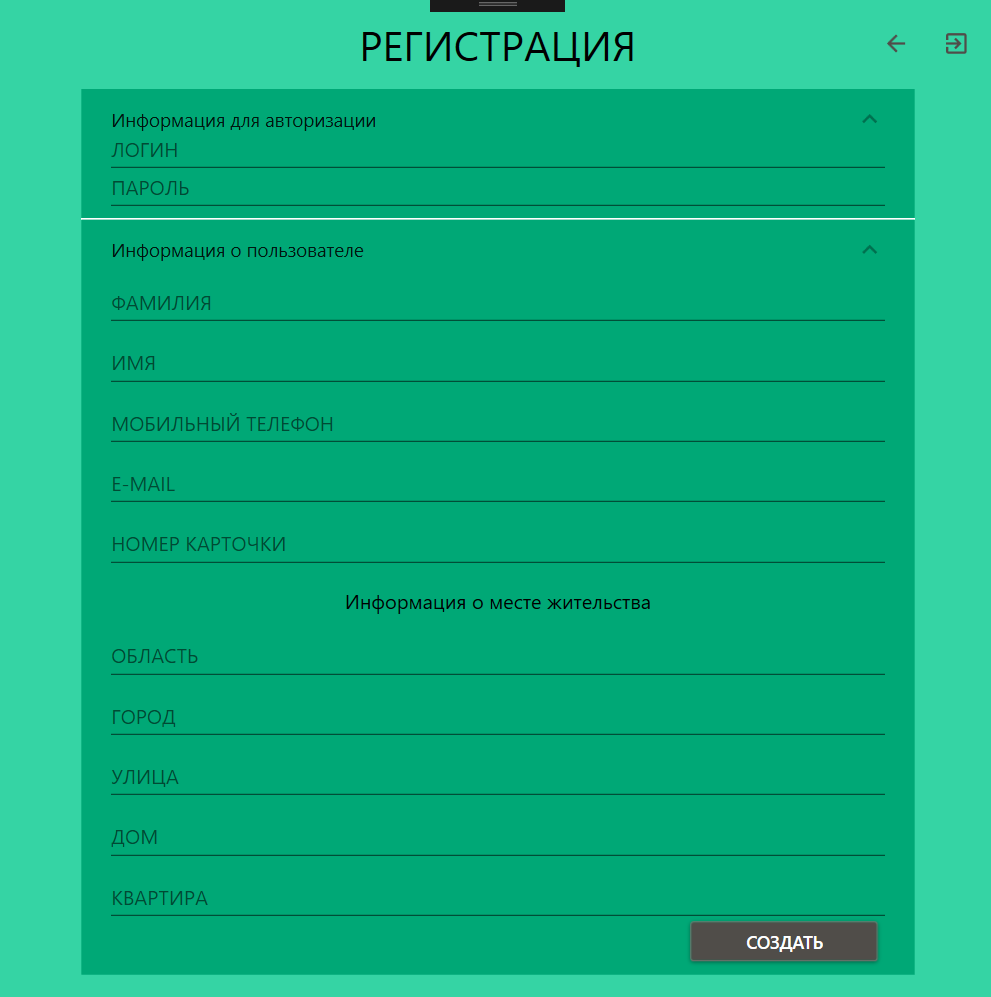


Рисунок 6.3. – Окно регистрации.

## 6.2 Использование приложения обычным пользователем

В случае успешного входа пользователь переходит на главную страницу приложения, в котором отображаются все товары. Слева находится меню в котором располагаются все категории товаров для более удобного просмотра и поиска товаров. По умолчанию после авторизации пользователь сразу находится на странице Главная. Главное стартовое окно, которое запускается после авторизации приведено на рисунке 6.4.

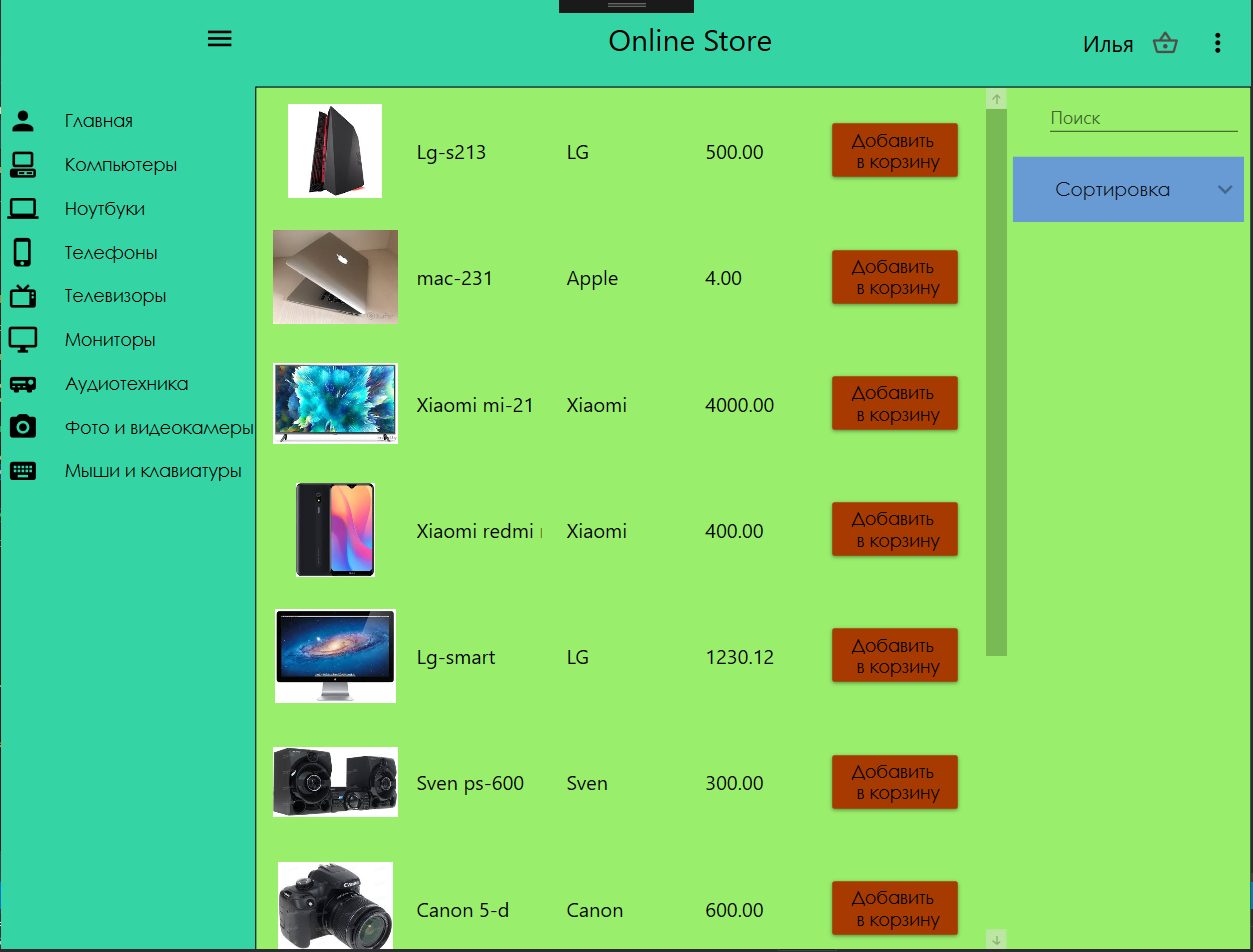


Рисунок 6.4. – Главное окно пользователя

При нажатии на кнопку мой профиль пользователь переходит в окно, которое показано на рисунке 6.5. В этом окне можно поменять информацию введенную при регистрации.

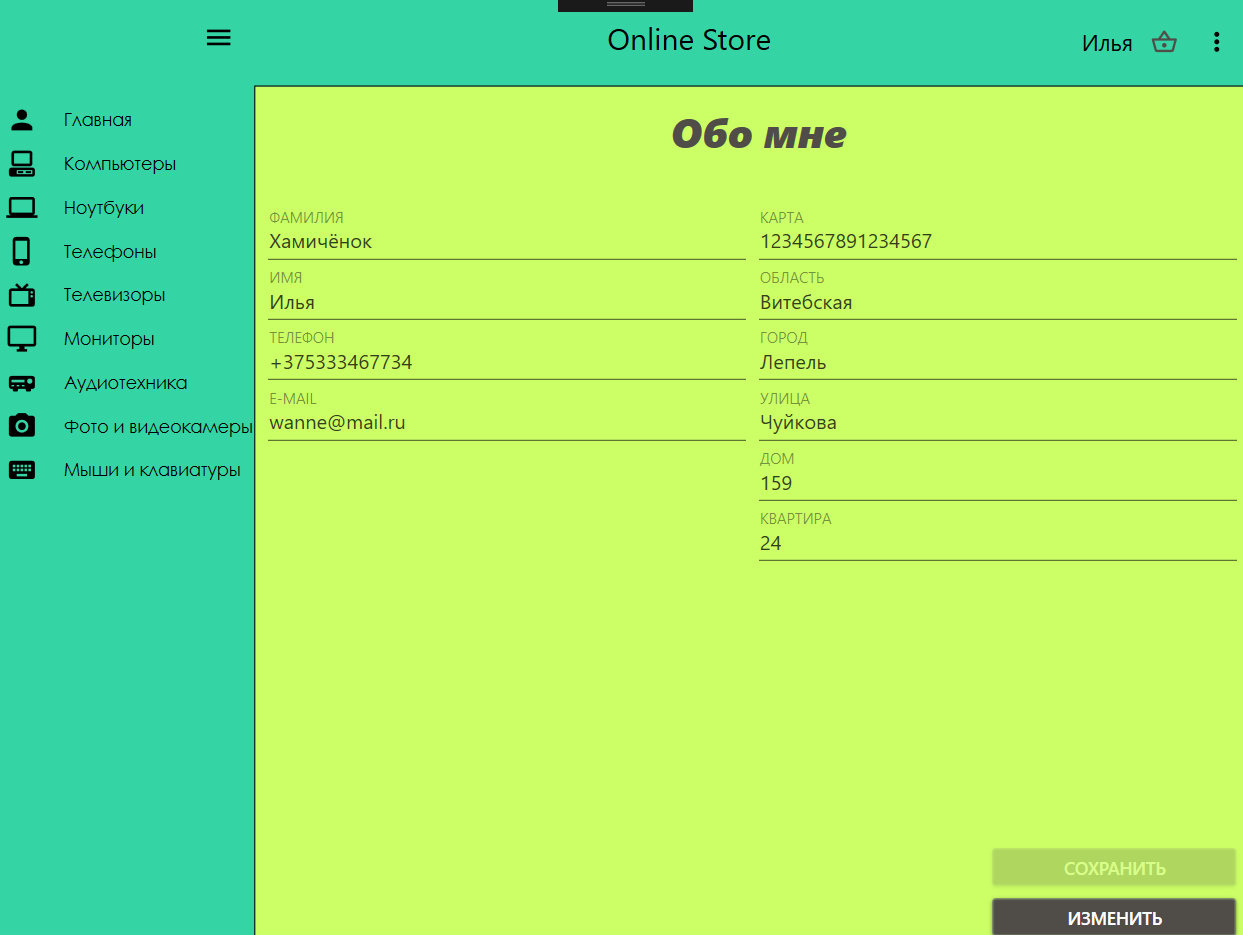


Рисунок 6.7. – Окно просмотра профиля.

При нажатии на иконку корзины, пользователь попадет на страницу, в которой буду отображаться товары, которые пользователь хочет заказать. Так же пользователь может выбирать галочками товары из корзины, которые он хочет заказать, выбирать дату поставки и оформлять заказ. Корзина представлена на рисунке 6.8.

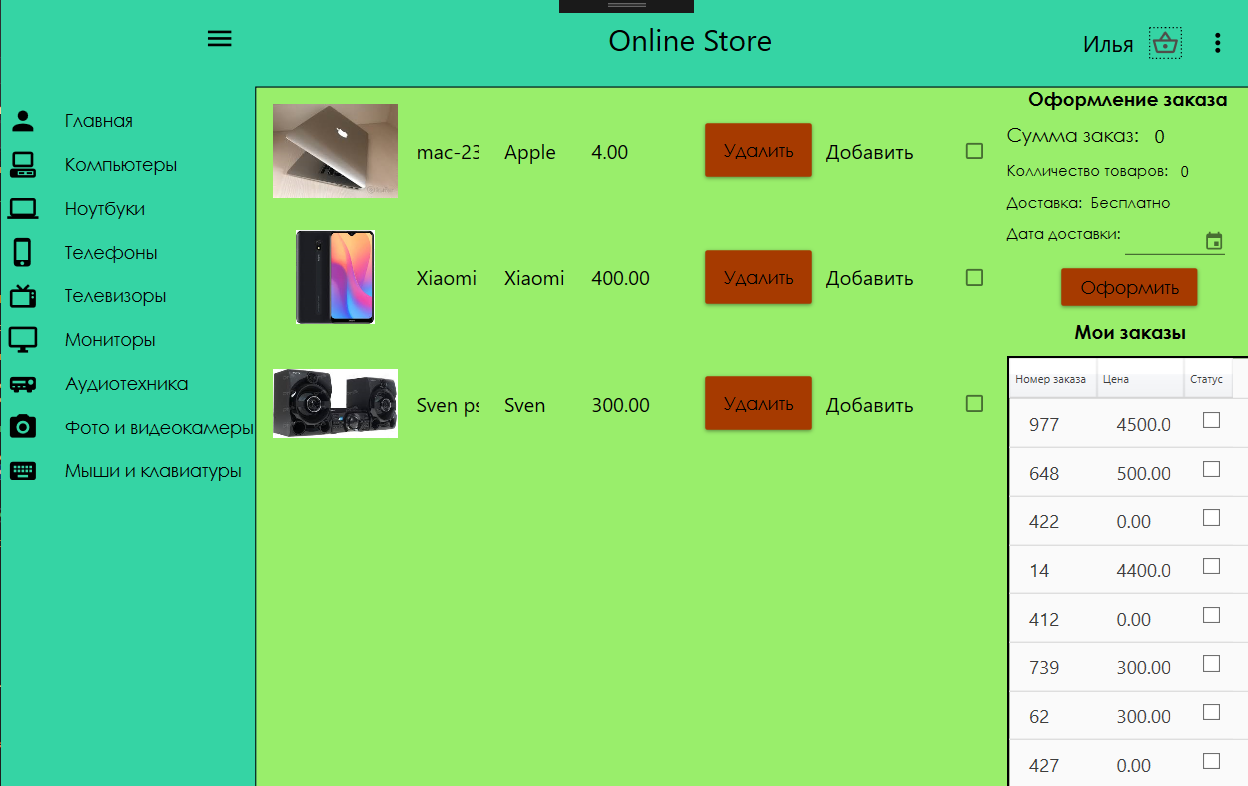


Рисунок 6.8. – Окно корзины товаров.

## 6.3 Использование приложения администратором

Если при авторизации пользователя, данные введены верно и статус акка­унта соответствует администратору то пользователь попадает на другое главное окно, которое обладает совершенно другим функционалом по сравнению с окном доступным обычным пользователям.

Пользователь со статусом администратор попадает на окно AdminWindow, которое показано на рисунке 6.9.

На этой странице администратор осуществляет работу с товарами и клиентами, находящимися в базе данных, он может посмотреть всех пользователей, а также заказы которые оформили те или иные пользователи. Может добавлять, удалять, редактировать товары в базе данных. Интерфейс очень простой, переходя по вкладкам администратор может быстро перемещаться по предметной области и решать поставленные задачи в кратчайший срок.

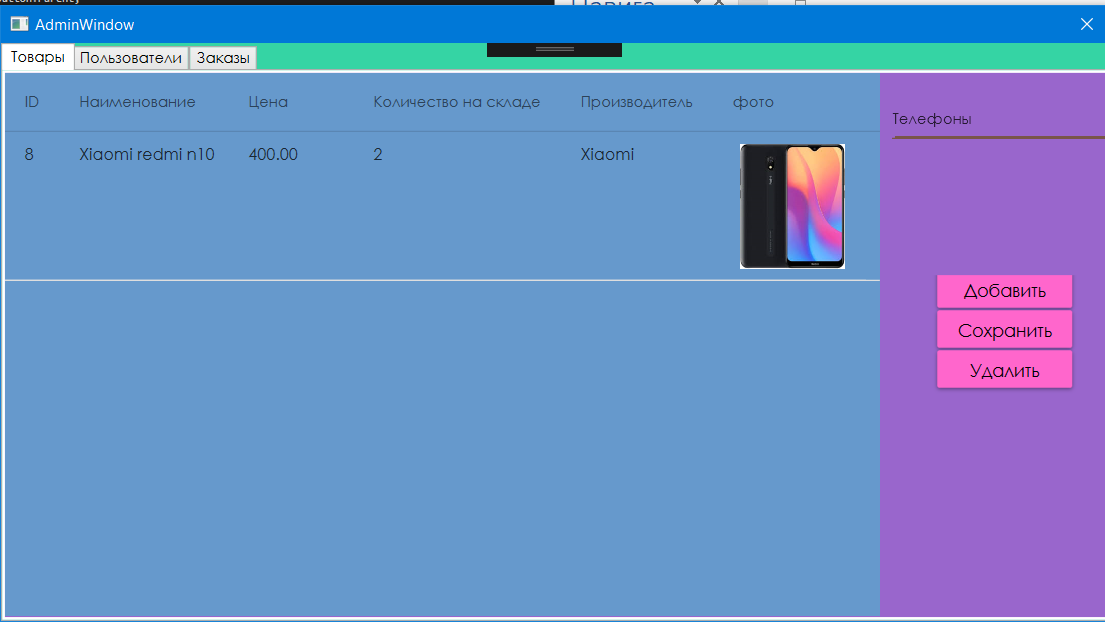


Рисунок 6.9. – Главное окно администратора.

Администратор также может легко перейти на вкладку пользователи, где может увидеть список всех зарегистрированных пользователей. Вкладка пользователи представлена на рисунке 6.10.

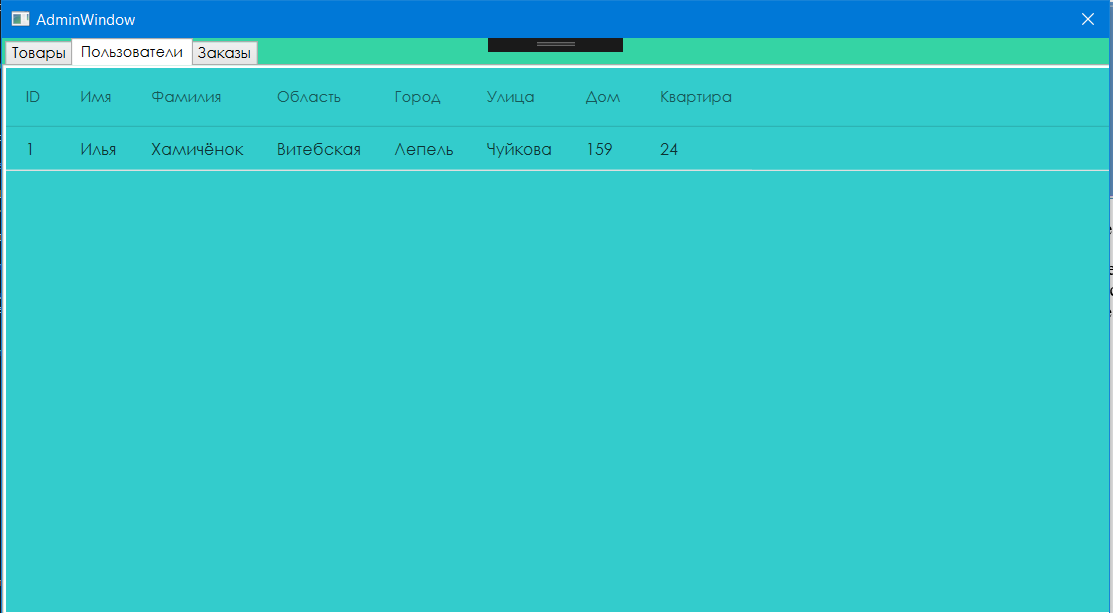


Рисунок 6.10. – Вкладка пользователи.

И последняя функция администратора, это просмотр и подтверждения заказов, которые совершили клиенты. Вкладка заказы представлена на рисунке 6.11.



Рисунок 6.10. – Вкладка Заказы.

**ЗАКЛЮЧЕНИЕ**

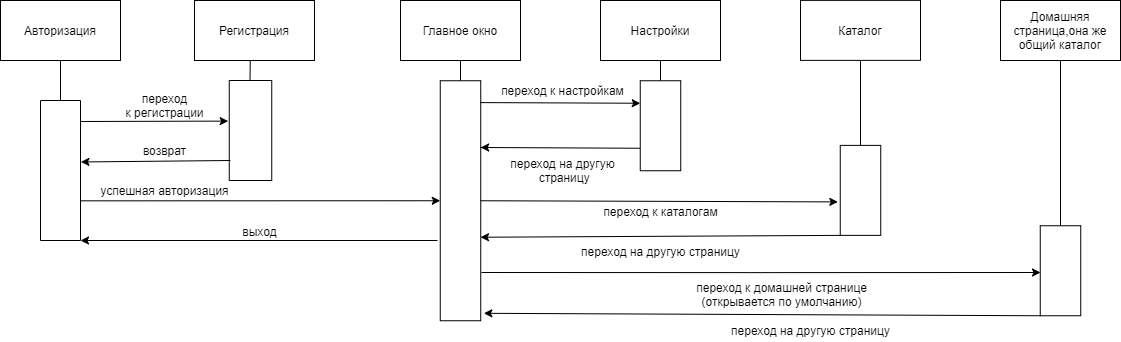
В итоге выполнения данного курсового проекта было разработано программное средство «Интернет магазин», представляющий возможность просмотра каталога товаров, а также оформление заказов. Достоинства этого приложения – это простота дизайна, юзабильность, возможность постоянно пополнять каталог товаров и оформлять заказы, от лица администратора.

В процессе реализации программного средства были достигнуты выше указанные требования и задачи.Также в процессе выполнения данного курсового проекта были закреплены навыки в программировании на языке C#, созданни приложений на WPF, использование Entity Framework 6, работа с современным паттерном MVVM, проектирование базы данных и реализация их в СУБД MS SQL Server 2018.

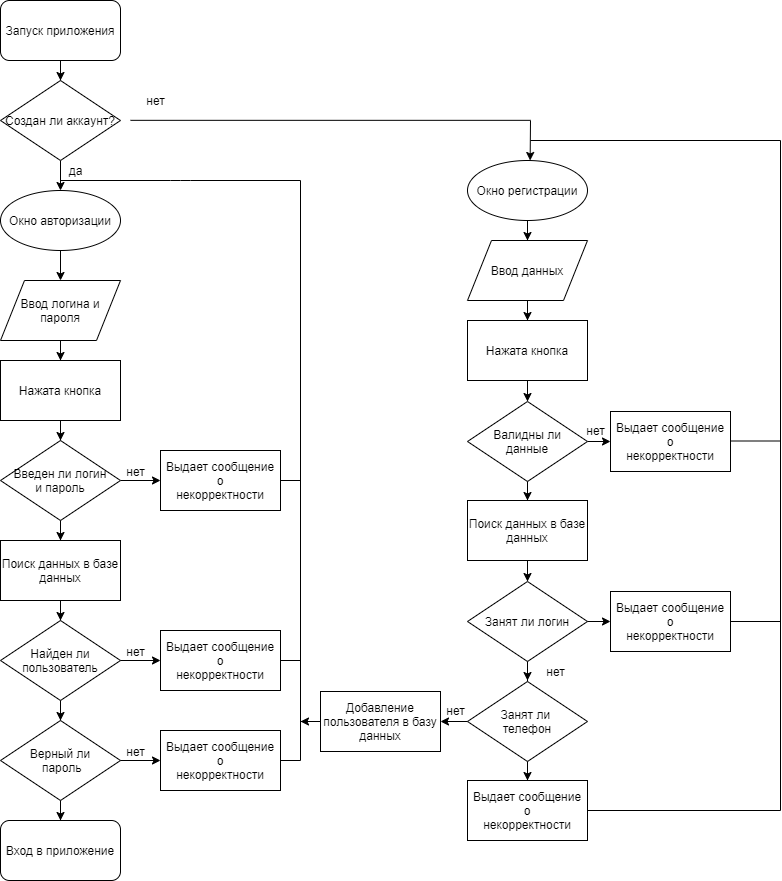
**СПИСОК ИСПОЛЬЗУЕМОЙ ЛИТЕРАТУРЫ**

1. MSDN сеть разработчиков в Microsoft [Электронный ресурс] – <http://msdn.microsoft.com/library/rus/>. Дата доступа: 10.05.2018
2. WPF | Определение паттерна MVVM [Электронный ресурс] – <https://metanit.com/sharp/wpf/22.1.php>. Дата доступа: 18.10.2016
3. Пацей, Н.В. Курс лекций по языку программирования С# / Н.В. Пацей. – Минск: БГТУ, 2016. – 175 с.
4. Generic Repository | Entity Framework 6 [Электронный ресурс] – <https://metanit.com/sharp/entityframework/3.13.php>. Дата доступа: 13.10.2016
5. ProfessorWeb .NET & Web Programming [Электронный ресурс] / Режим доступа: <https://professorweb.ru> Дата доступа: 13.04.2018
6. Рихтер Дж. CLR via C#. Программирование на платформе Microsoft .NET Framework 4.5 на языке C#. 4-е изд. — СПб.: Питер, 2013. — 896 с.: ил. — (Серия «Мастер-класс»).

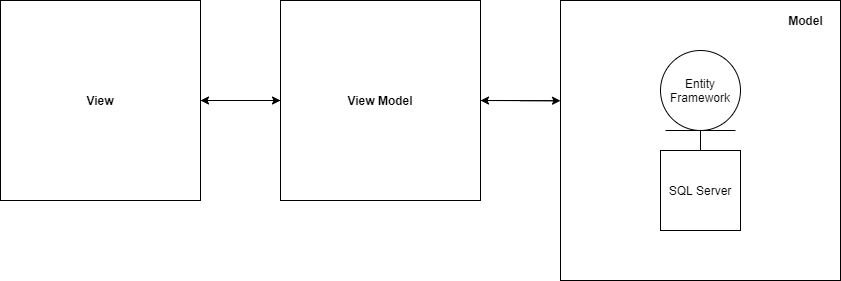
**ПРИЛОЖЕНИЕ А**



**ПРИЛОЖЕНИЕ Б**

****

**ПРИЛОЖЕНИЕ В**

****

**ПРИЛОЖЕНИЕ Г**

