МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

Учреждение образования «БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»

Факультет Информационных технологий

Кафедра Программной инженерии

Специальность 1-40 05 01-03 Информационные системы и технологии

Направление специальности 1-40 01 01 10 Программное обеспечение информационных технологий (программирование интернет-приложений)

**ПОЯСНИТЕЛЬНАЯ ЗАПИСКА**

**КУРСОВОГО ПРОЕКТА:**

по дисциплине «Объектно-ориентированное программирование»

Тема Программное средство «Ювелирная мастерская»

Исполнитель

студентка 2 курса группы 1 Потапейко Полина Павловна

Руководитель работы ст. препод. Сухорукова И. Г.

Курсовой проект защищён с оценкой

Председатель Пацей Н.В.

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 05 01-03 Информационные системы и технологии  Студент: Потапейко П. П. | Группа:\_\_1\_\_ |
| **Тема:** Программное средство «Ювелирная мастерская» | |

**1. Срок сдачи студентом законченной работы:** "20 мая 2021 г."

**2. Исходные данные к проекту:**

**2.1**. Функционально ПС поддерживает:

* Функции администратора сервиса:
  + Поддерживать работу c базой данных;
  + Управлять статусом заказа;
  + Оповещать клиента о выполнении заказа (почта, sms).
* Функции клиента:
  + Выполнять регистрацию и авторизацию;
  + Заполнять форму заказа по заданным формам;
  + Загружать изображения;
  + Просматривать информацию о статусе заказа.

**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 | 2.04.2021 |  |
| 5 | Кодирование программного средства | 23.04.2021 |  |
| 6 | Тестирования и отладка программного средства | 30.04.2021 |  |
| 7 | Оформление пояснительной записки | 7.05.2021 |  |
| 9 | Сдача проекта | 20.05.2021 |  |

**5. Дата выдачи задания** \_\_\_\_12.02.2021\_\_\_\_

Руководитель\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ *И. Г. Сухорукова*

(подпись)

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

(дата и подпись студента)

Cодержание

[Введение 3](#_Toc73067772)

[1. Анализ прототипов, литературных источников и формирование требований к проектируемому программному средству 5](#_Toc73067773)

[1.1. Сайт ювелирной мастерской https://runuillirun.ru/ 5](#_Toc73067774)

[1.2. Сайт ювелирной мастерской https://davinci.by/ 7](#_Toc73067775)

[2. Анализ и проектирование архитектуры приложения (построение диаграмм, проектирование бизнес-слоя, представления и данных) 9](#_Toc73067776)

[2.1. UML Use Case Diagram 9](#_Toc73067777)

[2.2. UML Class Diagram 10](#_Toc73067778)

[2.3. UML Sequence Diagram 11](#_Toc73067779)

[2.4. Блок-схемы алгоритмов 12](#_Toc73067780)

[3. Проектирование структуры базы данных. Разработка дизайна пользовательского интерфейса. 14](#_Toc73067781)

[3.1. Код Entity Framework Core 14](#_Toc73067782)

[3.2. Схема базы данных 14](#_Toc73067783)

[3.3. Интерфейс приложения 15](#_Toc73067784)

[4. Создание программного средства 20](#_Toc73067785)

[4.1. Уровень бизнес-логики 20](#_Toc73067786)

[4.2. Уровень представлений 20](#_Toc73067787)

[4.3. Уровень модели представлений 21](#_Toc73067788)

[5. Тестирования и отладка программного средства 23](#_Toc73067789)

[6. Руководство по установке и использованию 27](#_Toc73067790)

[Заключение 30](#_Toc73067791)

[Приложение A 31](#_Toc73067792)

[Приложение Б 32](#_Toc73067793)

[Список использованных источников 34](#_Toc73067794)

Введение

Со времен древности и по наши дни люди стремятся украсить свое жилье, а также самих себя. Этой цели служат разнообразные произведения искусства, в частности ювелирные украшения. Ювелирные украшения создаются мастерами ювелирного дела, и их качество зависит в большей степени от умения самого мастера.

Приобретать ювелирные изделия можно в специальных магазинах, но с развитием интернета это можно сделать и онлайн, однако все еще ощущается недостаток приложений данной тематики.

Целью данного курсового проекта является разработка программного средства для заказа и приобретения ювелирных изделий, а также оформления заявки на ремонт изделия.

Данное программное средство позволяет пользователям заказать интересующие ювелирные украшения и оформить заявку на ремонт изделия, а также мастерам контролировать статус заказа. Также программное средство должно предоставлять пользователю информацию о заказе.

При проектировании используются принципы объектно-ориентированного программирования и паттерны проектирования.

Пояснительная записка включает в себя следующие разделы:

**Введение** содержит краткий анализ темы проектирования, описание цели курсовой работы, принципов, использованных при разработке программного средства и описание разделов пояснительной записки

**Первый раздел** содержит обзор литературы и примеры решения аналогичных задач, анализ достоинств и недостатков решений, на основе анализа формулируются задачи и требования к программному средству.

**Второй раздел** содержит словесное описание и описание с помощью диаграмм и схем структуры программного средства.

**Третий раздел** содержит описание структуры базы данных, используемых серверной частью программного средства, а также описание и иллюстрации интерфейса программного средства.

**Четвёртый раздел** содержит описание сборки ПС, описание процедур, функций модулей.

**Пятый раздел** содержит описание и результаты тестирования для проверки работоспособности программного средства.

**Заключение** содержит результаты, характеризующие полноту и подытоживающие содержание курсового проекта, рекомендации и перспективы дальнейшей разработки и использования программного средства.

**Список используемых источников** содержит список использованной в курсовой работе литературы и интернет-ресурсов.

**Приложения** содержат листинги алгоритмов и программы, иллюстрации вспомогательного характера и т.д.

# Анализ прототипов, литературных источников и формирование требований к проектируемому программному средству

Был проведен поиск и анализ аналогов и литературных источников по задачам, схожим с задачами проектируемого программного средства.

## Сайт ювелирной мастерской <https://runuillirun.ru/>

В качестве первого аналога был выбран данный сайт. Для посетителей на главной странице представлены возможные для заказа ювелирные изделия (рис. 1.1.)

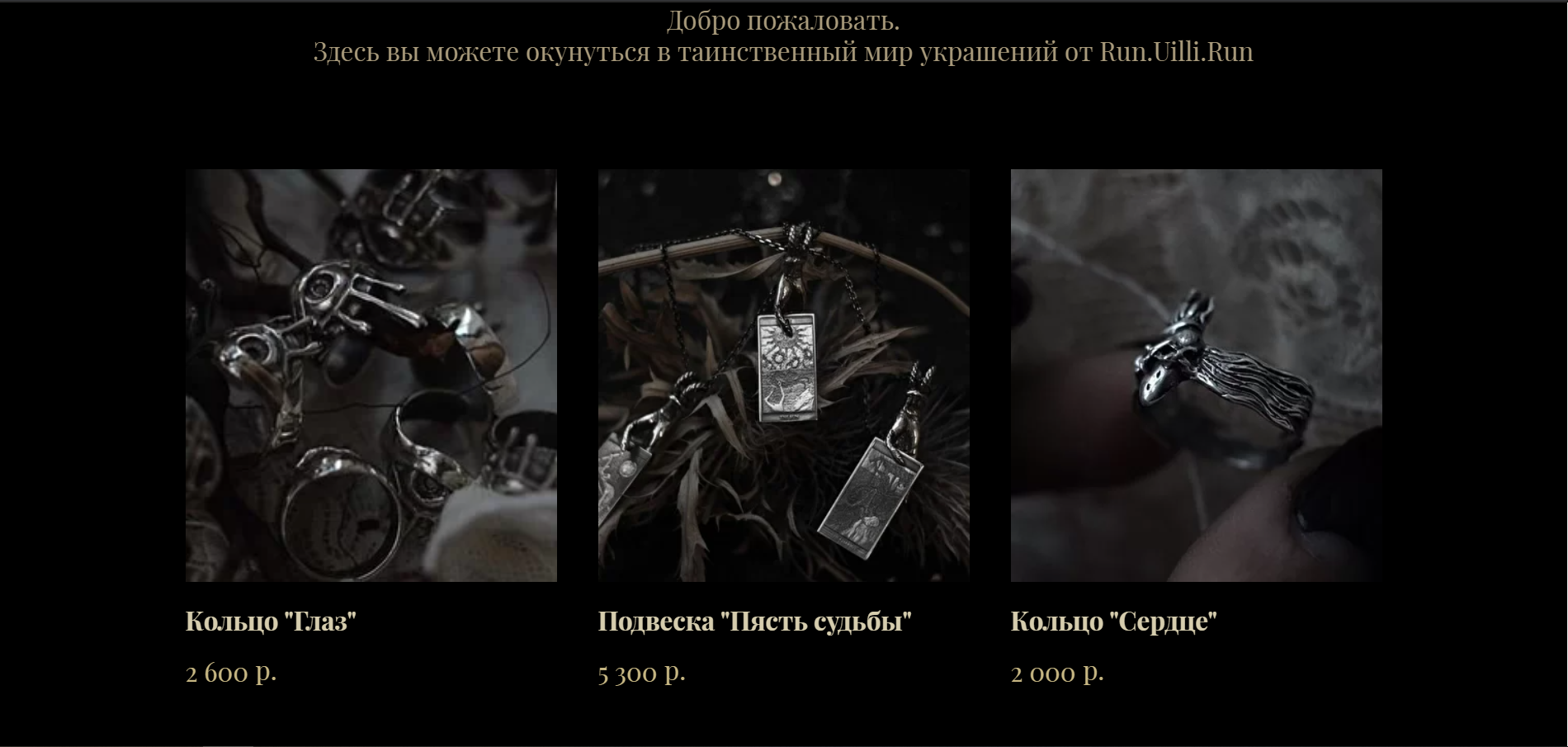


Рисунок 1.1 – Ассортимент товаров ювелирной мастерской

После выбора интересующего изделия появляется окно с информацией об этом изделии с возможностью выбора его характеристик для нужд конкретного покупателя (рис.1.2.)

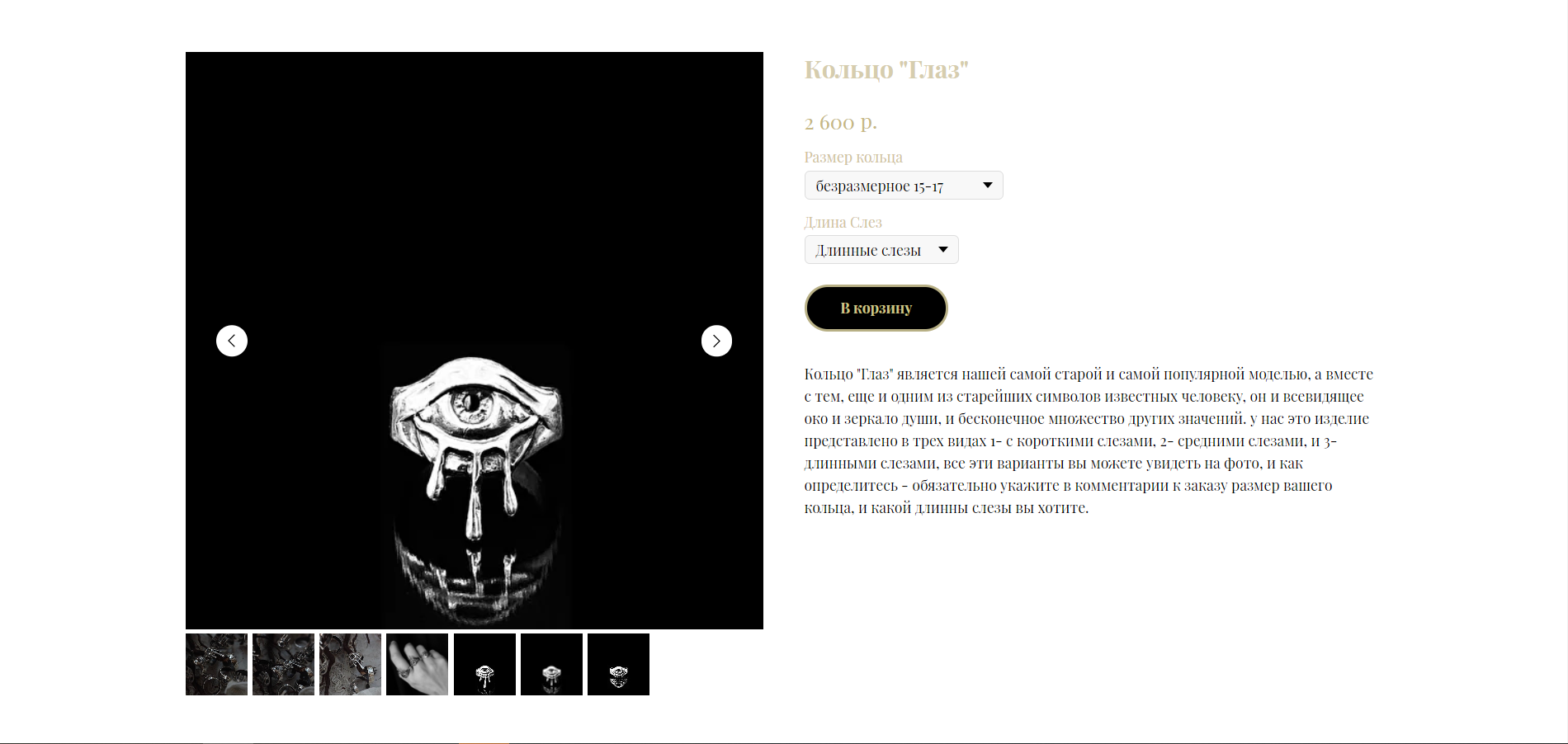


Рисунок 1.2 – Информация об изделии

Далее по нажатии на кнопку добавления изделия в корзину для покупки открывается форма ввода данных клиента: ФИО, e-mail, номер телефона, комментарий к заказу, а также промокод и способ доставки (рис 1.3.)



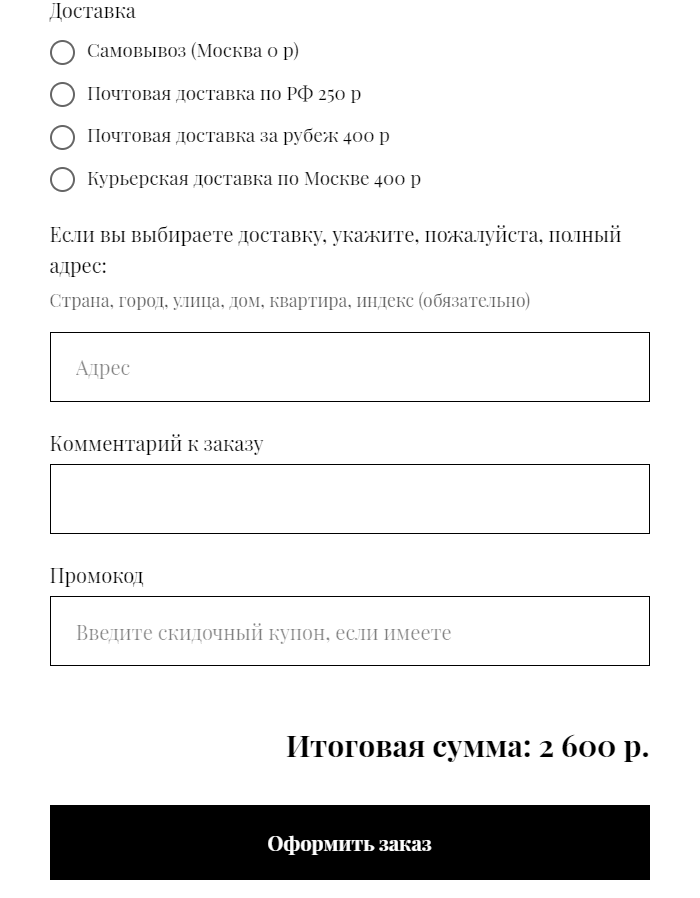


Рисунок 1.3 – Форма заполнения заказа

Сайт имеет хороший дизайн, однако отсутствует возможность оформления заявки на ремонт изделия.

## Сайт ювелирной мастерской <https://davinci.by/>

Интерфейс сайта davinci.by в сравнении с предыдущим аналогом проигрывает. На сайте можно выбрать услугу на любой вкус, но нельзя оформить заявку ни на одну из услуг. Также нельзя заказать изделие прямо на сайте. В общем можно сказать, что на сайте представлена информация для ознакомления. Интерфейс сайта представлен на рисунках 1.4 - 1.6.



Рисунок 1.4 – Главная страница сайта davinci.by

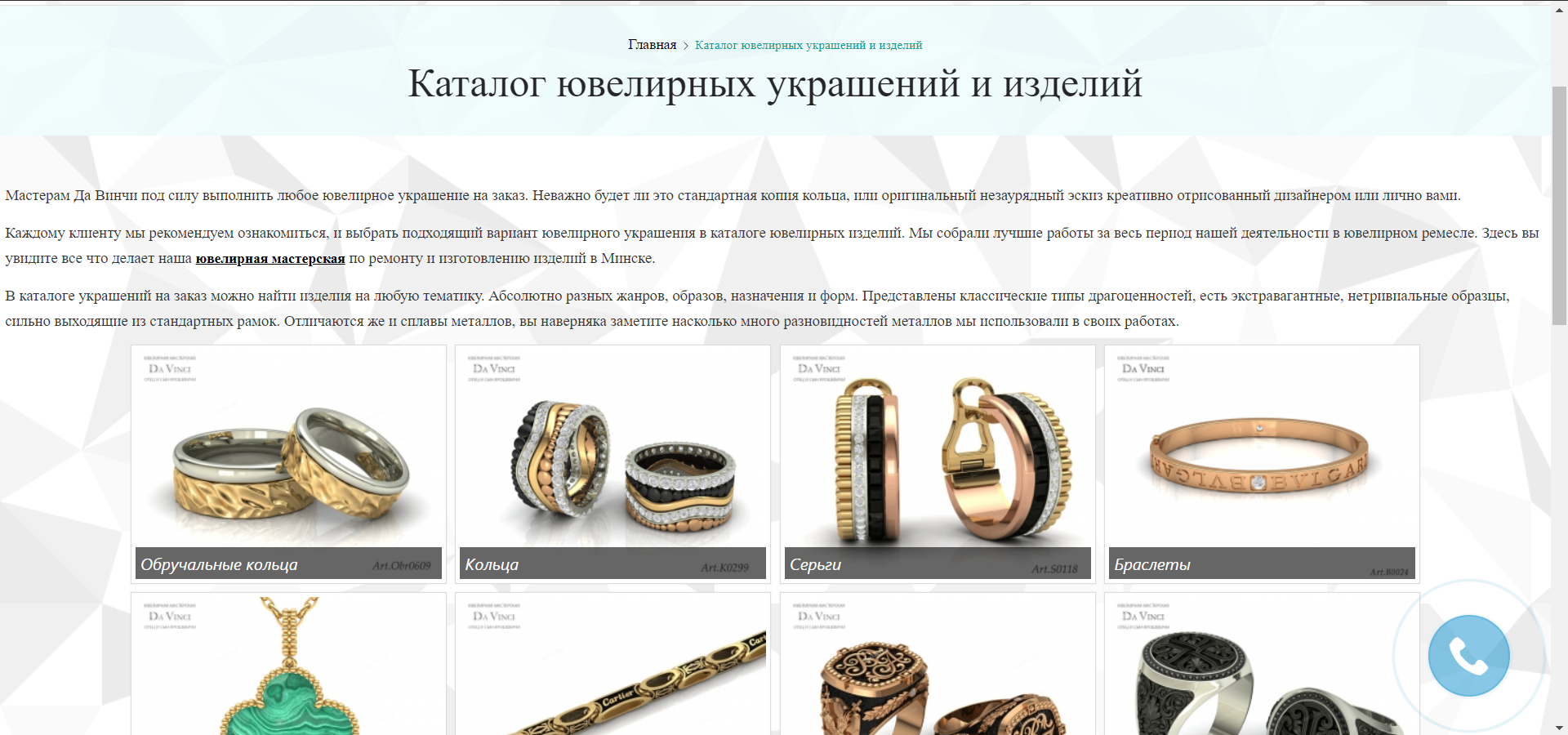


Рисунок 1.5 – Страница каталога сайта davinci.by

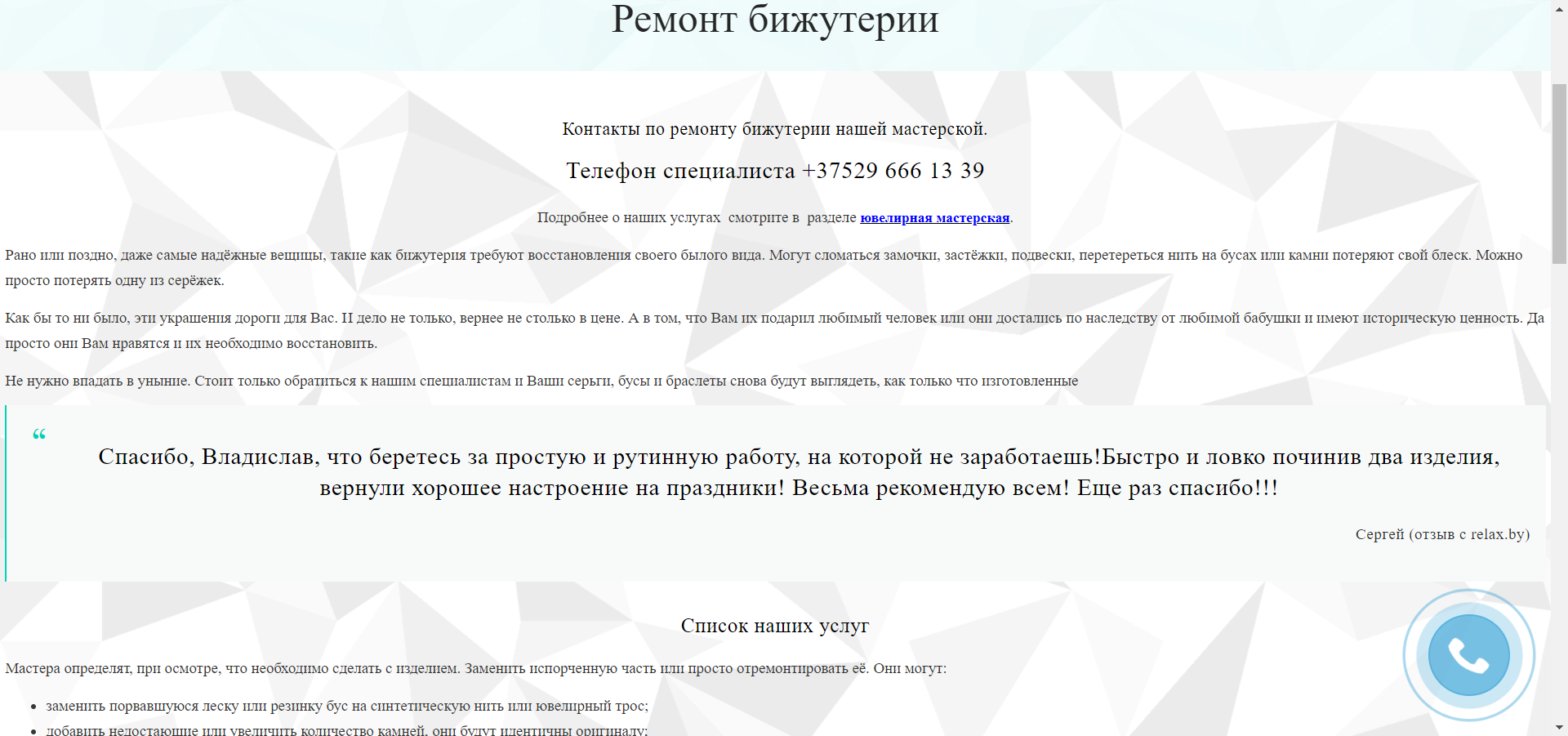


Рисунок 1.6 – Страница каталога сайта davinci.by

На основе анализа был сформирован следующий список требований к проектируемому средству:

* Пользователь должен регистрироваться перед первым использованием приложения и авторизоваться впоследствии.
* Функционал программного средства разделен в зависимости от типа пользователя (администратор и клиент).
* Администратор управляет ассортиментом товаров и может менять статус заказа.
* Клиент может просматривать ассортимент, создавать заказы на изделие и ремонт.
* При изменении статуса заказа клиент должен быть уведомлен об этом по электронной почте.

# Анализ и проектирование архитектуры приложения (построение диаграмм, проектирование бизнес-слоя, представления и данных)

Перед проектированием программного средства нужно определить особенности его использования и структуры. Для этих целей удобно использовать различные схемы и диаграммы.

## UML Use Case Diagram

UML Use Case диаграмма позволяет показать варианты использования программного средства различными пользователями. На диаграмме на рисунке 2.1 показаны варианты использования администратором и клиентом.

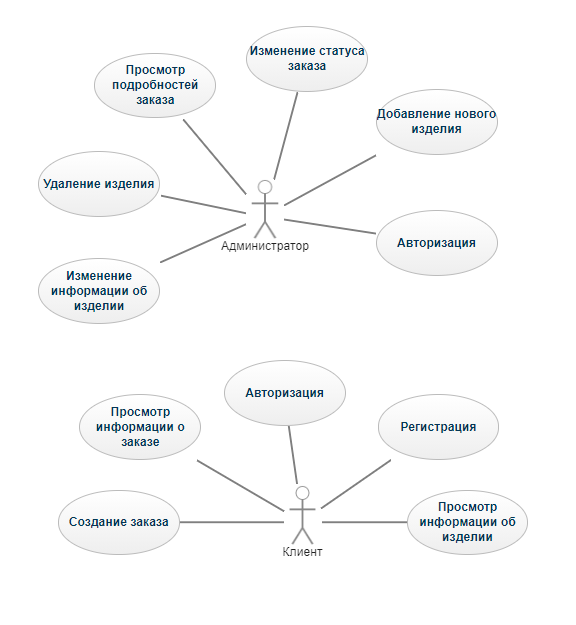


Рисунок 2.1 – Диаграмма UML Use Case проектируемого средства

## UML Class Diagram

Диаграмма классов позволяет визуально отобразить структуру классов и взаимосвязи между ними. Диаграммы классов представлены на рисунках 2.2 - 2.4.

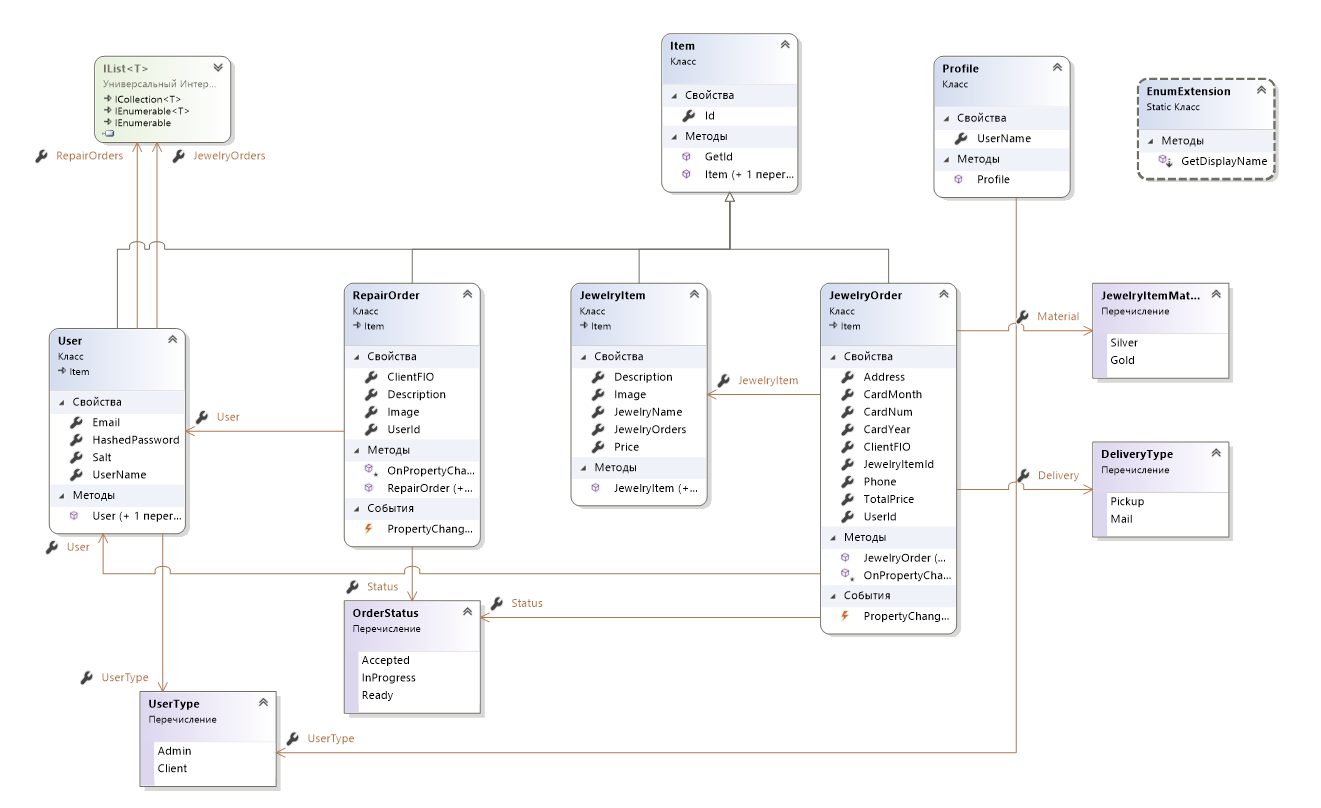


Рисунок 2.2 – Диаграмма классов бизнес-логики

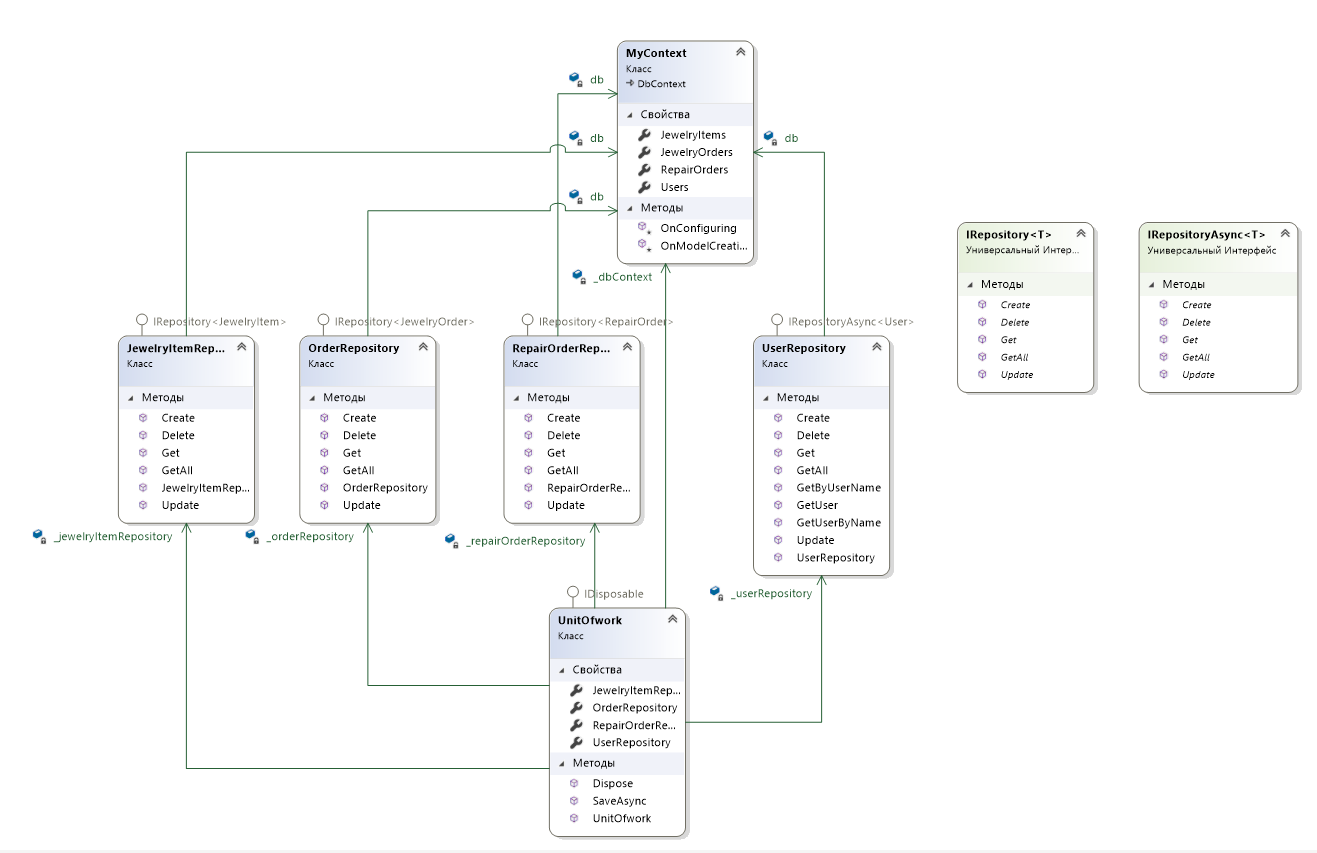


Рисунок 2.3 – Диаграмма классов для работы с базой данных

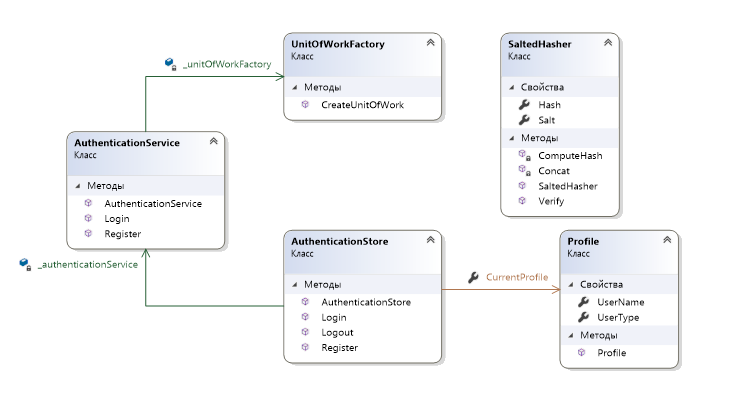


Рисунок 2.4 – Диаграмма классов для работы пользователей

## UML Sequence Diagram

Диаграмма последовательностей позволяет пронаблюдать за последовательностью взаимодействий объектов внутри приложения во времени. Диаграмма взаимодействия программного средства и пользователя со стороны клиента представлена на рисунке 2.5.

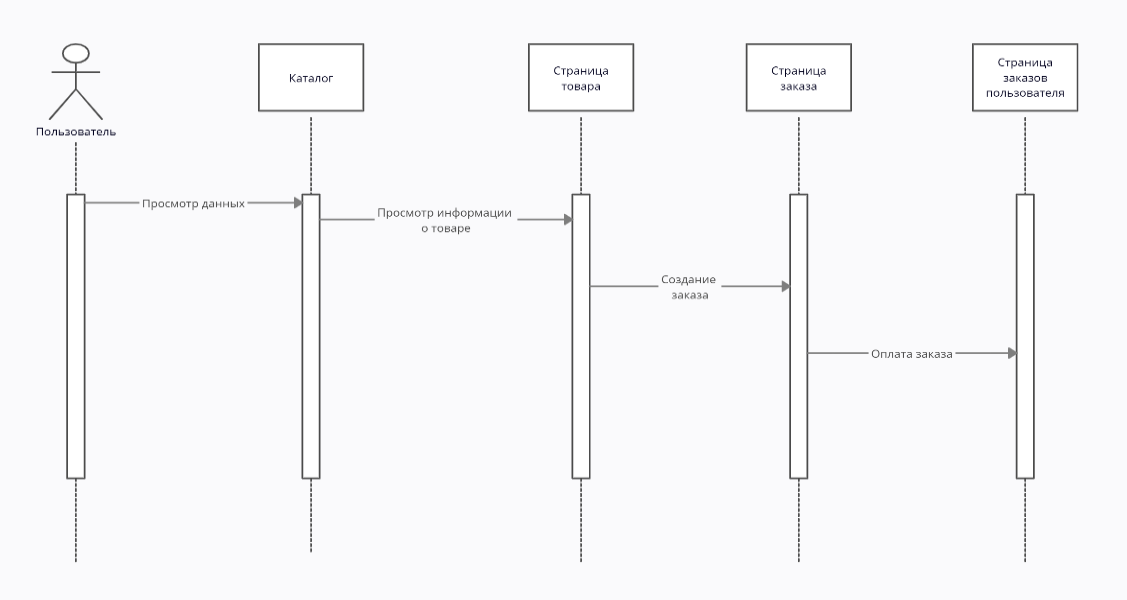


Рисунок 2.5 – Диаграмма UML Sequence

## Блок-схемы алгоритмов

Блок-схемы позволяют изобразить действия внутри алгоритма в виде последовательности операций. Блок-схемы алгоритмов авторизации и регистрации пользователя представлены на рисунке 2.6 и 2.7 соответственно.

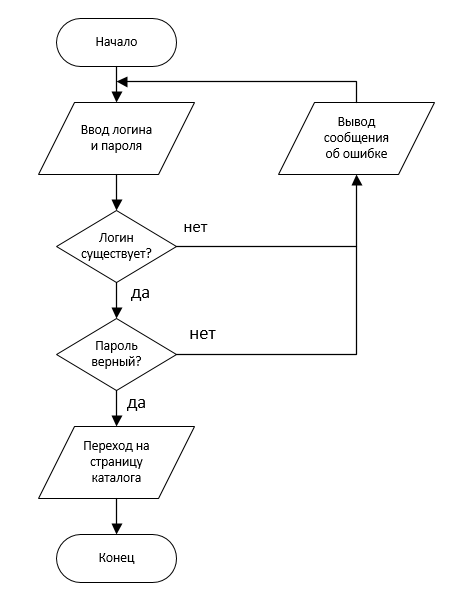


Рисунок 2.6 - Блок-схема алгоритма авторизации пользователя



Рисунок 2.7 - Блок-схема алгоритма регистрации пользователя

# Проектирование структуры базы данных. Разработка дизайна пользовательского интерфейса.

Для создания базы данных использовался фреймворк Entity Framework Core с подходом Code First, что позволяет однозначно связать отношения базы данных с моделями, используемыми в коде программы.

## Код Entity Framework Core

Код создания базы данных в MyContext представлен в Приложении А.

Для отношений заданы внешние ключи, а также режим удаления. Например, при удалении товара заказы удаляются вместе с ними каскадно (соответствующие предупреждения уведомляют пользователя при попытке удаления в приложении).

## Схема базы данных

Схема базы данных визуально отображает отношения базы данных, свойства этих отношений и связь между отношениями. Логическая схема базы данных представлена на рисунке 3.1.

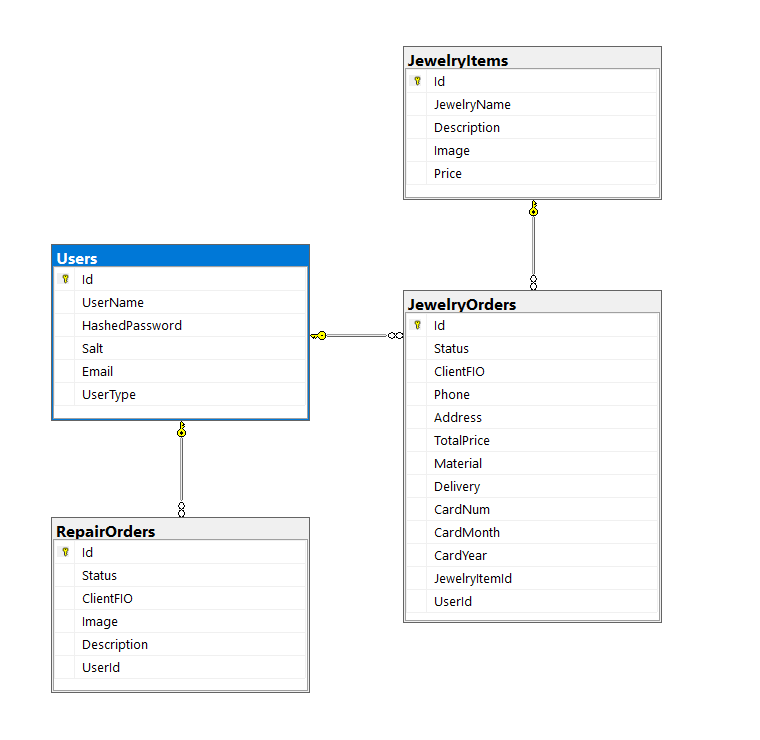


Рисунок 3.1 – Логическая схема базы данных проектируемого средства

Схема содержит следующие отношения:

JewelryItems – изделия.

Users – пользователи.

RepairOrders – заказы на ремонт, ссылается по внешнему ключу UserId на таблицу пользователей.

JewelryOrders – заказы изделий, ссылается по внешним ключам UserId и JewelryItemId на таблицы пользователей и изделий.

## Интерфейс приложения

Для проектирования приложения использована платформа WPF. В ходе курсового проектирования разработан интерфейс приложения для администратора и клиента. Некоторые экранные формы представлены на рисунках 3.2 - 3.9:

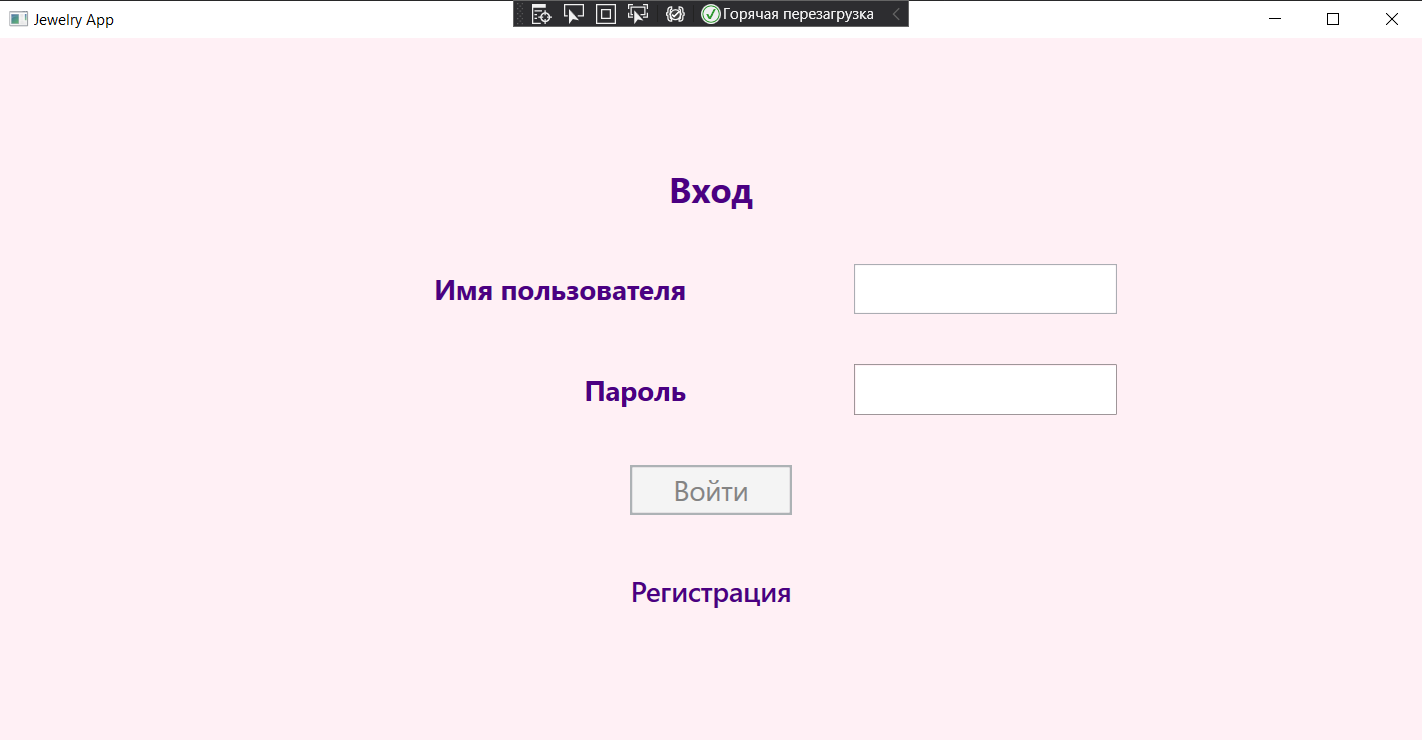


Рисунок 3.2 – Форма авторизации пользователя

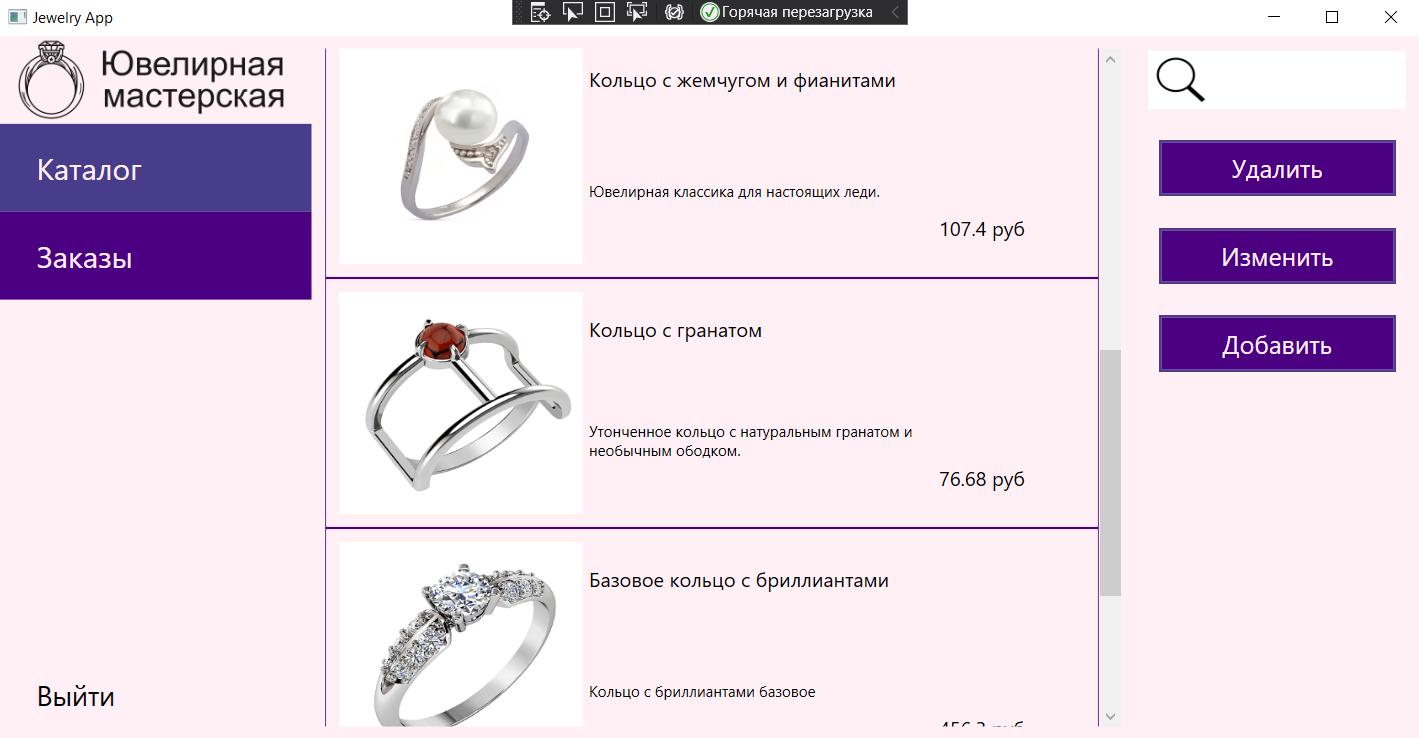


Рисунок 3.3 – Страница каталога (администратор)

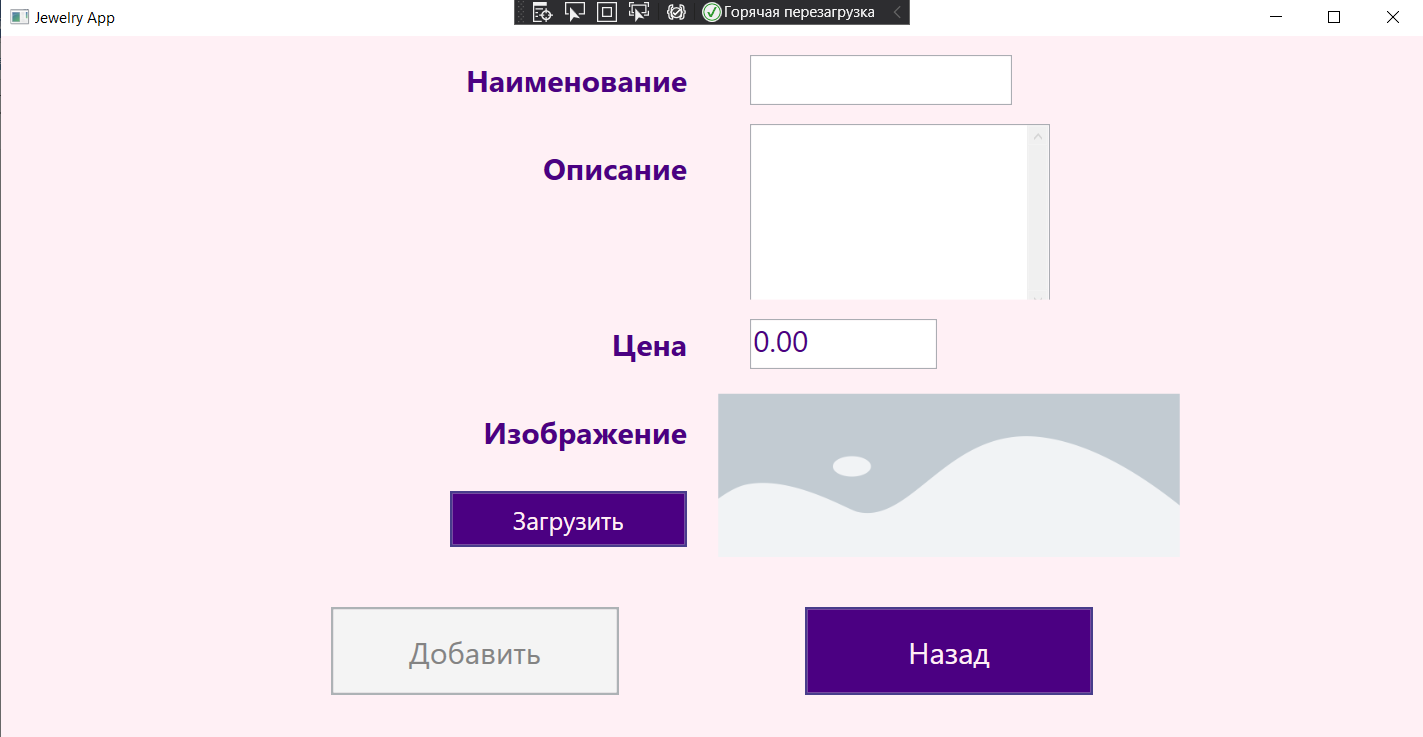


Рисунок 3.4 – Страница добавления нового товара

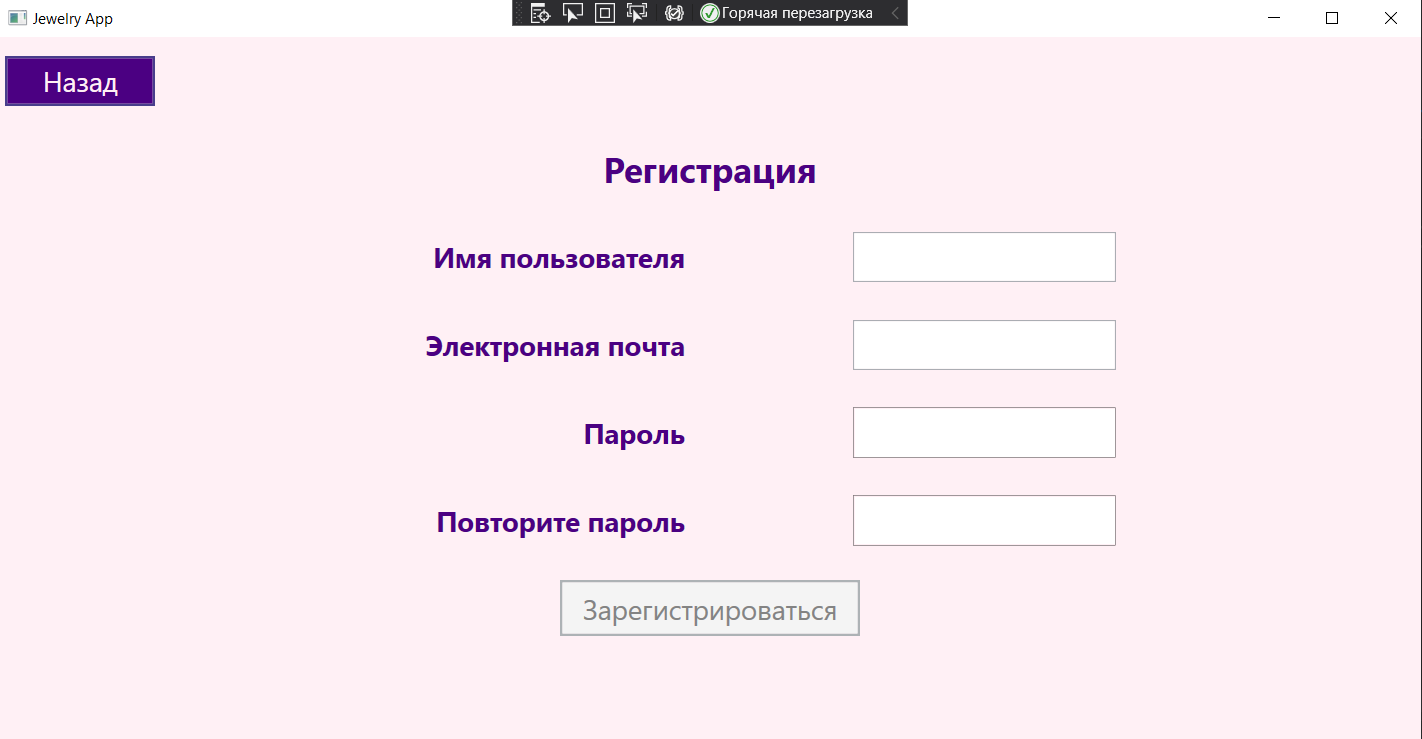


Рисунок 3.5 – Страница регистрации пользователя

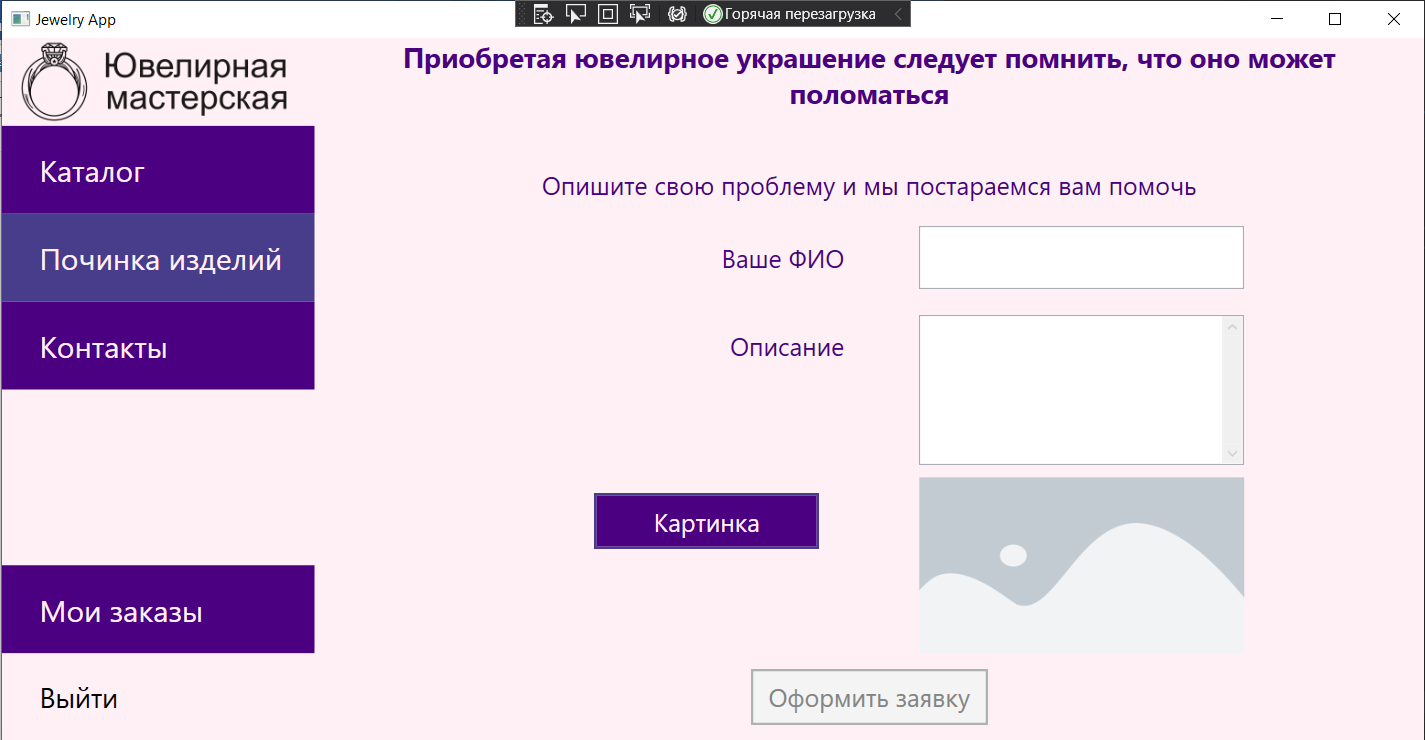


Рисунок 3.6 – Страница заказа на ремонт изделия

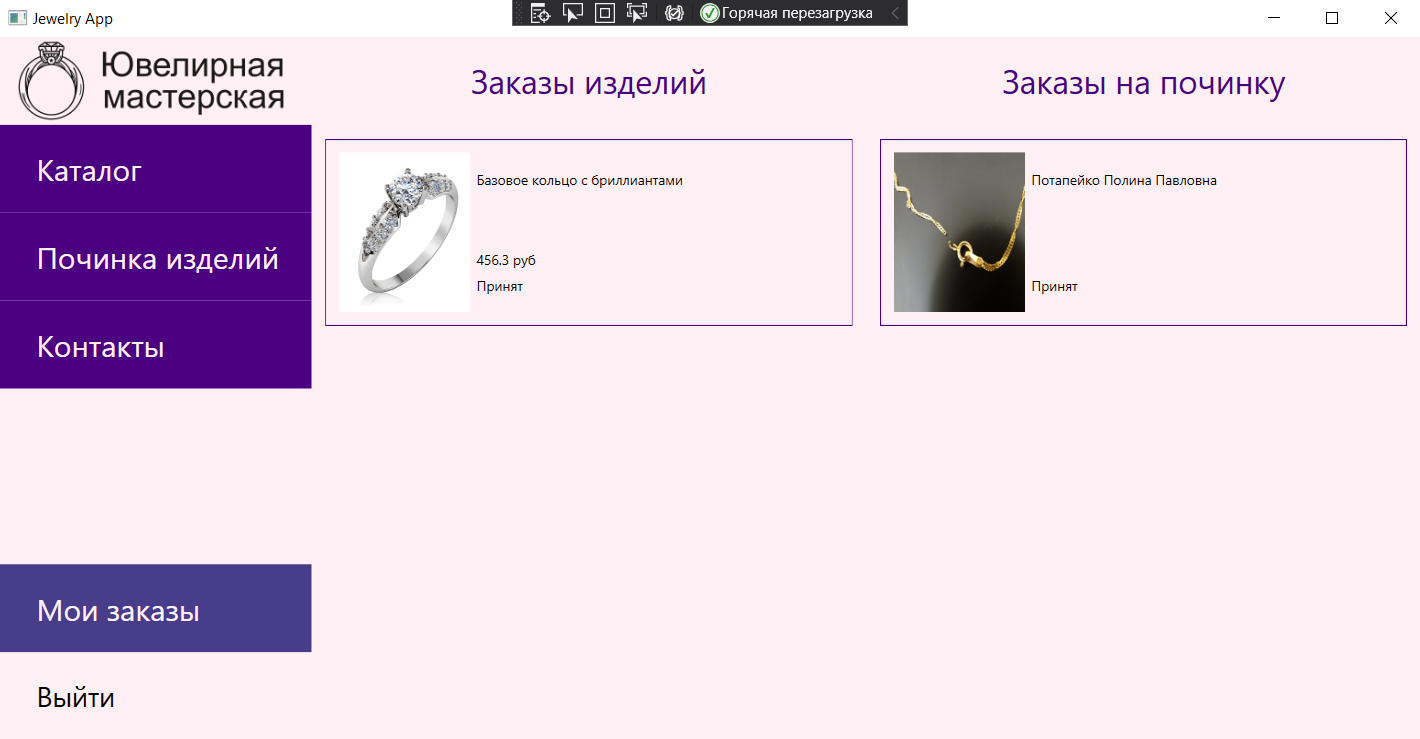


Рисунок 3.7 – Страница заказов клиента

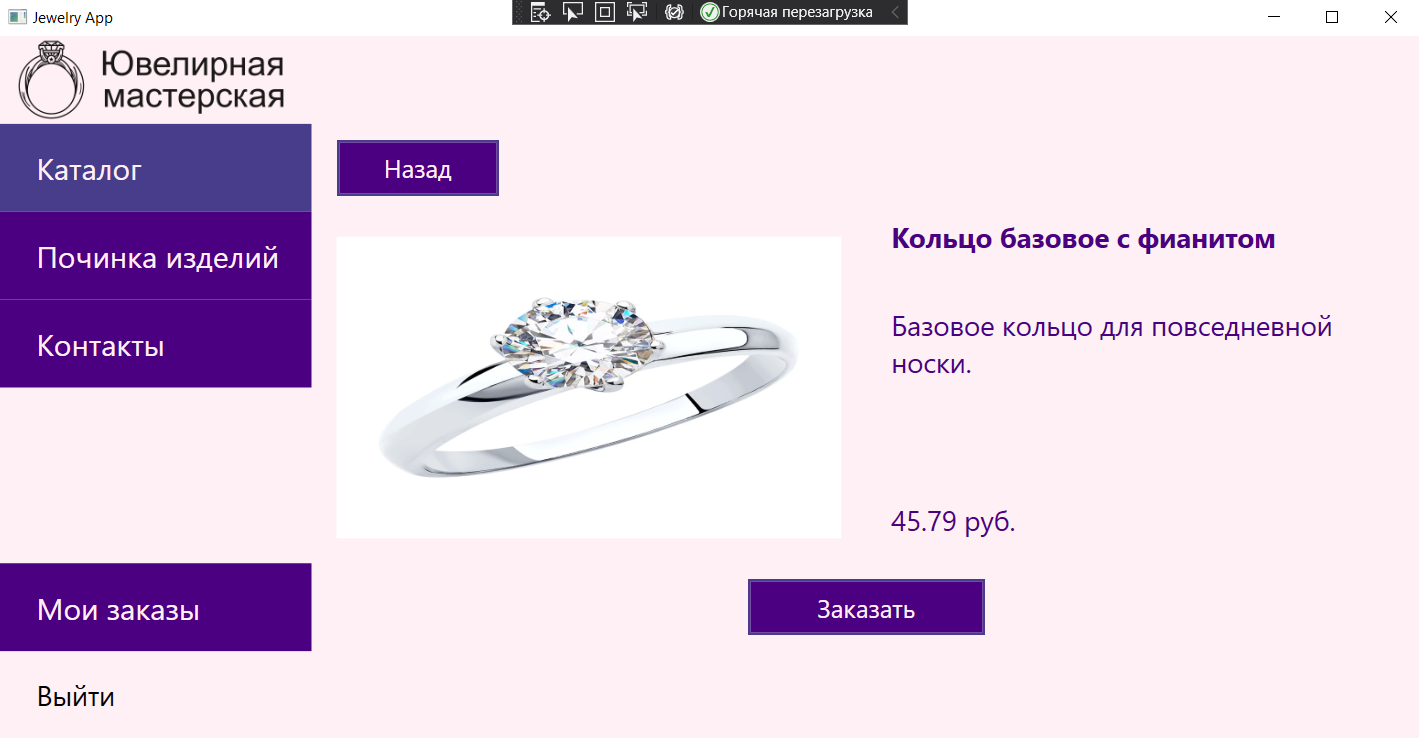


Рисунок 3.8 – Страница информации о конкретном изделии

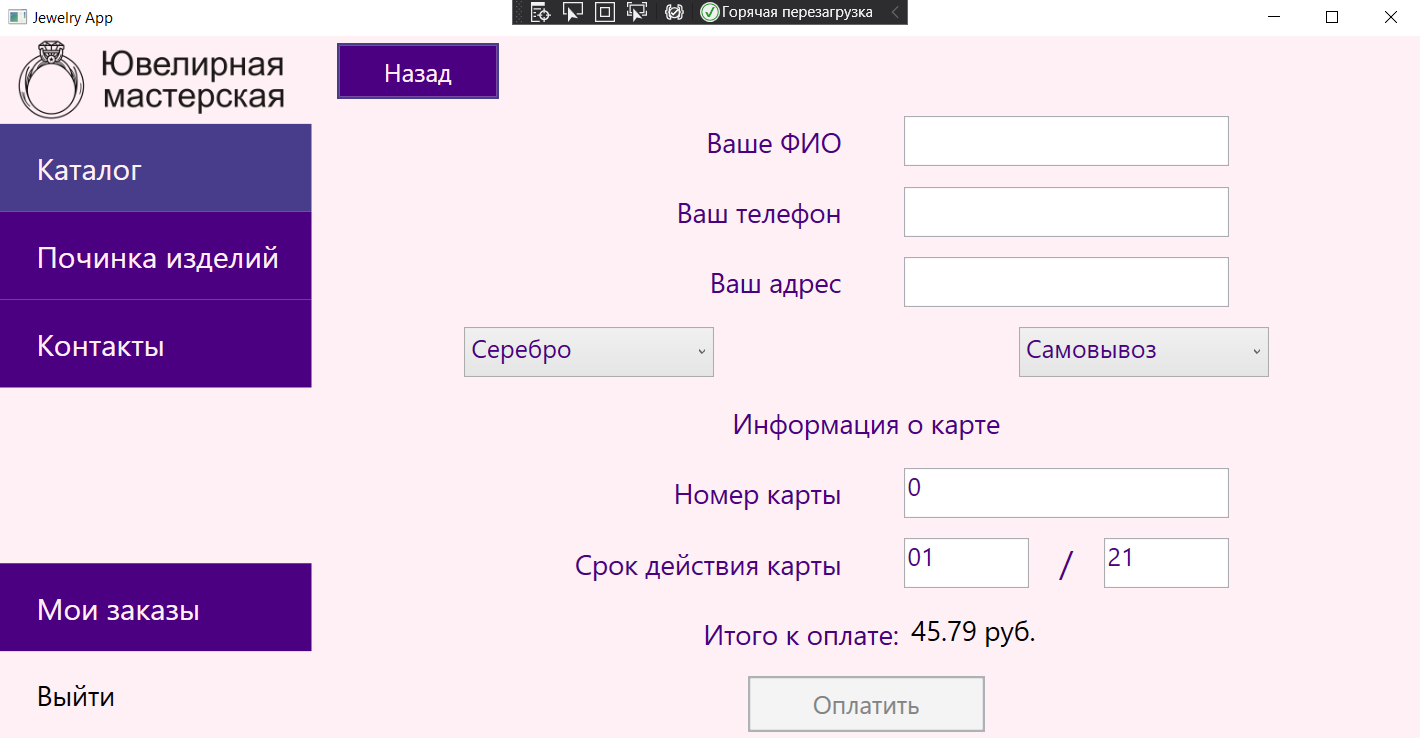


Рисунок 3.9 – Форма создания заказа

# Создание программного средства

Приложение выполнено на платформе WPF и представляет собой интерфейс для управления данными. За основу взят паттерн проектирования MVVM, значит, код разделен на представления (из .xaml файлов и .cs (code-behind)), модели и модели представления, к данным которых привязываются представления, и в которых производится работа над моделями. Такое разделение упрощает дальнейшую разработку приложения для расширения его функционала и поддержки продукта в целом.

## Уровень бизнес-логики

Для работы с данными используются паттерны UnitOfWork и Repository.

Для реализации паттерна Repository был создан интерфейс IRepository, принимающий параметром класс модели. В интерфейсе определены основные методы для манипуляций с данными базы данных: добавление, удаление, изменение, получение по уникальному номеру-id и получение всех записей соответствующей таблицы. Далее были созданы отдельные классы, наследуемые от интерфейса IRepository для работы с каждой сущностью базы данных.

Для реализации паттерна UnitOfWork был создан класс UnitOfwork, который будет в дальнейшем использоваться для обращения к каждому соответствующему репозиторию. Для создания объекта этого класса используется класс UnitOfWorkFactory, объект этого класса передается между всеми моделями представления приложения.

Листинг кода интерфейса IRepository, конкретного репозитория JewelryItemRepository и UnitOfwork приведены в приложении Б.

## Уровень представлений

Уровень представления представлен XAML-файлами. Каждый файл описывает сменяемую страницу, являющуюся пользовательским элементом управления. Навигация по страницам происходит с помощью нажатия на определенный пункт меню, которое присутствует на всех важных представлениях. Для ввода данных использовались TextBox и ComboBox, для вывода информации из базы данных использовались ListView и TextBlock. Для отображения данных использовался механизм привязки Binding. Навигация реализована с помощью команд, в которые передаются нужные модели представления, связанные с соответствующими представлениями.

Таблица 4.4 – описание представлений

|  |  |  |
| --- | --- | --- |
| Имя представления | Описание | Уровень доступности |
| ConcreteJewelryOrderAdminView | Информация о конкретном заказе изделия клиента | Администратор |
| ConcreteJewelryOrderClient | Информация о конкретном заказе изделия клиента | Клиент |
| ConcreteRepairOrderAdminView | Информация о конкретном заказе на починку | Администратор |
| ConcreteRepairOrderClientView | Информация о конкретном заказе на починку | Клиент |
| ContactsView | Контактная информация о мастерской | Клиент |
| ItemInfoClientView | Информация о конкретном товаре | Клиент |
| JewelryItemView | Информация о конкретном товаре с возможностью заполнения полей | Администратор |
| LogInView | Представление авторизации | Клиент, администратор |
| OrderClientView | Заполнение информации для создания заказа изделия | Клиент |
| OrdersAdminView | Все заказы клиентов | Администратор |
| OrdersClientView | Все заказы конкретного клиента | Клиент |
| RegisteryView | Регистрация клиентов | Клиент, администратор |
| RepairClientView | Заполнение информации для создания заказа на ремонт | Клиент |
| StoreAdminView | Просмотр всех товаров с возможностью их удаления, изменения, создания | Администратор |
| StoreClientView | Просмотр всех товаров | Клиент |

## Уровень модели представлений

Уровень модели представлений классами с добавлением части Model к названию соответствующего представления. Все классы наследуются от базового класса ViewModel, который реализует интерфейс INotifyPropertyChanged, благодаря которому осуществляется привязка данных из уровня представлений и соответствующей модели представлений. В каждой модели представления также присутствует реализация команд перехода в нужное представление.

# Тестирования и отладка программного средства

Тестирование нужно для обнаружения и устранения неполадок, способных привести к некорректной работе приложения, и как следствие, потери или искажении данных.

При тестировании были применены сценарии, которые могли вызвать исключительные ситуации, а также возможность некорректного ввода данных.

При заполнении форм для дальнейшего создания заказа, авторизации, регистрации и т.д. нужно заполнить обязательные поля. Пока эти поля не будут заполнены кнопки перехода в другое состояние приложения будут неактивны, так как команды, к которым привязаны соответствующие кнопки, проверяют эти поля на заполнение. Это позволяет избежать значительного числа ошибок по невнимательности пользователей. Пример неактивной кнопки приведен на рисунке 5.1. Эта же кнопка становится активной, когда все нужные поля заполнены (рисунок 5.2).

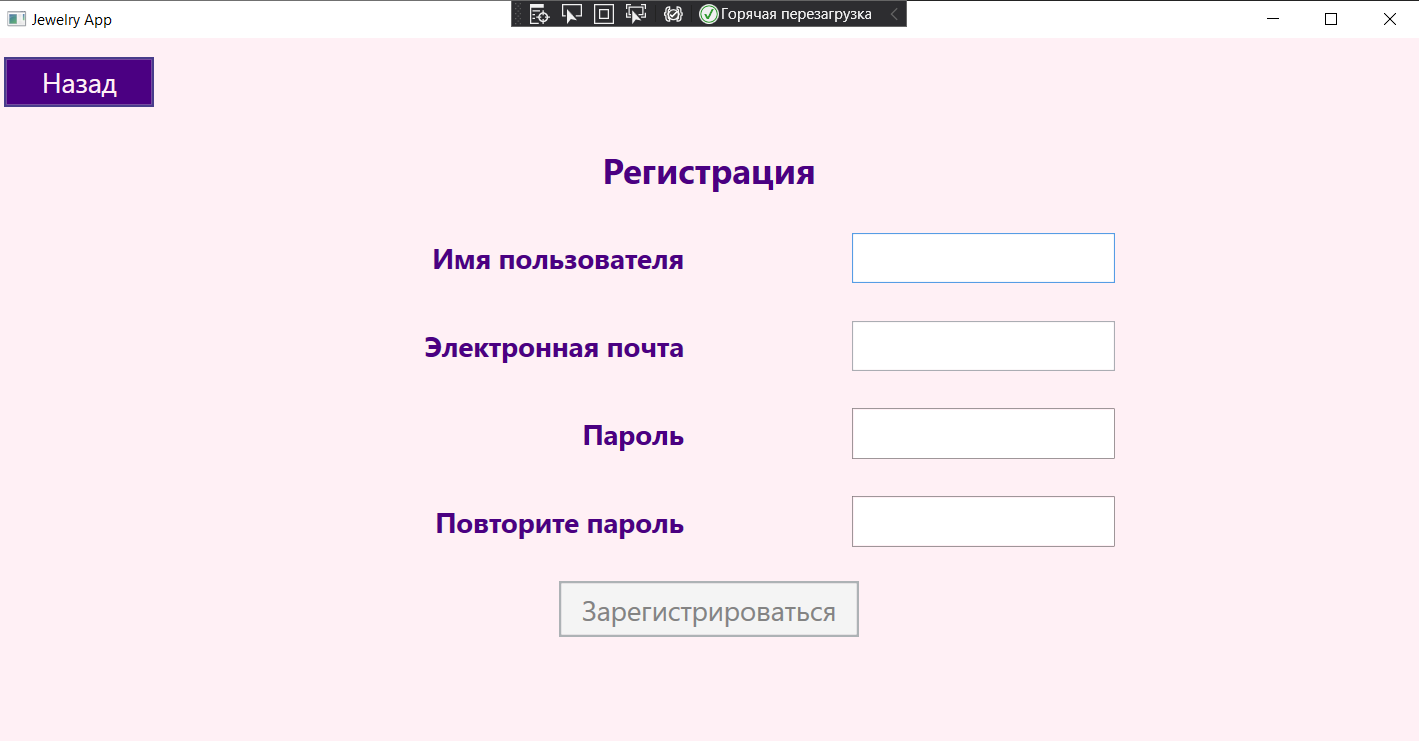


Рисунок 5.1 – Неактивная кнопка в форме регистрации

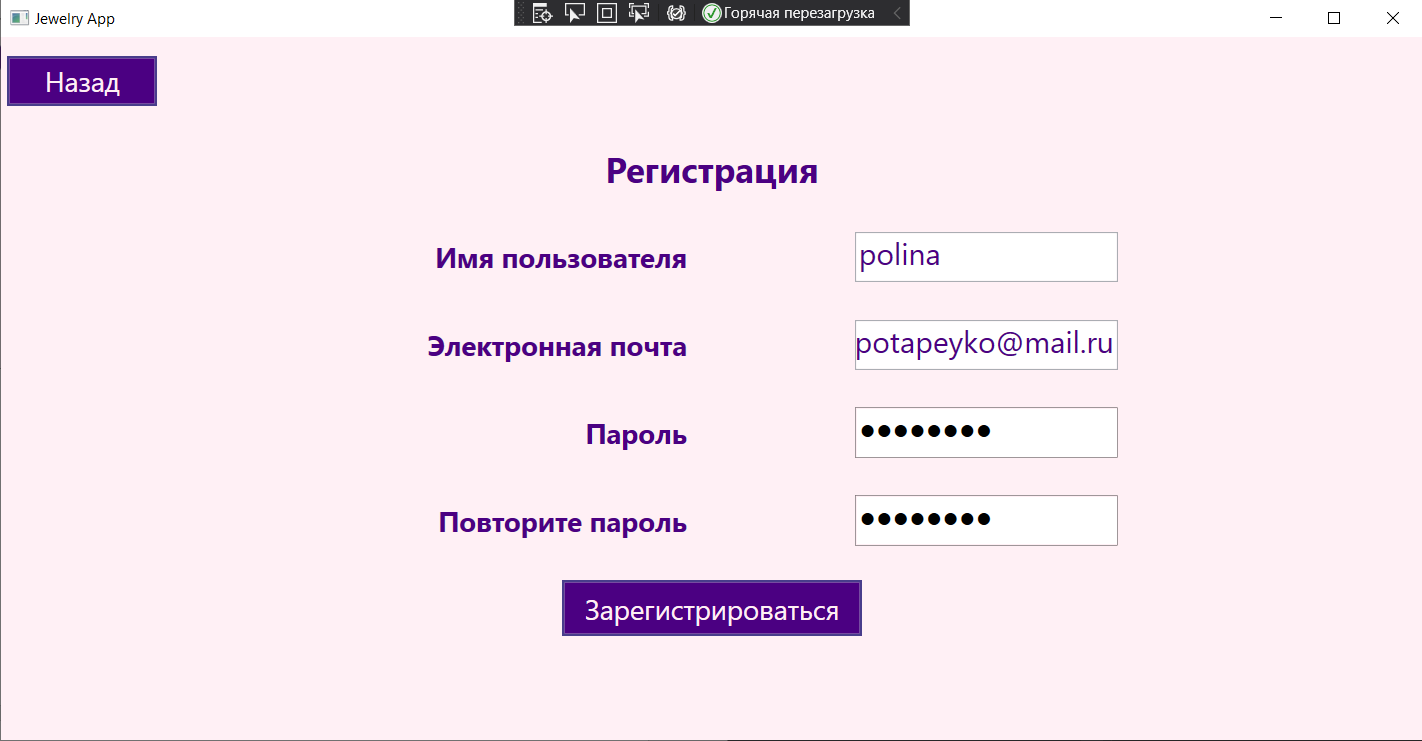


Рисунок 5.2 – Кнопка активна при заполнении всех обязательных полей

При попытке заполнить поля некорректными данными появляются сообщения об ошибке либо предупреждения о некорректном вводе данных. Пример некорректного ввода и устранения ошибки приведен на рисунке 5.3.

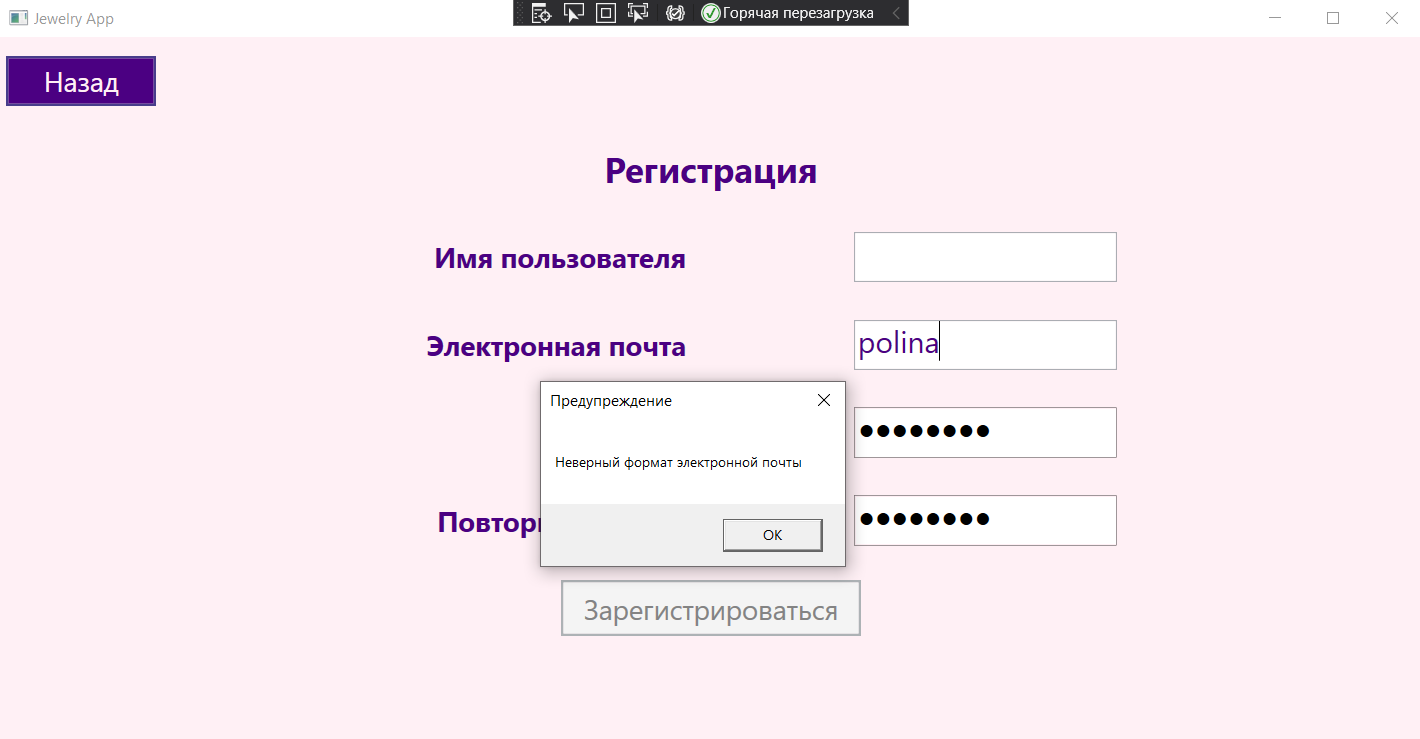


Рисунок 5.3 – Некорректный ввод электронной почты.

Далее отслеживаются существующие логины. Если логин существует, то пользователь увидит сообщение об ошибке. Пример приведен на рисунке 5.4.

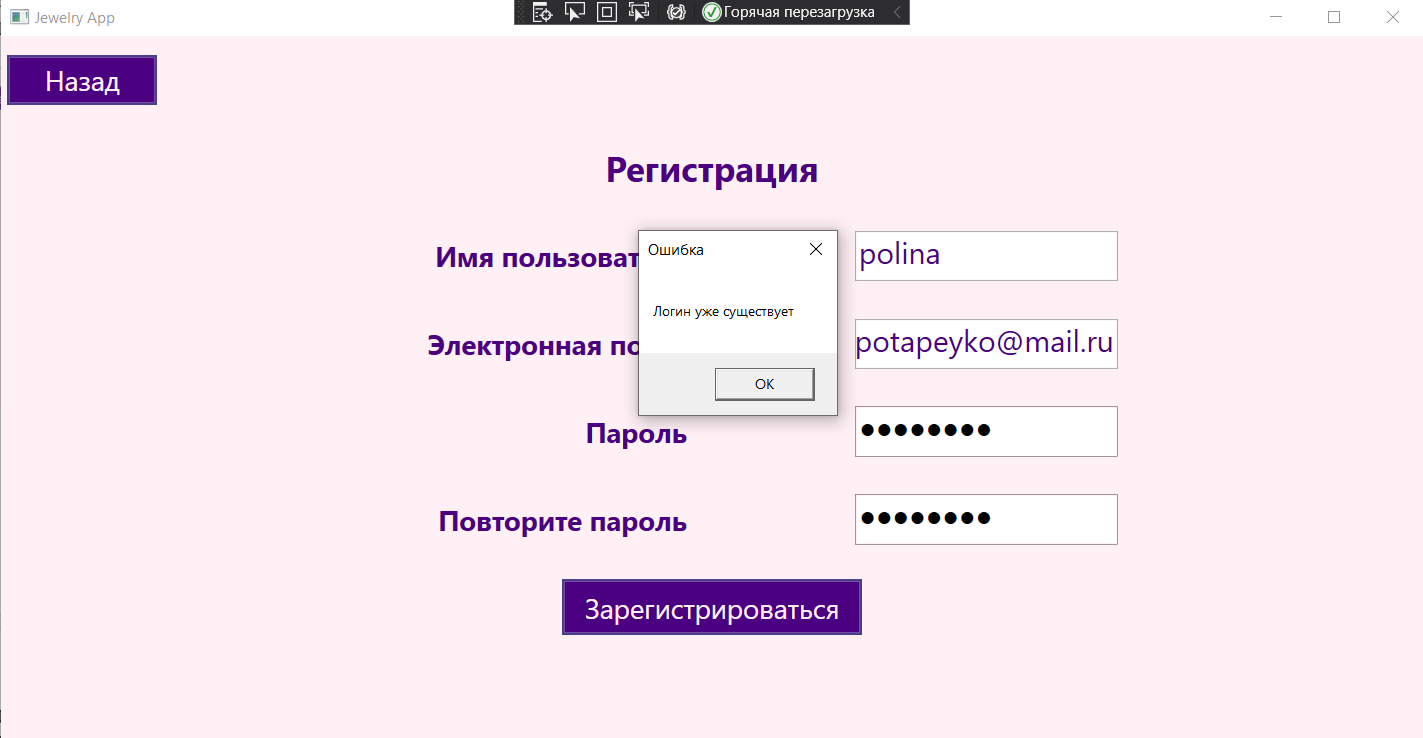


Рисунок 5.4 – Логин существует.

Также если пользователь в поле подтверждения пароля введет другую последовательность значений, то появится сообщение, что пароли не совпадают. Пример приведен на рисунке 5.5. Также пользователь не может ввести пароль короче 5 символов.

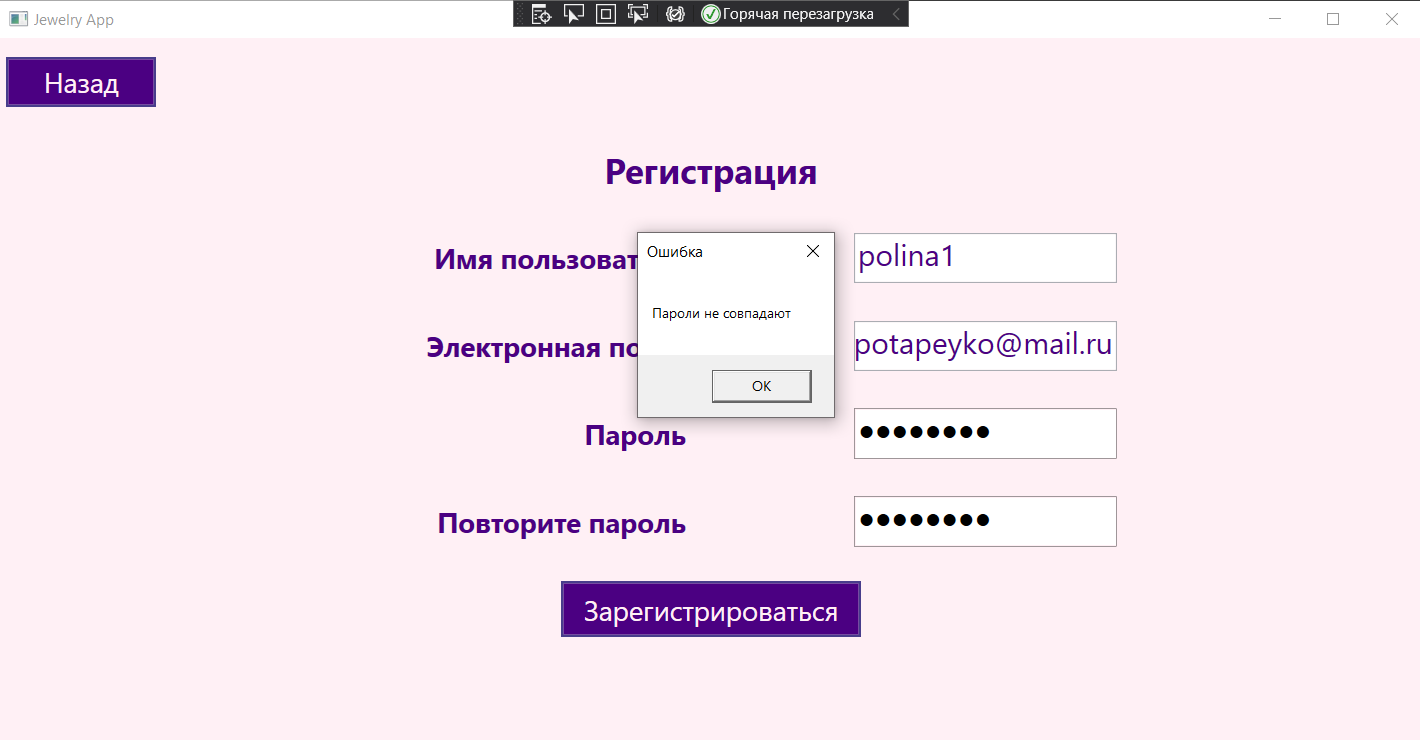


Рисунок 5.5 – Пароли не совпадают.

При вводе несуществующего логина или неправильного пароля, появится сообщение об ошибке. Пример приведен на рисунке 5.6.

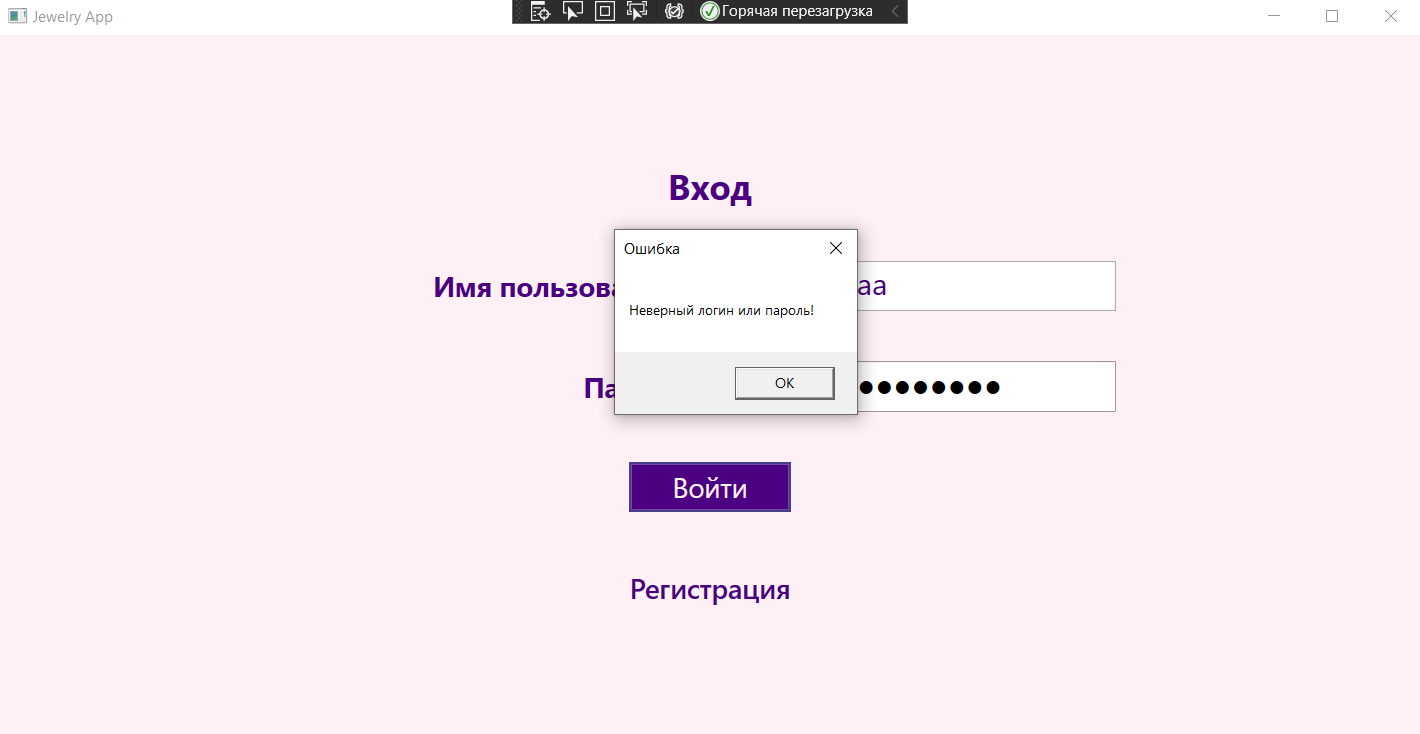


Рисунок 5.6 – Неверный логин или пароль.

При попытке изменения или удаления товара в представлении каталога администратора, если товар не выбран, появится сообщение об ошибке. Демонстрация приведена на рисунке 5.7.

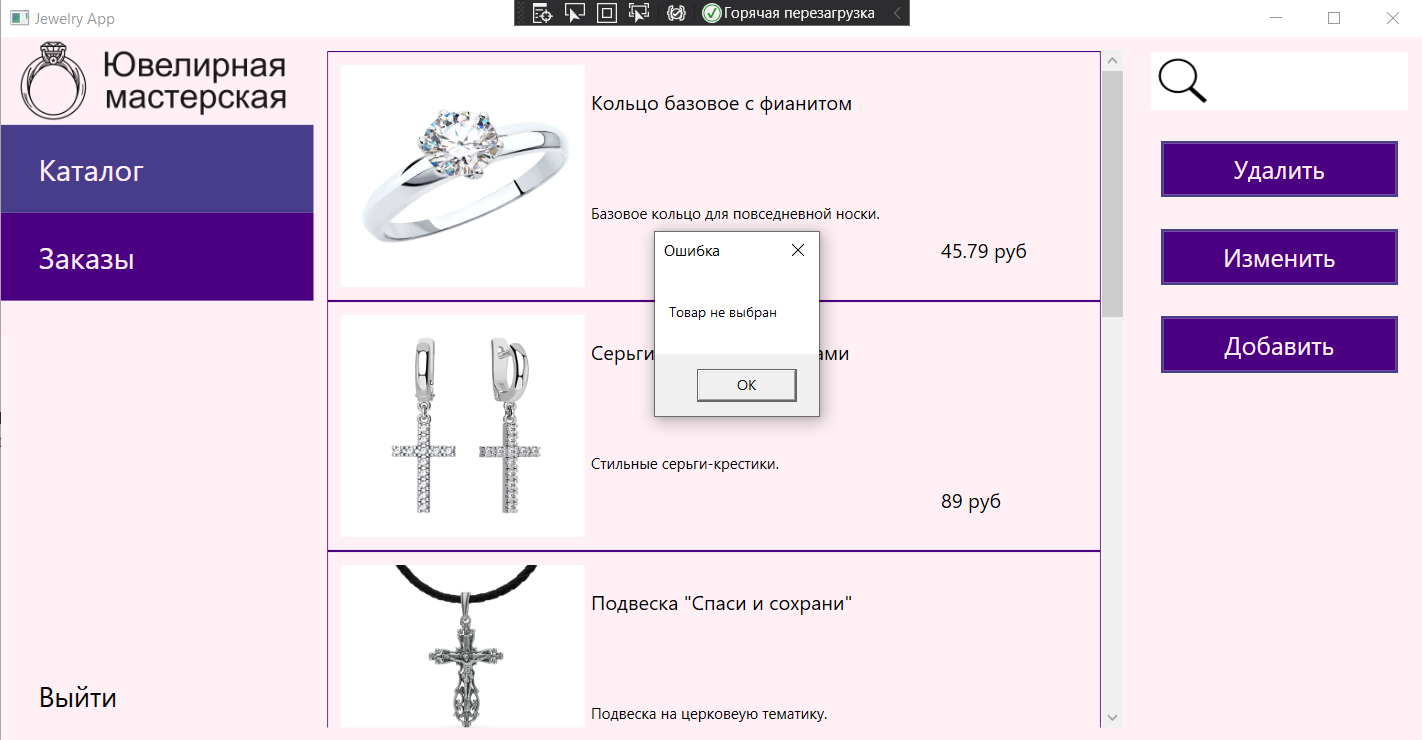


Рисунок 5.7 – Сообщение о невыбранном товаре.

# Руководство по установке и использованию

После запуска программы пользователю предлагается авторизоваться, однако, если пользователь впервые знакомиться с приложением, ему нужно пройти регистрацию. В случае успешной регистрации пользователь вновь попадает на страницу авторизации, где он уже может ввести свой логин и пароль. После успешной авторизации пользователь-клиент попадает на страницу каталога, где представлены все товары, доступные для заказа (рисунок 6.1).

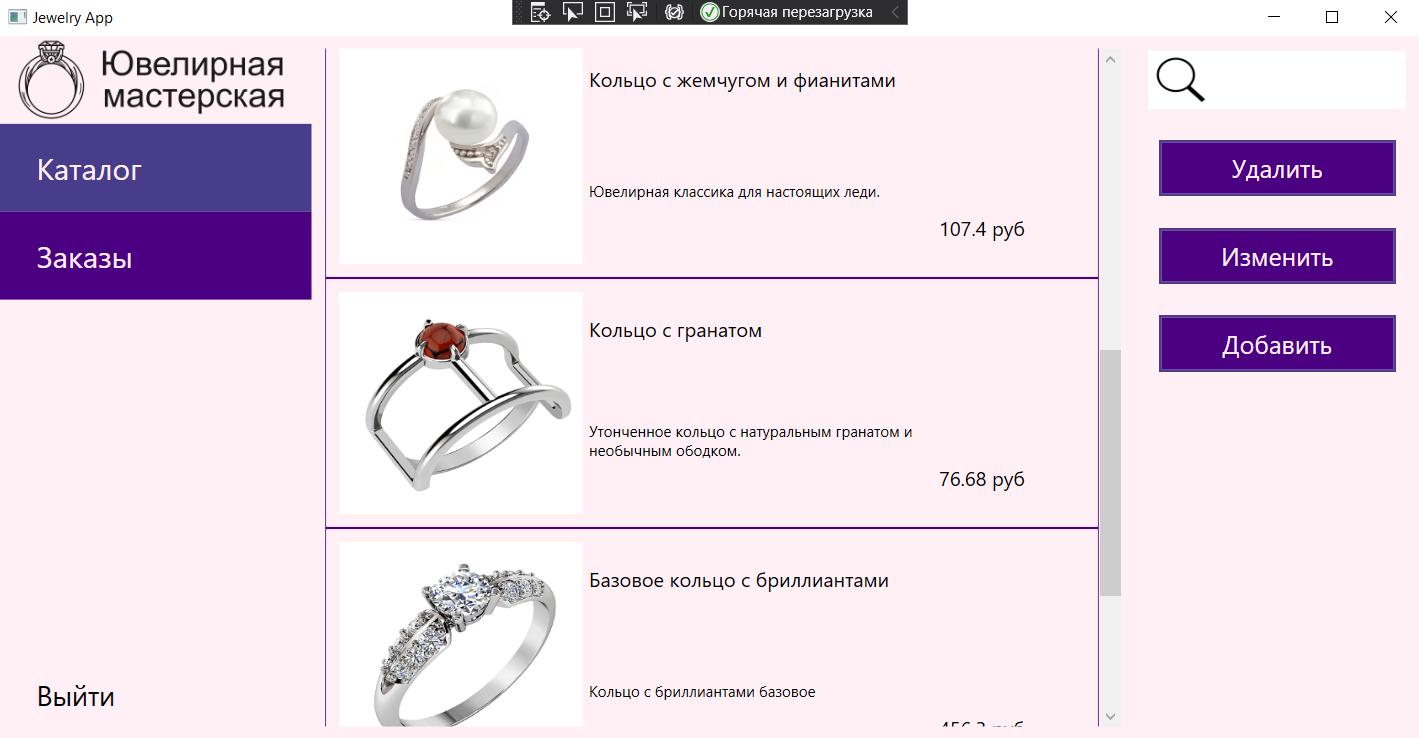


Рисунок 6.1 – Страница каталога.

Затем необходимо выбрать товар, который интересует пользователя и кликнуть по товару из списка двойным щелчком мыши. Откроется страница с информацией об этом товаре (рисунок 6.2).

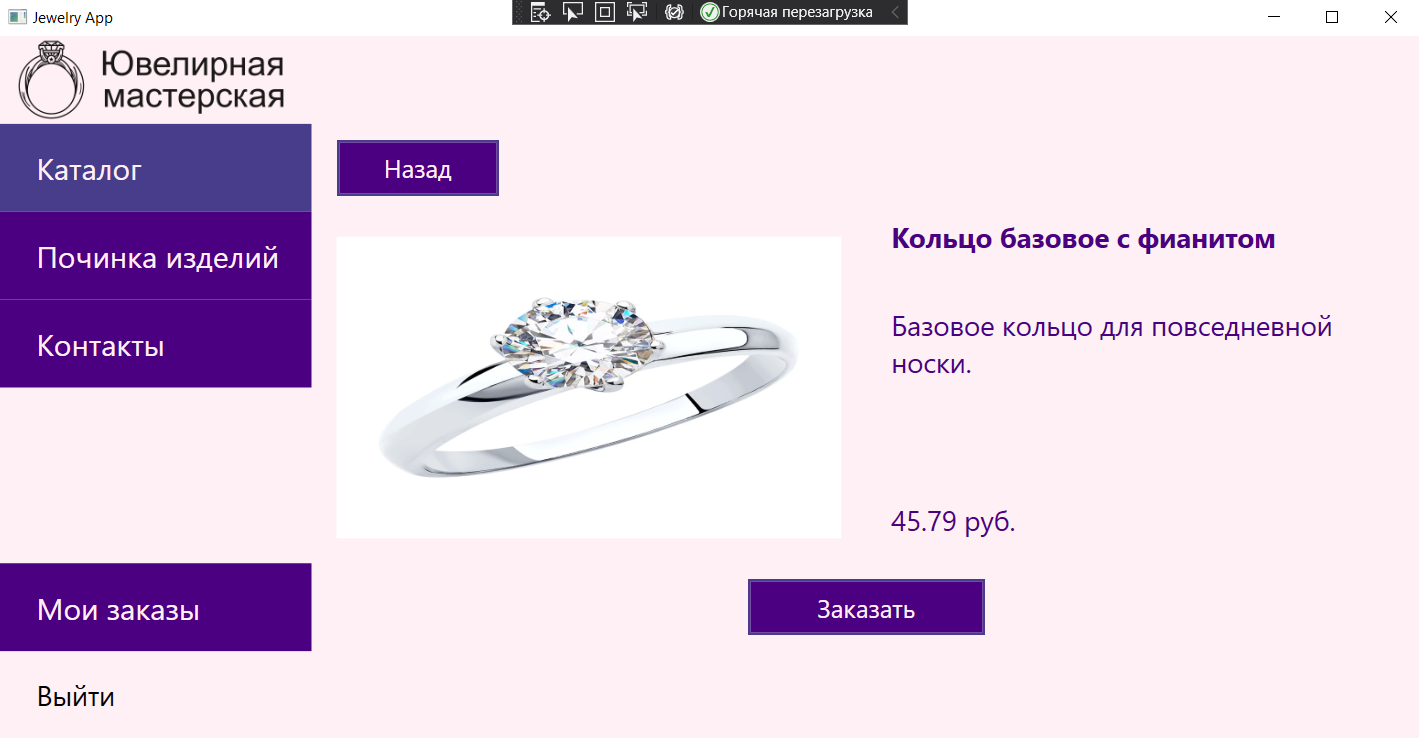


Рисунок 6.2 – Страница с информацией о конкретном изделии.

Следующим шагом необходимо нажать на кнопку *«Заказать».* Откроется страница заказа (рисунок 6.3).

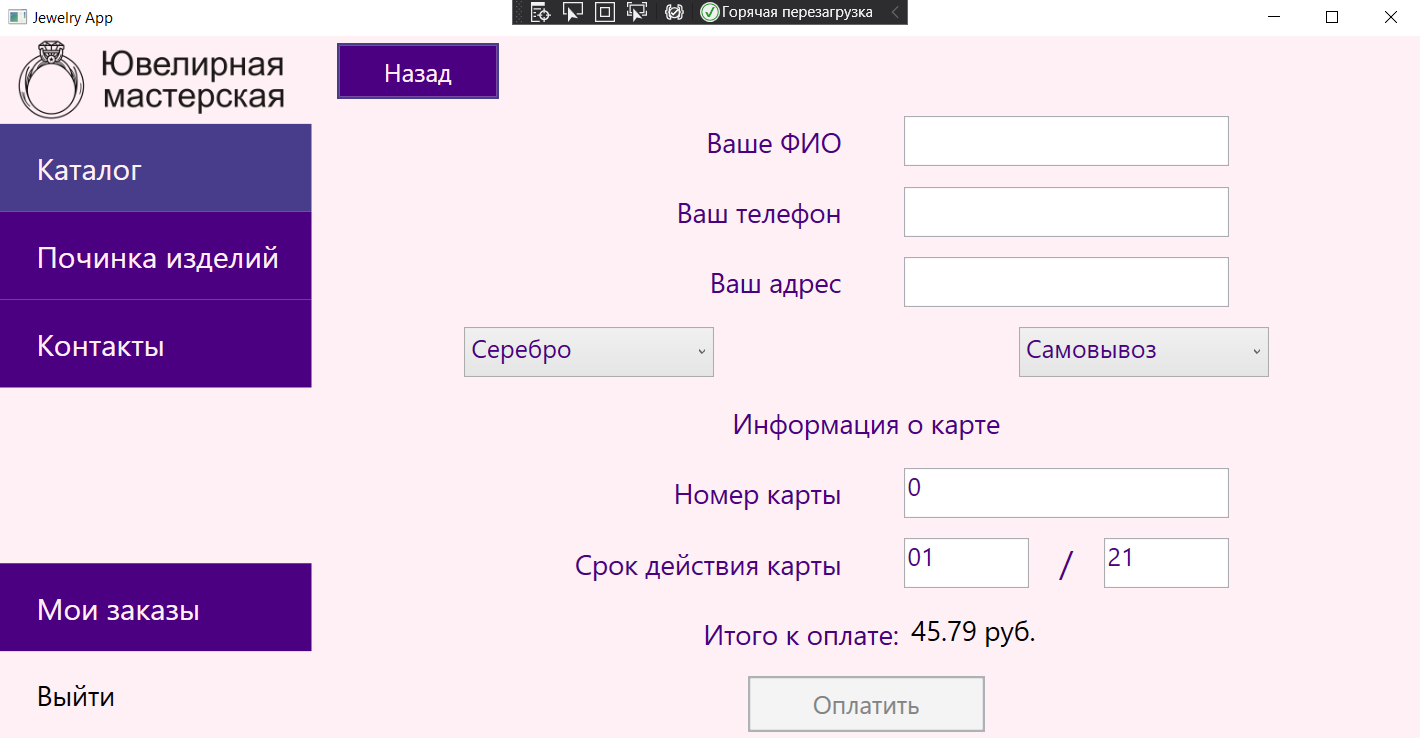


Рисунок 6.3 – Страница заказа.

Заполнив все необходимые поля, следует нажать на кнопку *«Оплатить»* –произойдет добавление заказа в список заказов клиента и переход на страницу заказов (рисунок 6.4).

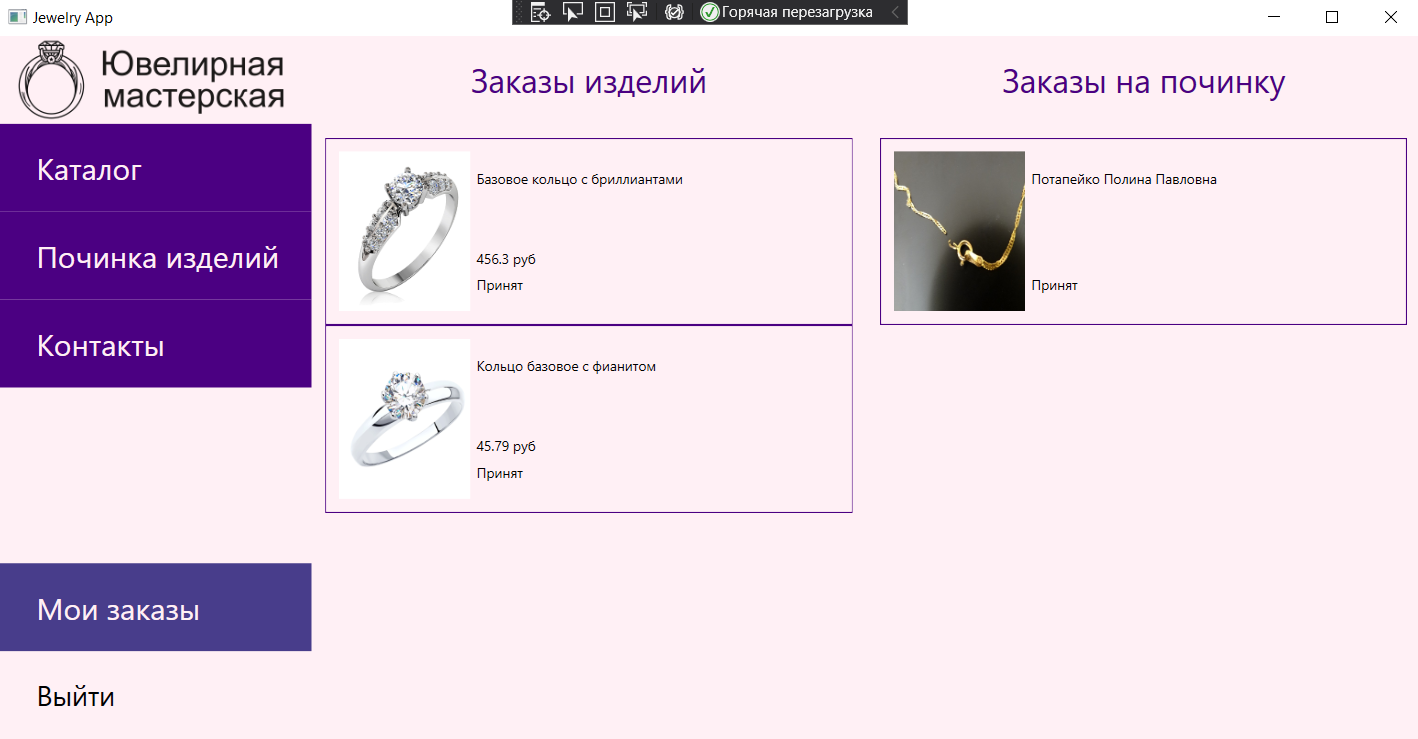


Рисунок 6.4 – Страница заказов клиента.

Затем двойным щелчком мыши кликнув по заказу, можно посмотреть подробности этого заказа (рисунок 6.5).

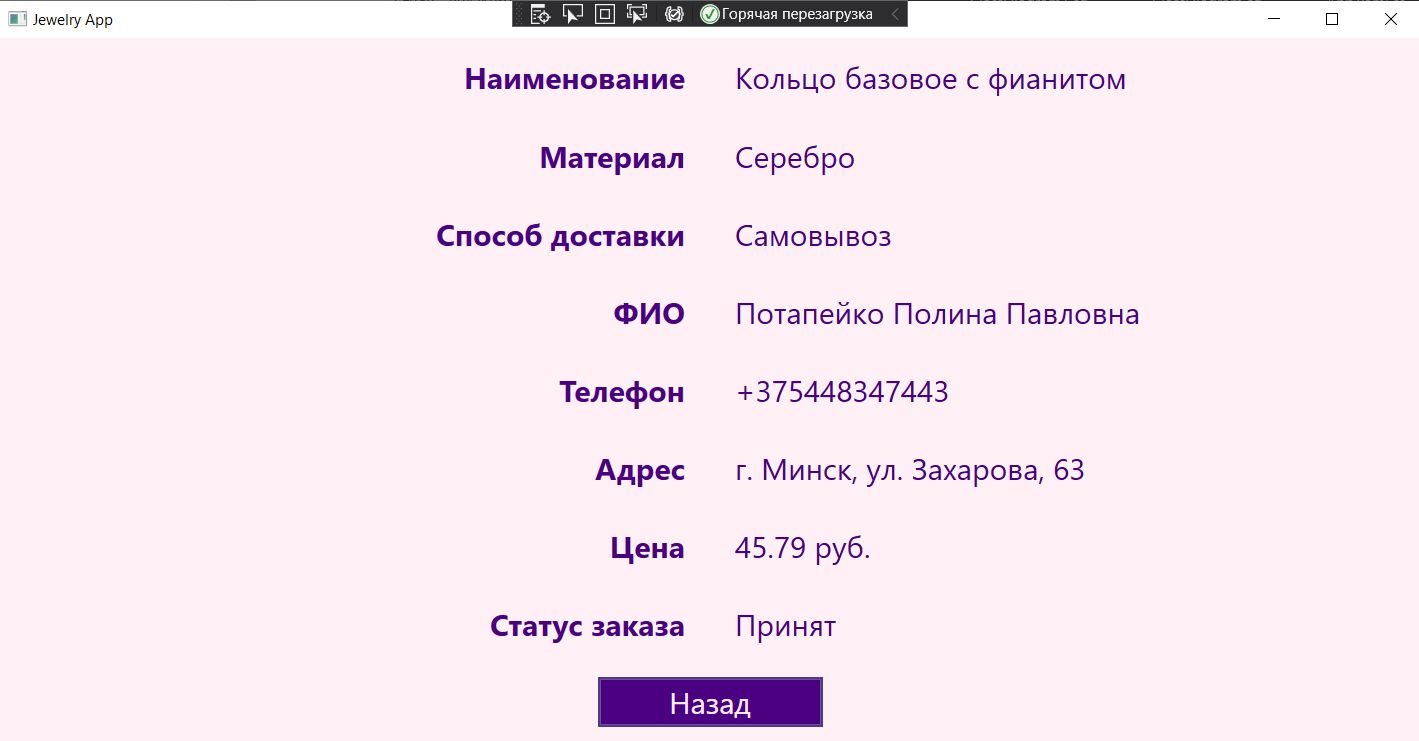


Рисунок 6.5 – Подробная информация о заказе.

Заключение

В результате проектирования курсового проекта было разработано приложение, реализующее основные функции ювелирной мастерской. В ходе анализа возможностей разработанного ПС был сделан вывод о том, что полноценная реализация подобного ПС требует более глубокой детализации предметной области.

Разработанное ПС реализовано с использованием платформы .NET Core, языка программирования C# и технологии WPF, применение базы данных в качестве хранилища информации позволяет оптимально и эффективно хранить информацию, структурировать ее.

В программном средстве были выполнены все пункты из указанного списка предполагаемого основного функционала приложения. Программное средство удовлетворяет поставленным задачам. В соответствии с полученным результатом работы программы можно сделать вывод, что разработанная программа работает верно, а требования технического задания выполнены в полном объеме.

Для более корректной работы вся система построена на проверках валидации и связи объектов между собой, поэтому любое действие со стороны администратора должно быть обдуманным, поскольку данная система является автоматизированной.

Приложение A



Листинг 1. Листинг создания базы данных в MyContext

Приложение Б



Листинг 2. Листинг кода интерфейса IRepository.



Листинг 3. Листинг кода конкретного репозитория JewelryItemRepository.



Листинг 4. Листинг кода класса UnitOfwork.

Список использованных источников

[1]. METANIT.COM Сайт о программировании [Электронный ресурс] / Режим доступа: <https://metanit.com>

[2]. ProfessorWeb .NET & Web Programming [Электронный ресурс] / Режим доступа: [https://professorweb.ru](https://professorweb.ru/)

[3]. Docs Microsoft [Электронный ресурс] / Режим доступа: [https://docs.microsoft.com](https://docs.microsoft.com/)

[4]. Пацей, Н.В. Курс лекций по языку программирования С# / Н.В. Пацей. – Минск: БГТУ, 2018. – 175 с.