МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ

БЕЛАРУСЬ

Учреждение образования «БЕЛОРУССКИЙ

ГОСУДАРСТВЕННЫЙ

ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»

Факультет Информационных технологий

Кафедра Программной инженерии

Специальность 6-05-0612-01 «Программная инженерия»

Направление специальности 6-05-0612-01 «Программная инженерия» (программирование интернет-приложений)

**ПОЯСНИТЕЛЬНАЯ ЗАПИСКА**

**КУРСОВОГО ПРОЕКТА:**

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

Тема Программное средство «Продажа и ремонт копировальной техники»

Исполнитель

Студент(ка) 2 курса группы 7 Качинскас Вацловас Вацловович

(Ф.И.О.)

Руководитель работы ст. преподаватель Белодед Н.И.

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

Курсовой проект защищен с оценкой

Председатель Смелов В.В.

(подпись)

Минск 2025

Содержание

[ВВЕДЕНИЕ 5](#_Toc198601417)

[1 Аналитический обзор литературы и формирование требований 6](#_Toc198601418)

[1.1 Анализ прототипов 6](#_Toc198601419)

[1.1.1 Интернет-ресурс «CopyPro.net» 6](#_Toc198601420)

[1.1.2 Интернет-ресурс «Copyservices.ru» 7](#_Toc198601421)

[1.1.3 Интернет-ресурс «Copierts.ru» 8](#_Toc198601422)

[1.2 Постановка задачи 9](#_Toc198601423)

[2 Анализ требований к программному средству и разработка функциональных требований 10](#_Toc198601424)

[2.1 Описание средств разработки 10](#_Toc198601425)

[2.1.1 Microsoft Visual Studio 2022 10](#_Toc198601426)

[2.1.2 Программная платформа .NET Framework 10](#_Toc198601427)

[2.1.3 Язык программирования C# 10](#_Toc198601428)

[2.1.4 Технология WPF 10](#_Toc198601429)

[2.1.5 Расширяемый язык разметки XAML 11](#_Toc198601430)

[2.1.6 Технология Entity Framework 11](#_Toc198601431)

[2.1.7 MS SQL Server 11](#_Toc198601432)

[2.1.8 Паттерн MVVM 12](#_Toc198601433)

[2.2 Спецификация функциональных требований к программному средству 12](#_Toc198601434)

[2.3 Спецификация функциональных требований 13](#_Toc198601435)

[3 Проектирование программного средства 14](#_Toc198601436)

[3.1 Общая структура 14](#_Toc198601437)

[3.2 Взаимоотношение между классами 16](#_Toc198601438)

[3.3 Модель базы данных 16](#_Toc198601439)

[3.4 Проектирование архитектуры приложения 20](#_Toc198601440)

[3.5 Проектирование последовательностей проекта 21](#_Toc198601441)

[4 Реализация программного средства 22](#_Toc198601442)

[4.1 Основные классы программного средства 22](#_Toc198601443)

[4.2 Авторизация 22](#_Toc198601444)

[4.3 Регистрация 22](#_Toc198601445)

[4.4 Добавление товара/услугу в корзину 22](#_Toc198601446)

[4.5 Создание заказа 23](#_Toc198601447)

[4.6 Добавление комментариев 23](#_Toc198601448)

[5 Тестирование, проверка работоспособности и анализ полученных результатов 24](#_Toc198601449)

[5.1 Тестирование авторизации 24](#_Toc198601450)

[5.2 Тестирование регистрации 24](#_Toc198601451)

[5.3 Тестирование добавление товара в корзину 25](#_Toc198601452)

[5.4 Тестирование редактирования профиля 26](#_Toc198601453)

[5.5 Тестирование добавления комментариев 26](#_Toc198601454)

[5.6 Тестирование оформления заказа 27](#_Toc198601455)

[5.6 Тестирование добавления товара/услуг у администратора 28](#_Toc198601456)

[5.7 Тестирование удаления элементов администратором или менеджером 28](#_Toc198601457)

[6 Руководство по установке и использованию 30](#_Toc198601458)

[ЗАКЛЮЧЕНИЕ 33](#_Toc198601459)

[Список использованных источников 34](#_Toc198601460)

[ПРИЛОЖЕНИЕ А 35](#_Toc198601461)

[ПРИЛОЖЕНИЕ Б 36](#_Toc198601462)

[ПРИЛОЖЕНИЕ В 37](#_Toc198601463)

[ПРИЛОЖЕНИЕ Г 38](#_Toc198601464)

# ВВЕДЕНИЕ

В современном мире эффективное офисное оборудование играет ключевую роль в обеспечении бесперебойной работы организаций, предприятий и учреждений. Особенно востребованными становятся услуги по продаже и ремонту копировальной техники, так как высокое качество документооборота и своевременное техническое обслуживание устройств напрямую влияет на продуктивность бизнеса. Всё больше клиентов обращают внимание на специализированные сервисы, где предлагается широкий выбор техники, квалифицированная помощь и оперативное обслуживание.

Целью данного курсового проекта является создание программного средства, которое обеспечит пользователей удобным доступом к информации о компании, занимающейся продажей и ремонтом копировальной техники. Пользователи смогут ознакомиться с общей информацией о компании, перечнем предоставляемых услуг, ассортиментом техники и особенностями каждого товара.

С помощью программного средства пользователи смогут легко изучить каталог копировальной техники, узнать характеристики, наличие и стоимость интересующего оборудования. Также будет доступна функция оформления заказа, для зарегистрированных пользователей, а также возможность отслеживания статуса заказа. Особое внимание будет уделено удобству навигации и простоте интерфейса для разных категорий пользователей (администраторы, менеджеры, клиенты).

Для успешной реализации курсового проекта необходимо:

* провести анализ соответствующей литературы;
* ознакомиться с существующими аналогами;
* определить функциональные требования;
* продумать структуру базы данных;
* разработать архитектуру программного обеспечения;
* реализовать программное средство;
* протестировать систему;
* подготовить руководство пользователя.

В процессе выполнения курсового проекта будут применены принципы объектно-ориентированного программирования (ООП), а также использована база данных MS SQL Server и технология Windows Presentation Foundation (WPF).

# 1 Аналитический обзор литературы и формирование требований

## Анализ прототипов

Курсовой проект представляет из себя разработку приложения для компании, занимающейся продажей и ремонтом копировальной техники. В этой сфере существует множество аналогичных решений, которые предоставляют пользователям возможность просматривать ассортимент товаров, оформлять заказы, а также отслеживать статус ремонта. В ходе выполнения курсового проекта были проанализированы цели и задачи, а также рассмотрены аналогичные примеры их решений. На основании анализа всех достоинств и недостатков данных альтернативных решений были сформулированы требования к данному программному средству.

* + 1. Интернет-ресурс «CopyPro.net»

CopyPro — это компания, специализирующаяся на продаже и обслуживании копировальной техники. Их официальный сайт предоставляет пользователям широкий спектр возможностей — от подбора техники до оформления заявки на ремонт или покупку расходных материалов. Интерфейс сайта представлен на рисунке 1.1.

Изображение выглядит как текст, человек, снимок экрана, одежда

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 1.1 – Интернет-ресурс «CopyPro.net»

На сайте представлен обширный каталог копировальной техники с подробным описанием каждой модели, фотографиями и видеоматериалами. Для удобства пользователей реализованы фильтры по техническим характеристикам, цене и производителю, что облегчает поиск нужного оборудования.

Одной из ключевых функций сайта является возможность онлайн-записи на ремонт или заказ сервиса. Пользователь может заполнить форму и отправить заявку, не прибегая к звонкам или визитам. Также реализована интеграция с картами, благодаря чему клиент может легко найти ближайший сервисный центр.

Сайт корректно отображается как на настольных компьютерах, так и на мобильных устройствах, что делает его кросс-платформенным и удобным для использования в любом формате.

Положительным аспектом является наличие личного кабинета, где можно отслеживать историю заказов и сохранять предпочтения, а также наличие различных способов оплаты.

Интерфейс сайта интуитивно понятен, с четко структурированными разделами — каталог, сервис, поддержка. Карточки товаров содержат все необходимые данные, что помогает клиенту быстро принять решение о покупке.

Недостатков у сайта практически не выявлено, однако потенциальным улучшением могло бы стать добавление пользовательских отзывов и рейтингов для продукции.

* + 1. Интернет-ресурс «Copyservices.ru»

Copy-Service — это сервисный центр по ремонту и обслуживанию офисной печатной техники. Сайт компании ориентирован преимущественно на оказание услуг и предоставляет клиенту удобный инструмент для заказа заправки картриджей, ремонта и профилактического обслуживания. Интерфейс сайта представлен на рисунке 1.2.

Изображение выглядит как человек, снимок экрана, спорт, текст

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 1.2 – Интернет-ресурс «Copyservices.ru»

На сайте размещён перечень предлагаемых услуг с подробным описанием и фотографиями, что помогает клиенту заранее оценить объем и стоимость работы. Реализована возможность оформления онлайн-заявки на ремонт, что позволяет клиенту оперативно обратиться за помощью.

Сайт также содержит информацию о сервисных центрах с возможностью быстро определить местоположение через интеграцию с картами.

Преимущество ресурса — его простота и функциональность: с главной страницы можно перейти к интересующей услуге, узнать цены, условия и оставить заявку.

Функция фильтрации на сайте базовая, но достаточная для навигации по основным видам услуг. Также представлены сертификаты и отзывы клиентов, что способствует повышению доверия со стороны новых пользователей.

Сайт адаптирован под мобильные устройства и имеет быстрый отклик, однако отсутствует полноценный личный кабинет с возможностью управления заявками и предпочтениями, что можно отнести к минусам.

* + 1. Интернет-ресурс «Copierts.ru»

Copierts.ru — сайт сервисной компании КТС, ориентированной преимущественно на корпоративный сегмент. Основное внимание на сайте уделяется ремонту и техническому обслуживанию копировальной техники. Интерфейс сайта показан на рисунке 1.3.

Изображение выглядит как текст, снимок экрана, Шрифт, логотип

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 1.3 – Интернет-ресурс «Copierts.ru»

Ресурс предоставляет пользователю подробную информацию о видах ремонта, техническом обслуживании и диагностике. Описание включает спецификации и детали, что особенно важно для клиентов из B2B-сегмента.

На сайте доступна форма для онлайн-заявки, благодаря чему клиент может обратиться за ремонтом дистанционно, без необходимости визита в сервис.

Реализована интеграция с картами для отображения адресов сервисных точек, а также указаны контактные данные и каналы связи.

Сайт имеет четко ориентированный на профессионалов интерфейс: акцент сделан на структуре услуг, документации и технических деталях. Есть фильтрация по типам выполняемых работ, что упрощает поиск нужного сервиса.

Из плюсов стоит отметить наличие возможности ведения истории заказов и отчетности для корпоративных клиентов, что позволяет бизнесу отслеживать свои расходы и обращения.

Однако сайт больше ориентирован на функциональность, чем на визуальное оформление: отсутствие мультимедийных материалов (фото, видео), а также ограниченные пользовательские функции могут быть минусом для некоторых клиентов.

1.2 Постановка задачи

Обзор аналогичных решений позволяет проанализировать преимущества и недостатки существующих интернет-ресурсов, что дает возможность сформировать перечень требований к программному обеспечению, разрабатываемому в рамках данного курсового проекта.

Программное средство для компании, занимающейся продажей и ремонтом копировальной техники (Printinvest), должно обеспечивать удобство взаимодействия как для клиентов, так и для сотрудников компании. Ниже приведены ключевые функции, которые должны быть реализованы для разных ролей пользователей: пользователь, администратор и менеджер.

Функции пользователя:

* регистрация и авторизация в системе;
* просмотр каталога копировальной техники с фильтрацией;
* оформление заявки на покупку или ремонт оборудования;
* возможность оставления отзывов о товарах и качестве обслуживания.

Функции администратора:

* управление товарами и услугами в каталоге;
* создание и редактирование специальных предложений и акций;
* управление учетными записями пользователей;

Функции менеджера:

* управление заявками на ремонт и заказы (просмотр, изменение статуса, удаление);
* просмотр аналитики;
* оперативная обработка заявок на ремонт и покупку;
* просмотр и управление отзывами;

# 2 Анализ требований к программному средству и разработка функциональных требований

# 2.1 Описание средств разработки

При разработке приложения были использованы:

* интегрированная среда разработки Microsoft Visual Studio 2022;
* программная платформа .NET Framework 4.7.2;
* язык программирования C#;
* расширяемый язык разметки XAML;
* технология WPF;
* технология Entity Framework;
* MS SQL Server.

### 2.1.1 Microsoft Visual Studio 2022

Microsoft Visual Studio 2022 - это интегрированная среда разработки (IDE) от Microsoft, которая обеспечивает разработчиков инструментами и возможностями для создания приложений с использованием Windows Presentation Foundation (WPF).

### 2.1.2 Программная платформа .NET Framework

Платформа .NET Framework — это созданная в 2002 году технология Microsoft, которая поддерживает создание и выполнение веб-служб и приложений Windows, обеспечивающая согласованную объектно-ориентированной среду программирования для локального сохранения и выполнения объектного кода, локального выполнения кода, распределенного в Интернете, либо для его удаленного выполнения. Основой платформы является общеязыковая среда Common Language Runtime, выполняющая код на разных поддерживаемых языках программирования.

### 2.1.3 Язык программирования C#

В качестве языка программирования используется C# – основной язык разработки в .NET Framework. Язык объектно-ориентированный, имеет строгую статическую типизацию, поддерживает перегрузку операторов, указатели на функции-члены классов, атрибуты, события, свойства, исключения. Используется как основной язык в технологии WPF.

### 2.1.4 Технология WPF

Для предоставления пользовательского интерфейса и разграничения дизайна и бизнес-логики используется технология Microsoft WPF – аналог WinForms, система для построения клиентских приложений Windows с возможностями взаимодействия с пользователем и графическая подсистема в составе .NET, использующая язык разметки XAML.

### 2.1.5 Расширяемый язык разметки XAML

WPF предоставляет средства для создания визуального интерфейса, включая язык XAML (eXtensible Application Markup Language элементы управления, привязку данных, макеты, двухмерную и трёхмерную графику, анимацию, стили, шаблоны, документы, текст, мультимедиа и оформление. XAML представляет собой язык декларативного описания интерфейса, основанный на XML.

### 2.1.6 Технология Entity Framework

Для взаимодействия с базой данных в приложении WPF используется технология Entity Framework Core — кроссплатформерная и расширяемая версия технологии Entity Framework с открытым исходным кодом, которая в текущий момент развивается и дополняется.

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. Entity Framework Core – это последняя версия Entity Framework 7, которая совместима с .NET Core и на данный момент продолжает развиваться. Entity Framework Core предполагает три возможных способа взаимодействия с базой данных:

* Database First: Entity Framework Core создает набор классов, которые отражают модель конкретной базы данных;
* Model First: сначала разработчик создает модель базы данных, по которой затем Entity Framework Core создает реальную базу данных на сервере;
* Code First: разработчик создает класс модели данных, которые будут храниться в базе данных, а затем Entity Framework Core по этой модели генерирует базу данных и ее таблицы.

### 2.1.7 MS SQL Server

Для организации баз данных MS SQL Server использует реляционную модель, которая предполагает хранение данных в виде таблиц, каждая из которых состоит из строк и столбцов. Каждая строка хранит отдельный объект, а в столбцах размещаются атрибуты этого объекта. Для взаимодействия с базой данных применяется язык SQL (Structured Query Language). Клиент (например, внешняя программа) отправляет запрос 10 на языке SQL должным образом интерпретирует и выполняет запрос, а затем посылает клиенту результат выполнения. Основной используемый язык запросов — Transact-SQL — реализован на структурированном языке запросов (SQL) с расширениями.

### 2.1.8 Паттерн MVVM

MVVM (Model-View-ViewModel) - это паттерн архитектуры, используемый в разработке программного обеспечения для разделения пользовательского интерфейса (View) от бизнес-логики (Model) и связывания их через промежуточный слой ViewModel.

В MVVM модель представляет данные и бизнес-логику приложения, которая может включать в себя операции чтения/записи данных из и в источники данных, такие как база данных или веб-сервисы.

Представление (View) отвечает за отображение данных и взаимодействие с пользователем. Оно не содержит бизнес-логику и должно быть максимально независимым от модели.

ViewModel представляет промежуточный слой между View и Model. Он содержит логику, необходимую для обработки пользовательских действий, обновления данных в модели и уведомления View об изменениях. ViewModel предоставляет свойства и команды, которые привязываются к элементам пользовательского интерфейса во View, позволяя им отображать данные и реагировать на действия пользователя.

MVVM позволяет достичь разделения ответственности между компонентами приложения, облегчает тестирование и повышает переиспользуемость кода. Он также способствует лучшей поддержке параллельной разработки пользовательского интерфейса и бизнес-логики, так как разработчики могут работать независимо над своими частями приложения.

## 2.2 Спецификация функциональных требований к программному средству

Программное средство должно предоставлять следующие функциональные возможности:

Для пользователя:

* регистрация;
* авторизация;
* просмотр каталога копировальной техники;
* фильтрация и поиск техники;
* оформление заявки на покупку или ремонт оборудования;
* редактирование личных данных в профиле;
* возможность оставить отзыв о товаре или предоставленной услуге;
* просмотр отзывов других пользователей.

Для администратора:

* авторизация в системе;
* просмотр данных зарегистрированных пользователей;
* управление каталогом товаров и услуг;

Функции менеджера:

* авторизация в системе;
* просмотр всех заявок (включая отменённые).
* оперативная обработка заявок на ремонт и покупку;
* изменение статуса заявок, удаление или передача в работу;
* просмотр статистики и аналитики по продажам и ремонтам;
* управление и модерация отзывов клиентов.

## 2.3 Спецификация функциональных требований

Для функциональности ПС необходимо создание базы данных для хранения информации приложения. Подробно о базе данных описано в главе 3.

В программном средстве при запуске необходимо реализовать регистрацию и авторизацию пользователей для дальнейшего использования приложения. Для авторизации входными параметрами являются логин и пароль пользователя, которые содержатся в базе данных. Для того чтобы зарегистрироваться необходимо ввести информацию клиента. Введенные данные, успешно прошедшие валидацию, заносятся в базу данных, а пользователь получает доступ к приложению.

При авторизации пользователь перемещается на главную страницу приложения. На всех страницах размещается навигационное меню, с помощью которого клиент может перемещаться по приложению. На странице «Домой» пользователь имеет возможность изучить доступные товары и сервисы, а также добавить их в корзину. На странице «Профиль» пользователь может изучить информацию о своём профиле, а также поменять в нём данные. На странице «Корзина» пользователь может изучить список корзины, а также совершить заказ. В разделе «Комментарии» на странице товара или услуги пользователь может просмотреть комментарии других пользователей и оставить свой отзыв.

Администратор после входа в систему получает доступ к навигации по приложению, аналогичной пользовательской. В его распоряжении находятся разделы для управления содержимым и пользователями. В частности, администратор может работать с таблицами, содержащими информацию о зарегистрированных пользователях и доступных товарах и услугах.

Менеджер после авторизации получает доступ к специализированному функционалу, связанному с обработкой заявок и взаимодействием с клиентами. Менеджеру доступны разделы с таблицами заявок на покупку и ремонт, где он может просматривать информацию, изменять статусы заявок (например, «В обработке», «Выполнена», «Отменена»), а также удалять их при необходимости. Дополнительно менеджер имеет доступ к просмотру аналитических данных и управлению отзывами пользователей.

# 3 Проектирование программного средства

## 3.1 Общая структура

Программное средство «Printinvest» имеет следующею структуру, представленную на рисунке 3.1.

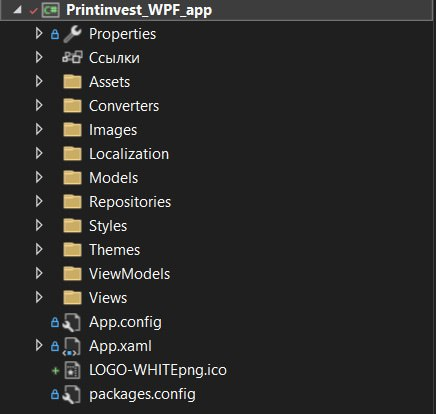


Рисунок 3.1 – Структура проекта

Описание структуры основных папок и файлов проекта и библиотеки классов представлено в таблице 3.1.

Таблица 3.1 – Описание структуры папок и файлов проекта

|  |  |
| --- | --- |
| Имя файла/папки | Содержание |
| App.config | Файл с параметрами проекта. |
| App.xaml | Определение глобальных ресурсов |
| Папка Assets | Ресурсы проекта |
| Папка Converters | Конвертеры |
| Папка Images | Ресурсы фотографий |
| Папка Localization | Ресурсы локализации |
| Папка Models | Данные и состояние приложения |
| Папка Repositories | Репозитории для управления данными |
| Папка Styles | Ресурсы стилей |
| Папка Themes | Ресурсы тем |
| Папка ViewModels | Классы модели представления для окон |
| Папка View | XAML разметка окон приложения |

Таким образом, сформированная таблица помогает понять общую стркутура проекта проектируемого программного средства

Cтруктура папки Models (сущностей приложения) представлена на рисунке  3.2.

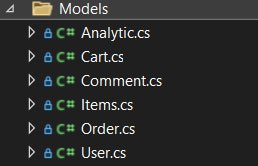


Рисунок 3.1 – Структура папки «Model»

Боллее подробное описание структуры и файлов данной папки представлено в таблице 3.2

Таблица 3.1 – Описание файлов папки Model

|  |  |
| --- | --- |
| Имя файла | Содержание |
| Analytic.cs | Класс Analytic для хранения аналитики действий пользователей |
| Cart.cs | Класс Cart для хранения содержимого корзин пользователей (CartItem, как позиция корзины) |
| Comment.cs | Класс Comment для хранения комментариев пользователей |
| Items.cs | Классы Product и Services для хранения товаров и услуг соответственно |
| Orders.cs | Классы Order для хранения заказов и его содержимое (OrderItem, как позиция заказов) |
| User.cs | Класс User для определения хранения разных типов аккаунта и разделения по ролям |

Описание структуры проекта позволяет лучше понимать, как устроено программное средство и какие компоненты в нем присутствуют.

## 3.2 Взаимоотношение между классами

Для визуализации взаимосвязей между классами используется диаграмма UML – графическое представление набора элементов, изображаемое чаще всего в виде связанного графа с вершинами (сущностями) и ребрами (отношениями).

Для представления внутренней структуры программы в виде классов и связей между ними используется диаграмма классов. Приложение спроектировано таким образом, что каждый класс выполняет свои функции и практически не зависит от других. Диаграмма классов представлена в приложении А.

## 3.3 Модель базы данных

Для реализации поставленной задачи была создана база данных Printinvest. Для ее создания использовалась система управления реляционными базами данных MS SQL Server. База данных состоит из таблиц, представленных на рисунке 3.2.

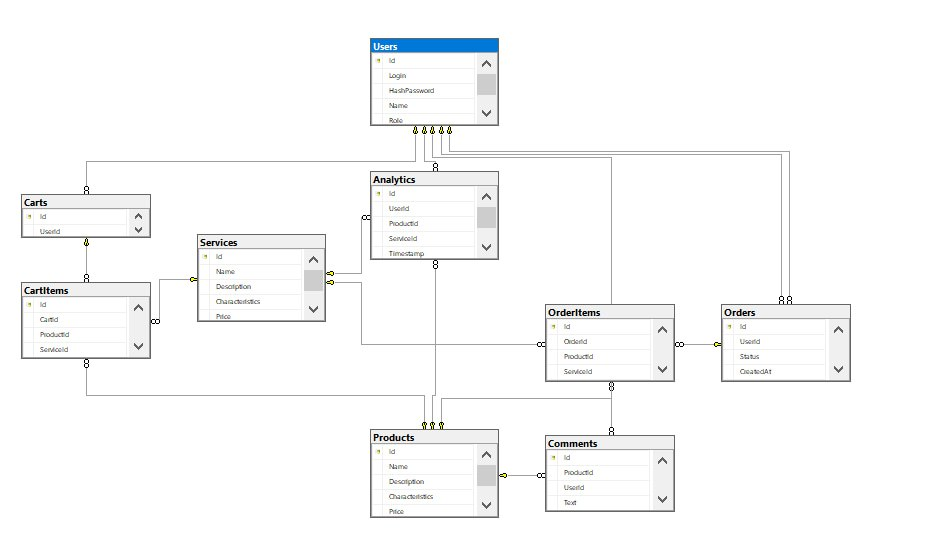


Рисунок 3.2 — База данных Printinvest

На рисунке 3.3 представлена структура таблицы «Analytics», которая содержит информацию об аналитике. В данной таблице есть Id — целочисленное поле для идентификации записи в базе данных. UserId — идентификатор пользователя. ProductId — идентификатор продукта. ServiceId — идентификатор услуги. Timestamp — дата и время записи. Action — описание действия.

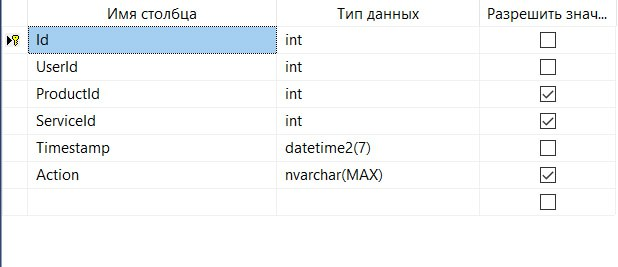


Рисунок 3.3 — Структура таблицы «Analytics»

На рисунке 3.4 представлена структура таблицы «CartItem», которая содержит информацию о товарах в корзине. В данной таблице есть Id — целочисленное поле для идентификации записи в базе данных. CartId — идентификатор корзины. ProductId — идентификатор продукта. ServiceId — идентификатор услуги. Quantity — количество товара..

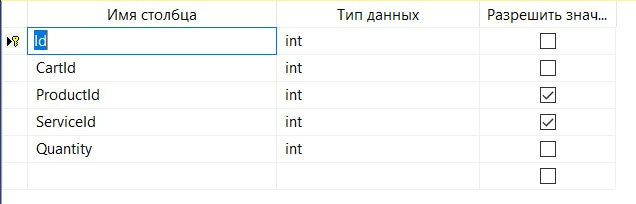


Рисунок 3.4 — Структура таблицы «CartItem»

На рисунке 3.5 представлена структура таблицы «Cart», которая содержит информацию о корзинах. В данной таблице есть Id — целочисленное поле для идентификации записи в базе данных. UserId — идентификатор пользователя. ProductId — идентификатор продукта. ServiceId — идентификатор услуги. Quantity — количество товара.

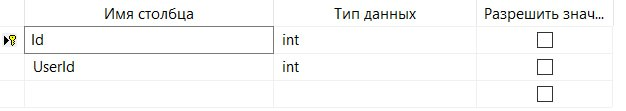


Рисунок 3.5 — Структура таблицы «Cart»

На рисунке 3.6 представлена структура таблицы «Comments», которая содержит информацию о комментариях. В данной таблице есть Id — целочисленное поле для идентификации записи в базе данных. UserId — идентификатор пользователя. Text — текст комментария. Timestamp — дата и время создания.

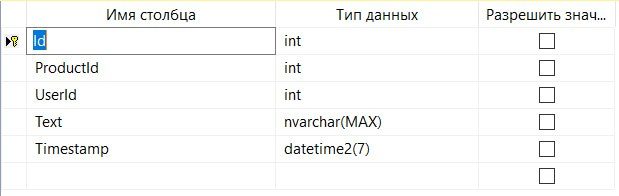


Рисунок 3.6 — Структура таблицы «Comments»

На рисунке 3.7 представлена структура таблицы «OrderItems», которая содержит информацию о позициях заказа. В данной таблице есть Id — целочисленное поле для идентификации записи в базе данных. OrderId — идентификатор заказа. ProductId — идентификатор продукта. ServiceId — идентификатор услуги. Quantity — количество товара.

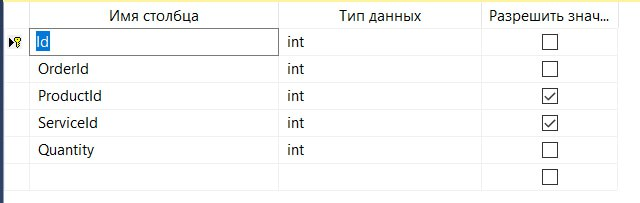


Рисунок 3.7 — Структура таблицы «OrderItems»

На рисунке 3.8 представлена структура таблицы «Orders», которая содержит информацию о заказах. В данной таблице есть Id — целочисленное поле для идентификации записи в базе данных. UserId — идентификатор пользователя. CreatedAt — дата и время создания заказа.

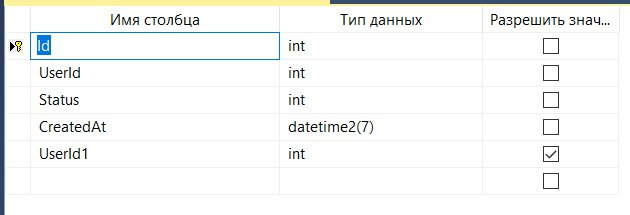


Рисунок 3.8 — Структура таблицы «Orders»

На рисунке 3.9 представлена структура таблицы «Products», которая содержит информацию о продуктах. В данной таблице есть Id — целочисленное поле для идентификации записи в базе данных. Name — название продукта. Description — описание продукта. Characteristics — характеристики продукта. Price — цена продукта. Photo — изображение в бинарном виде.

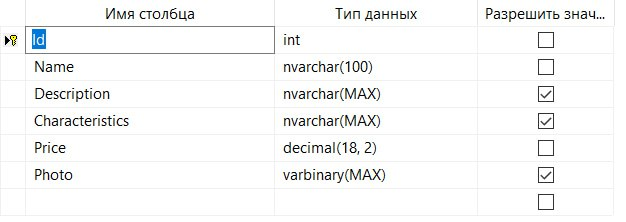


Рисунок 3.9 — Структура таблицы «Products»

На рисунке 3.10 представлена структура таблицы «Services», которая содержит информацию об услугах. В данной таблице есть Id — целочисленное поле для идентификации записи в базе данных. Name — название услуги. Description — описание услуги. Characteristics — характеристики услуги. Price — цена услуги. Photo — изображение в бинарном виде.

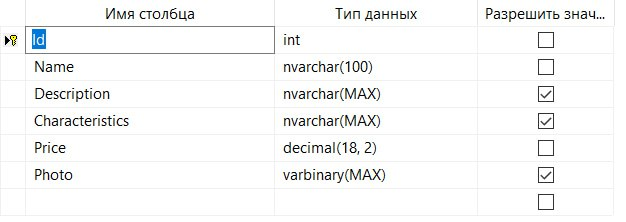


Рисунок 3.10 — Структура таблицы «Services»

На рисунке 3.11 представлена структура таблицы «Users», которая содержит информацию о пользователях. В данной таблице есть Id — целочисленное поле для идентификации записи в базе данных. Login — логин пользователя. HashPassword — хешированный пароль. Name — имя пользователя. Role — роль пользователя. Photo — изображение в бинарном виде.

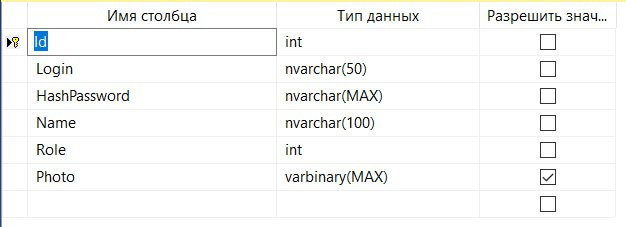


Рисунок 3.11 — Структура таблицы «Users»

В целом, структура базы данных "Printinvest" предоставляет необходимую основу для хранения и управления информацией о пользователях, товарах, заказах, корзин и комментариев в приложении "Printinvest".

## 3.4 Проектирование архитектуры приложения

Для общего представления функционального назначения системы используется диаграмма использования, описывающая, какой функционал разрабатываемой программной системы доступен каждой группе пользователей. На диаграмме использования применяются два типа основных сущностей: варианты использования и группы пользователей.

Разные группы пользователей в диаграмме называются актёрами, и обозначают любые сущности, использующие систему. Любая функция системы называется вариантом использования. Каждый вариант использования обозначает набор действий, который может быть использован актёром для взаимодействии с системой, и определяет набор действий, выполняемых этой системой.

Диаграмма использования представлена в приложении Б.

## 3.5 Проектирование последовательностей проекта

В целях визуализации взаимодействия объектов системы между собой во времени в едином сценарии использования используется ещё одна UML-диаграмма – диаграмма последовательностей. Данная диаграмма иллюстрирует, как различные части системы взаимодействуют друг с другом для выполнения функции, а также порядок, в котором происходит взаимодействие при выполнении конкретного случая использования.

Для отображения течения времени используется линия жизни объекта, которая изображается с помощью штриховой линии, которая проводится вертикально вниз. С помощью линии жизни показывается период, в течение которого объект существует в системе. Сами объекты изображаются в виде прямоугольников, а сообщения, которыми они обмениваются – в виде линий со стрелками.

Пример диаграммы последовательности представлен в приложении В.

# 4 Реализация программного средства

## 4.1 Основные классы программного средства

Для выполнения технических задач программного средства «Printinvest» должны быть реализованы следующие функции и методы:

* авторизация;
* регистрация;
* добавление товара/услугу в корзину;
* создание заказа;
* добавление отзывов на товар/услугу.

## 4.2 Авторизация

Необходимый функционал для возможности входа пользователя в аккаунт, а также для проверки корректности данных, реализован в классе LoginViewModel.

Основным методом класса является метод ExecuteLogin, выполняющийся при нажатия пользователя на кнопку «Войти». В данном методе вызывается метод HashHelper.VerifyPassword, проверяющий введённый пользователем пароль с сохранённым в базе данных паролем.

При успешной проверке на существование введённого логина и пароля в базе данных пользователь перенаправляется на страницу профиля, а противном случае – получает сообщение о соответствующей ошибке.

Листинг реализации метода ExecuteLogin и класса HashHelper представлен в приложении Г.

## 4.3 Регистрация

Для совершения пользователем регистрации нового аккаунта реализован метод RegisterExecute класса RegisterViewModel.

При нажатии пользователя на кнопку «Зарегистрироваться» все введённые пользователем данные проходят проверку на корректность. После успешного прохождения проверок происходит попытка добавить пользователя в базу данных и перенаправление пользователя на страницу профиля. Если логин уже зарегистрирован в базе данных, то пользователь получает соответствующее уведомление об ошибке.

Листинг реализации метода RegisterExecute класса представлен в приложении Г.

## 4.4 Добавление товара/услугу в корзину

Для добавления товара или услуги в корзину пользователь переходит на главную страницу (каталог), где отображаются все доступные позиции. При нажатии кнопки «В корзину» вызывается метод ExecuteAddToCart(), который сохраняет выбранный товар или услугу в таблицы БД.

Перед выполнением добавления система проверяет, есть ли активный пользователь. Если нет активного пользователя, пользователю выводится соответствующее уведомление и открывается страница авторизации. В противном случае товар успешно добавляется, и отображается сообщение об успешном действии.

Добавление и удаление товаров или услуг осуществляется с помощью взаимодействия с контекстом базы данных, в котором фиксируются действия пользователя.

Листинг метода ExecuteAddToCart() представлен в приложении Г.

## 4.5 Создание заказа

После добавления всех необходимых товаров и услуг в корзину пользователь может перейти к оформлению заказа. При нажатии кнопки «Оформить заказ» вызывается метод PlaceOrder(), который формирует новый заказ на основе содержимого корзины.

В процессе оформления производится проверка корректности введённых данных. После подтверждения заказ сохраняется в базе данных, корзина очищается, и пользователю отображается уведомление об успешном оформлении.

Листинг метода PlaceOrder() приведён в приложении Г.

## 4.6 Добавление комментариев

В разделе «Комментарии» пользователь может просмотреть комментарии других пользователей и оставить свой. Функционал для этого предоставляет метод ExecuteAddComment().

Метод сразу проверит наличие вошедшего в систему пользователя, и в случае, если его не существует выведет окно ошибки и переадресует на страницу входа. Далее из интерфейса метод получит значение текстового поля и создаст объект контекста базы данных с указанием пользовательского идентификатора, после чего поле ввода очистится и пользователь может повторить процедуру добавления комментария.

Листинг реализаций команды ExecuteAddComment() представлен в приложении Г.

## 5 Тестирование, проверка работоспособности и анализ полученных результатов

## 5.1 Тестирование авторизации

При попытке авторизации пользователя может возникнуть ситуация, когда тот ввел неверный пароль, который соответствует логину или же ввел логин, которого не существует, в таком случае система оповестит пользователя об ошибке. Результат тестирования представлен на рисунке 5.1.

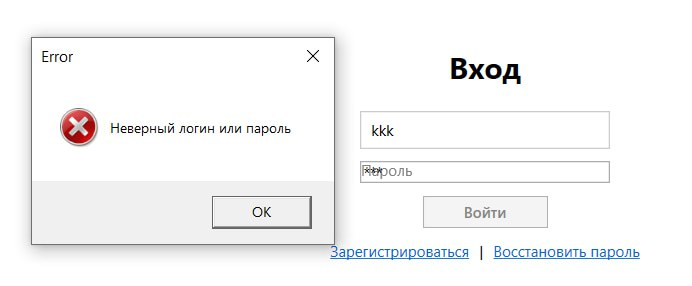


Рисунок 5.1 — Результат тестирования авторизации

Если есть пустые поля, то кнопка просто не будет активна.

## 5.2 Тестирование регистрации

При регистрации пользователь может совершить попытку использования уже занятого логина. Проводится тестирование данной ситуации, в базе данных уже существует запись с идентичным логином. Результат данного теста представлен на рисунке 5.2.

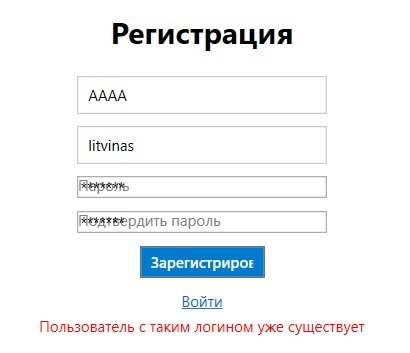


Рисунок 5.2 — Результат первого тестирования регистрации

При регистрации пользователь может ввести некорректные данные в поля формы или же оставить пустое поле. Проводится тест данного случая – пользователь вел некорректно данные при регистрации. Результат данного тестирования представлен на рисунке 5.3.

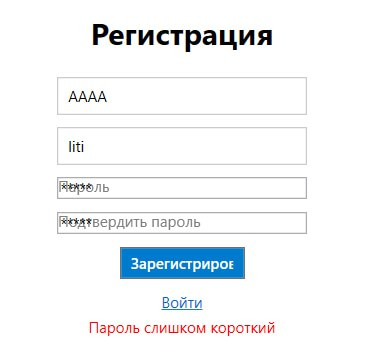


Рисунок 5.3 — Результат второго тестирования регистрации

Если есть пустые поля, то кнопка просто не будет активна.

## 5.3 Тестирование добавление товара в корзину

При добавлении товара в корзину при неактивном пользователе, приложение выведет сообщение, о том, что надо авторизироваться, после чего переадресует на окно входа. Результат данного тестирования представлен на рисунке 5.4.

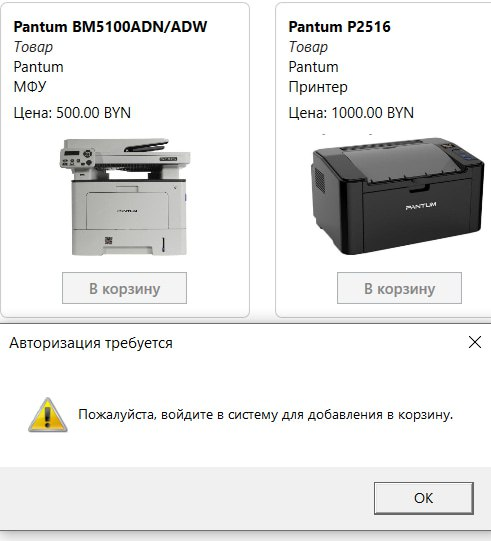


Рисунок 5.4 — Результат тестирования добавления товара в корзину

## 5.4 Тестирование редактирования профиля

У пользователя есть возможность редактировать данные в своем профиле такие как: логин, имя. При корректном изменение, данные будут успешно сохранены. Однако, возможна такая ситуация, когда данные введены не корректно (не прошли валидацию, такую же, как и при регистрации) или же исправленные данные такие как: логин совпал с другим пользователем, в таком случае пользователю выдаст предупреждение о том, что данные совпали или неверны и произойдет откат на прежние значения. Тестирование этой ситуации представлено на рисунке 5.5.

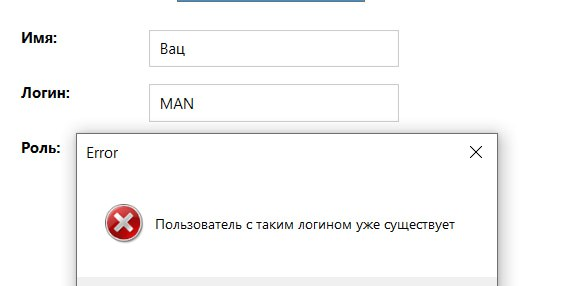


Рисунок 5.5 — Результат тестирования редактирования профиля

Если есть пустые поля, то кнопка просто не будет активна.

## 5.5 Тестирование добавления комментариев

Пользователи могут оставить свои комментарии в приложении, для этого им нужно ввести текст комментария в поле ввода, в случае если поле осталось пустым пользователю выведется сообщение об ошибке. Тестирование этой ситуации представлено на рисунке 5.6.

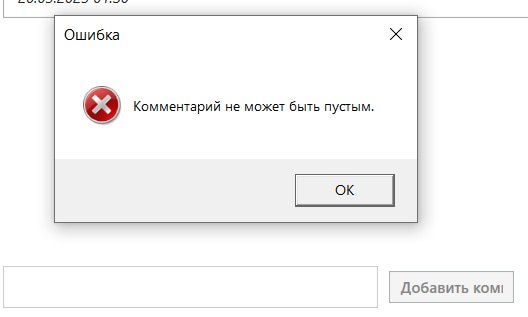


Рисунок 5.6 — Результат тестирования добавления комментариев

Если пользователь без активного профеля попробует оставить комментарий, появится сообщение, о том что надо авторизироваться, после чего переадресует на окно входа. Результаты тестирования на рисунке 5.7.

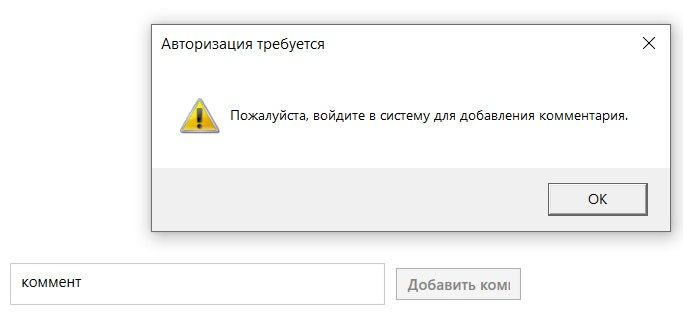


Рисунок 5.7 — Результат второго тестирования добавления комментариев

## 5.6 Тестирование оформления заказа

При попытке оформления заказа в пустой корзине, высветиться соответствующее сообщение об ошибке. Результат тестирования представлен на рисунке 5.8.

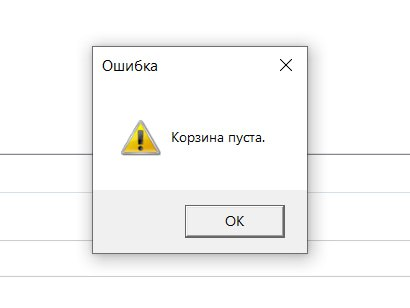


Рисунок 5.8 — Результат тестирования оформления заказа

При попытке оформить заказ с пустыми полями формы, высветиться соответствующее сообщение об ошибке. Результат тестирования представлен на рисунке 5.9.

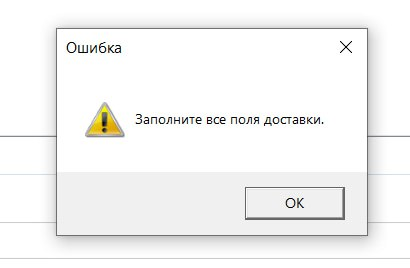


Рисунок 5.9 — Результат второго тестирования оформления заказа

В ходе тестирование функций пользователя не было обнаружено проблем, каждый из тестов завершился успехом.

## 5.6 Тестирование добавления товара/услуг у администратора

Одной из функций администратора является добавление товара или услуг. Для этого он должен заполнить форму добавления. Для добавления необходимо чтобы все поля были заполнены и удовлетворяли условиям валидации (кроме фотографии). Тестирование добавления товара представлено на рисунке 5.10

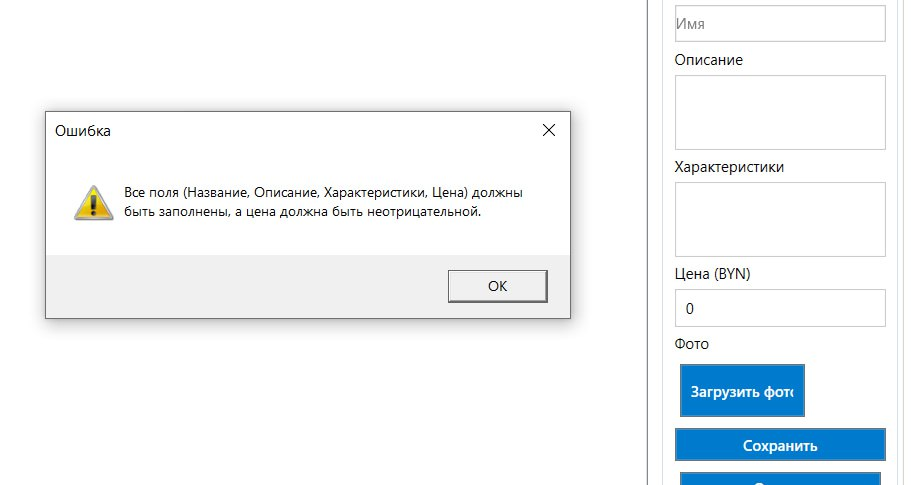


Рисунок 5.10 — Результат тестирования добавления товара

## 5.7 Тестирование удаления элементов администратором или менеджером

Главными функциями администратора и менеджера является удаление данных. При удалении данных, строчка данных должна быть выбрана, иначе будет высветлена ошибка. Тестирование добавления тренировки представлено на рисунке 5.11

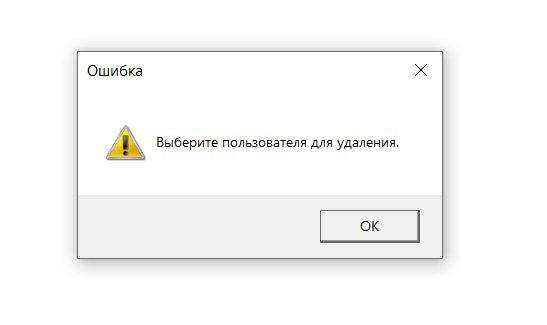


Рисунок 5.11 — Результат тестирования удаления пользователя

# 6 Руководство по установке и использованию

При запуске приложения (или по нажатию на кнопку «Главная») будет открыто окно каталога, которое показано на рисунке 6.1. Справа навигационное меню для удобного передвижения между страницами.

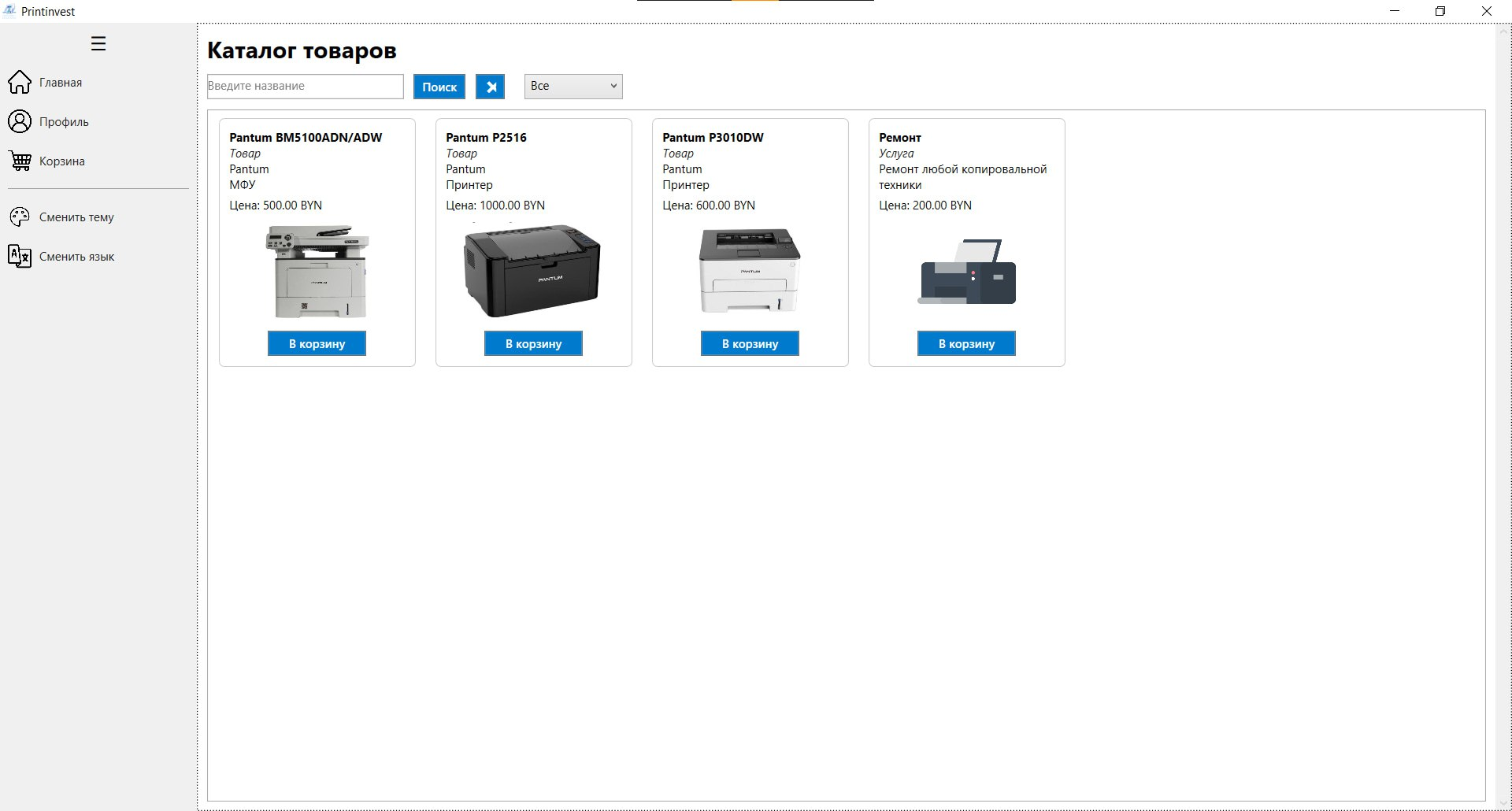


Рисунок 6.1 — Окно «Главнвная»

При нажатии на кнопку профиля, когда нет активного пользователя, происходит адресация на окно авторизации (рисунок 6.2)

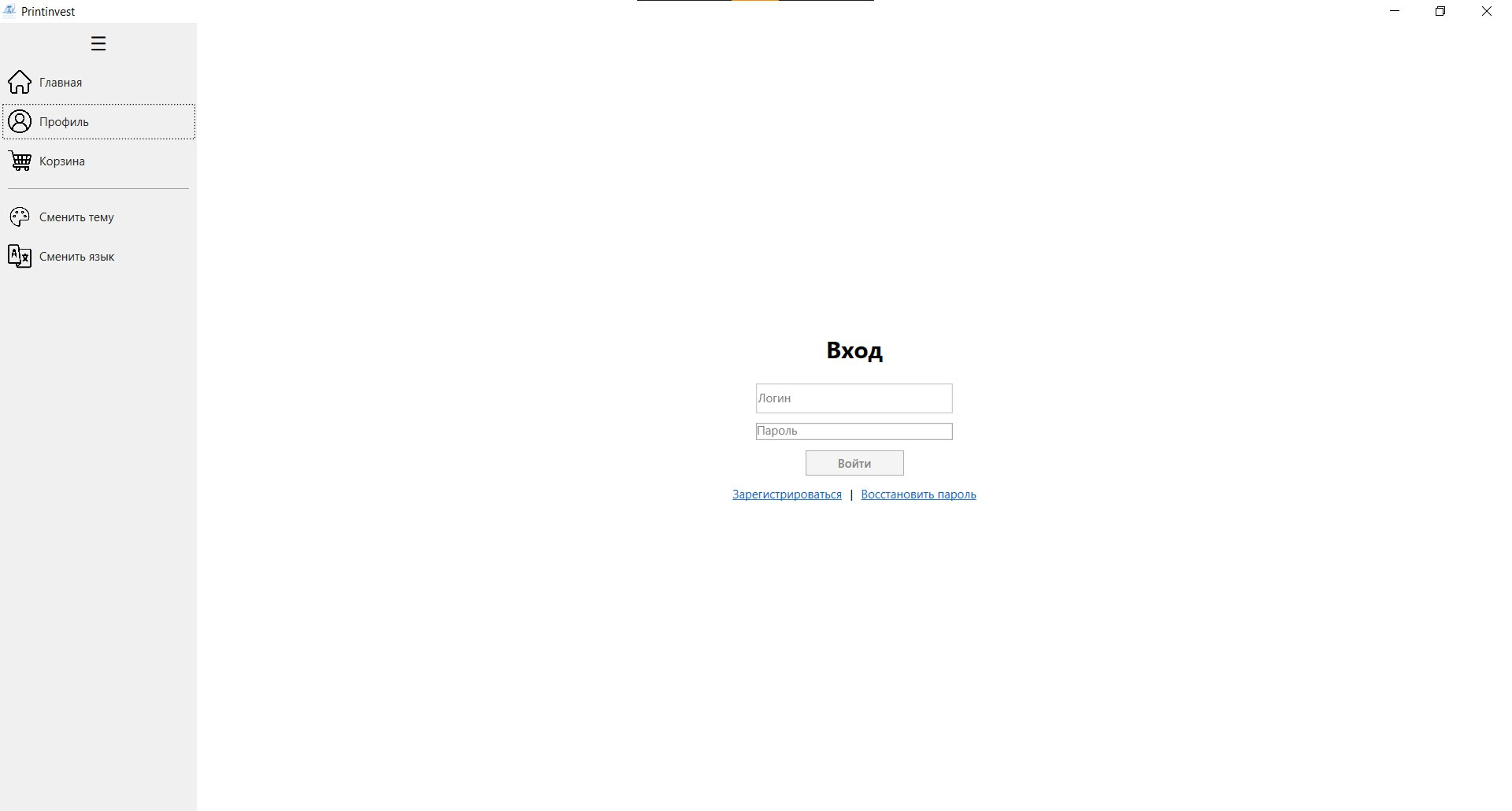


Рисунок 6.2 — Окно авторизации

Если у клиента не аккаунта, он может нажать на надпись «Зарегистрироваться» для того, чтобы создать свой аккаунт (рисунок 6.3).

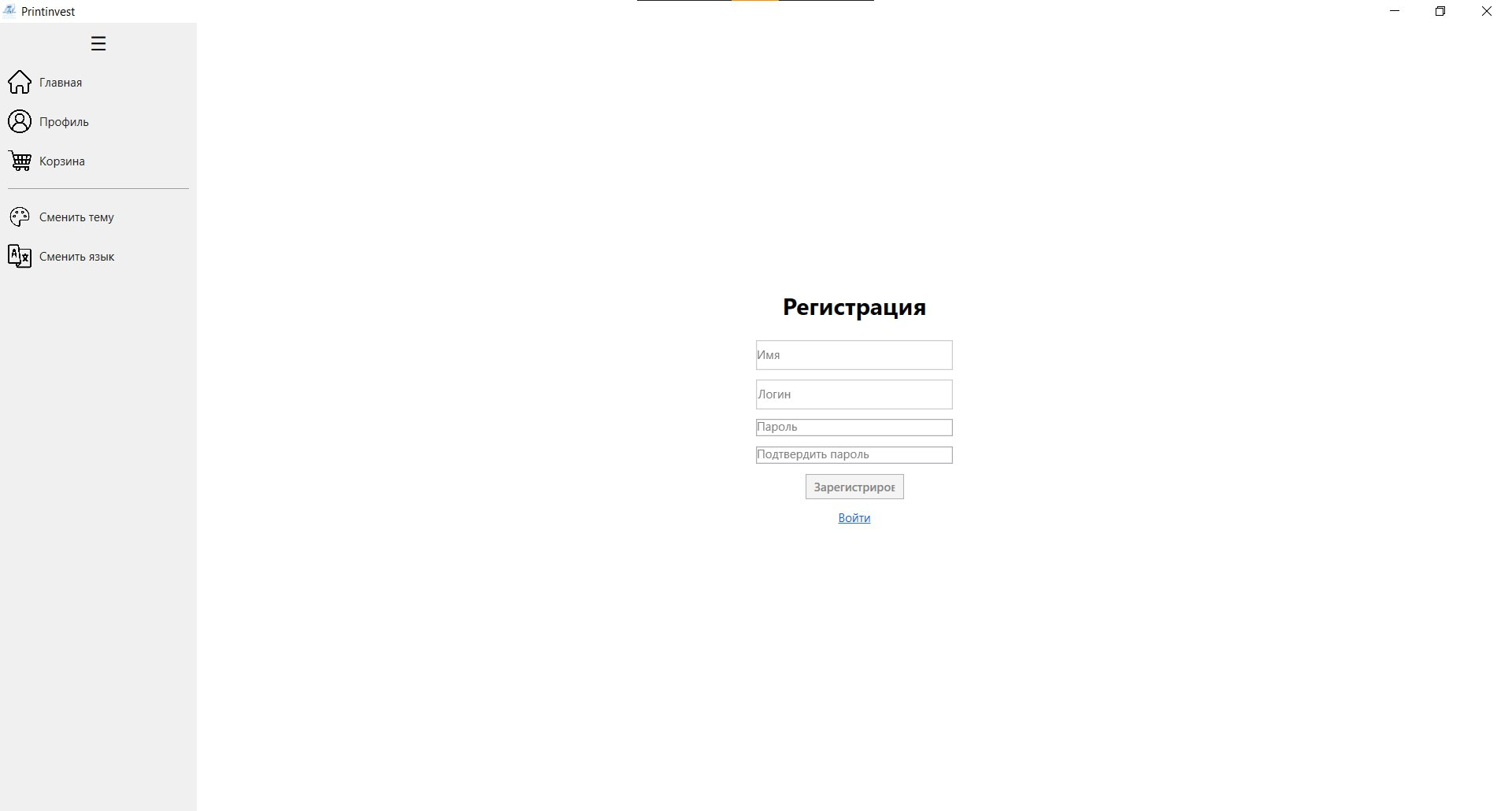


Рисунок 6.3 — Окно регистрации

После нажатия на кнопку регистрации откроется окно «Регистрация». Это окно представлено на рисунке 6.3. Поля не должны быть пустыми, логин должен быть уникальным, пароль не короче 6 символов, а поля пароль и подтвердить пароль совпадать.

После успешной регистрации пользователя перекинет на страницу профиля (рисунок 6.4).

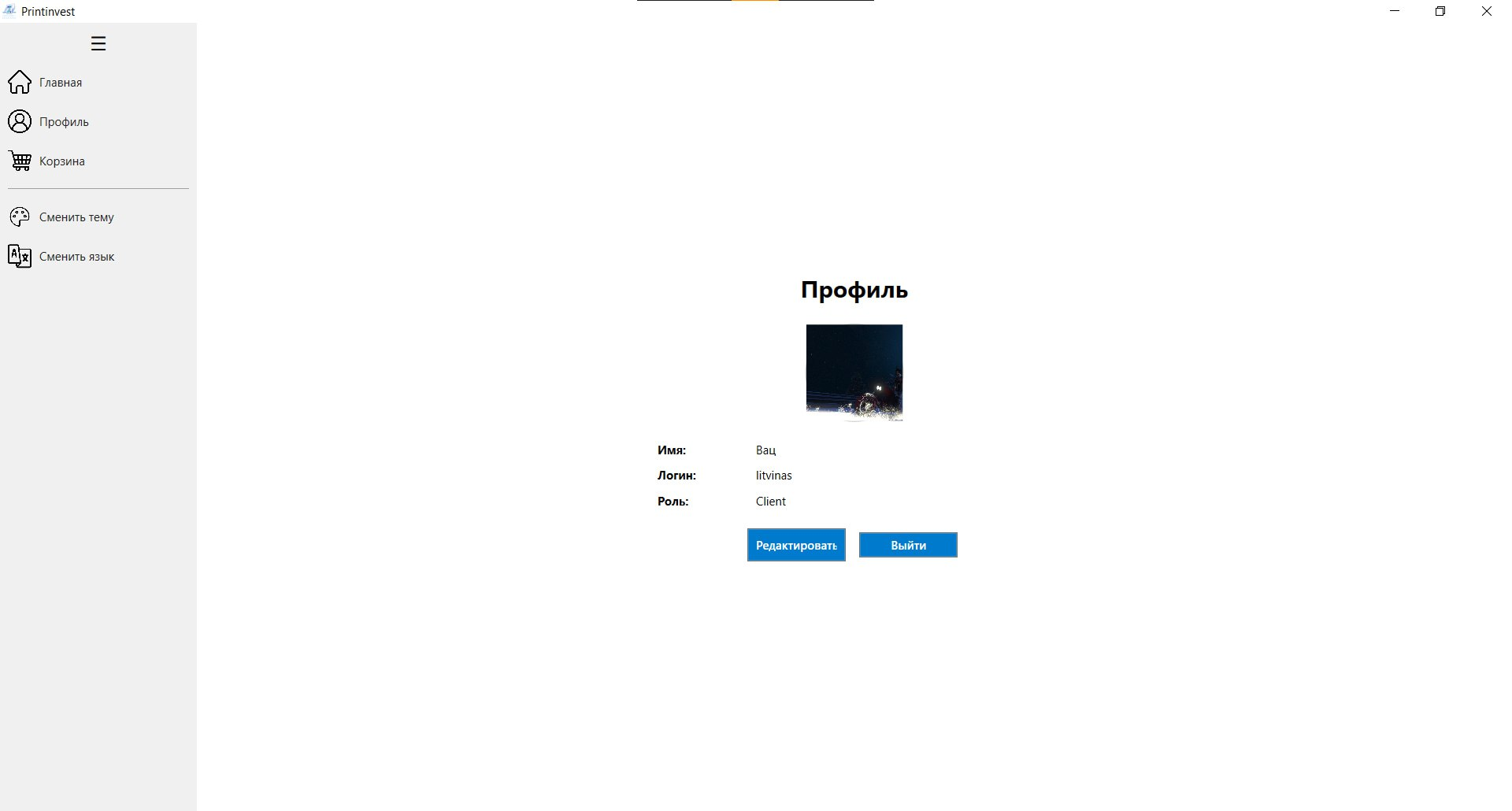


Рисунок 6.3 — Страница профиля

На странице «Профиль» пользователь может изменить свои данные,

а также изменить стандартную фотографию на свою.

По нажатию на товар на главной странице, откроется страница товара. На данной странице пользователь может добавить товар в корзину и оставить комментарий. Страница «Детали товара» представлена на рисунке 6.4.

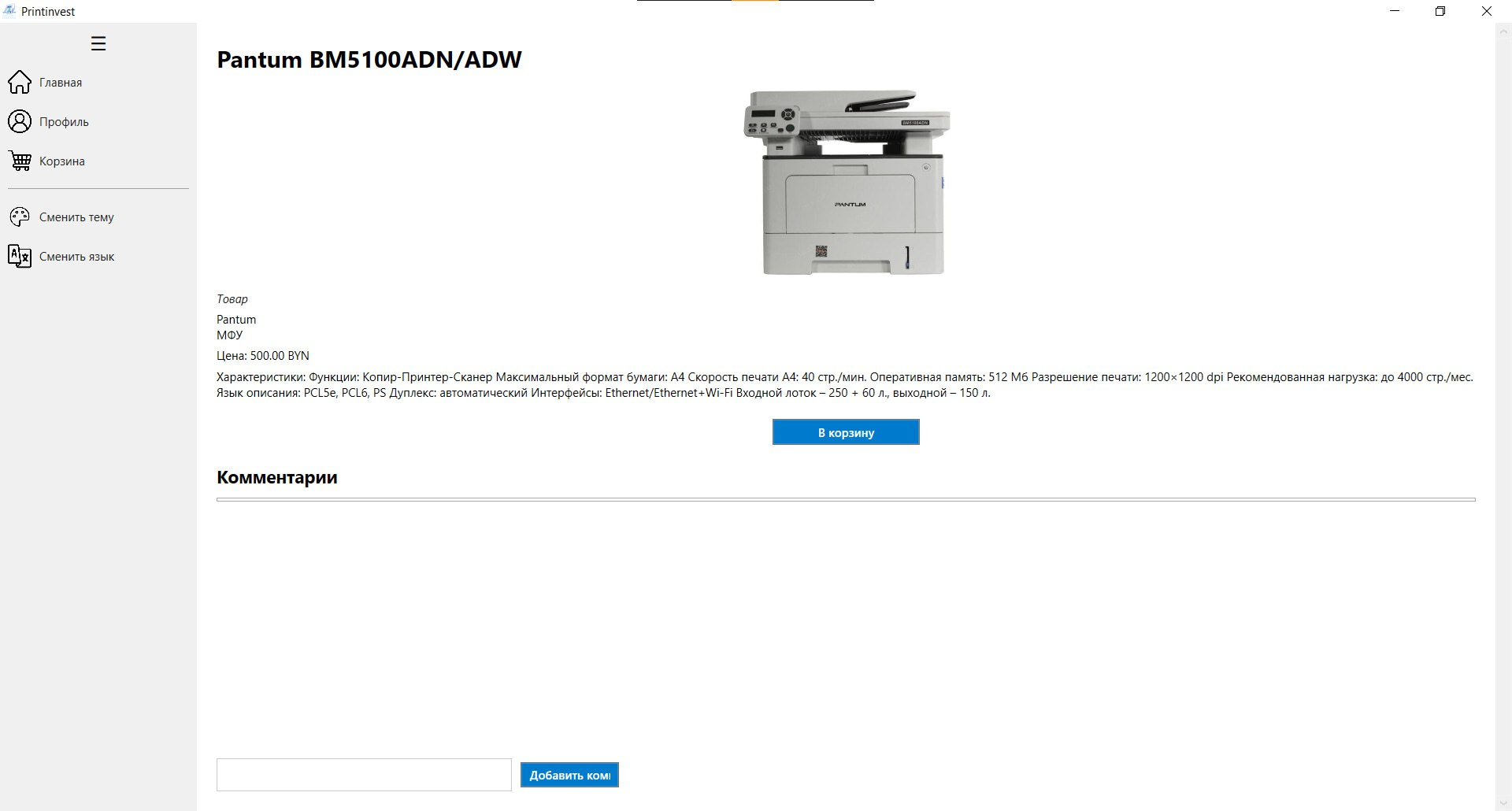


Рисунок 6.4 — Страница «Детали товара»

На странице «Корзина» выведена информация о содержимом корзине. Так же на странице присутствует форма оформления заказа. По нажатию кнопки «оформить заказ», происходит валидация данных, и если всё нормально, то происходит оформление заказа. В противном случае выводиться сообщение о соответствующей ошибке. Страница «Корзина» представлена на рисунке 6.5.

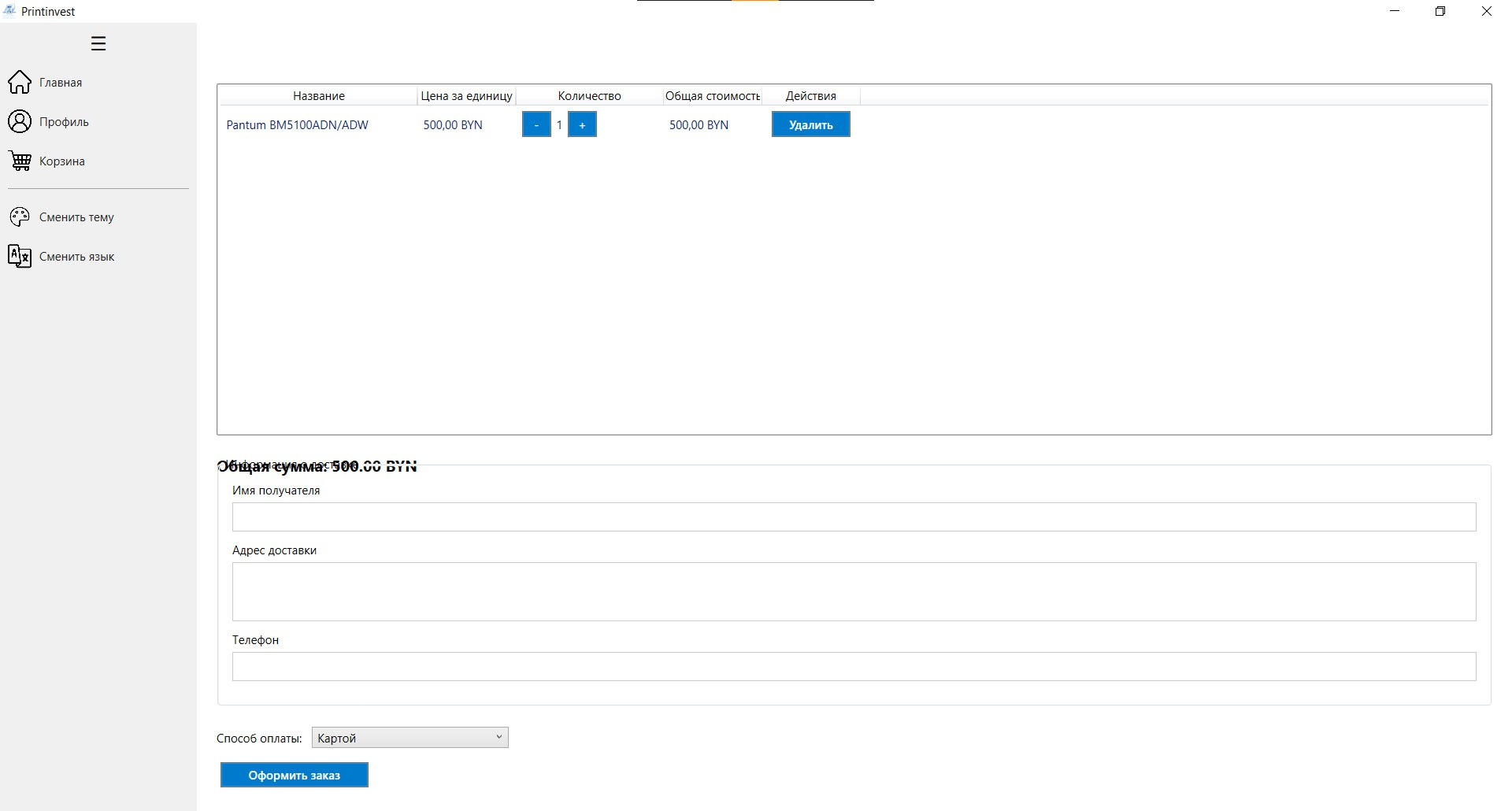


Рисунок 6.6 — Страница «Oнлайн тренировка»

Приложение «Printinvest» обеспечивает удобный пользовательский интерфейс, для изучения информации о товарах и услугах, добавления в корзину и оформления заказов.

# ЗАКЛЮЧЕНИЕ

В рамках выполнения курсового проекта было разработано программное средство для компании Printinvest, специализирующейся на продаже и ремонте копировальной техники. Основной целью работы являлось создание удобного клиентского приложения, позволяющего просматривать каталог оборудования, оформлять заявки на покупку и ремонт, а также административной и менеджерской частей для управления бизнес-процессами.

В результате выполнения проекта были достигнуты следующие ключевые результаты:

* разработан интуитивно понятный интерфейс для взаимодействия клиентов с ассортиментом и заявками;
* реализован механизм авторизации и регистрации, личного кабинета;
* обеспечена функциональность добавления товаров и услуг в корзину, создания заказа;
* создана административная панель для управления каталогом, пользовательскими данными;
* разработан раздел менеджера для оперативной обработки заявок, изменения их статусов и работы с отзывами;
* выполнено тестирование основных сценариев работы системы, подтверждающее корректность бизнес-логики и стабильность приложения.

Для реализации проекта был выбран язык C# и технология Windows Presentation Foundation (WPF), обеспечившие создание современного и отзывчивого пользовательского интерфейса. В качестве СУБД использован Microsoft SQL Server для надёжного хранения информации о пользователях, товарах, заявках и отзывах. Архитектурно приложение построено с использованием паттерна MVVM (Model-View-ViewModel), что позволило чётко разделить представление, логику и данные, упростить сопровождение и расширение функциональности.

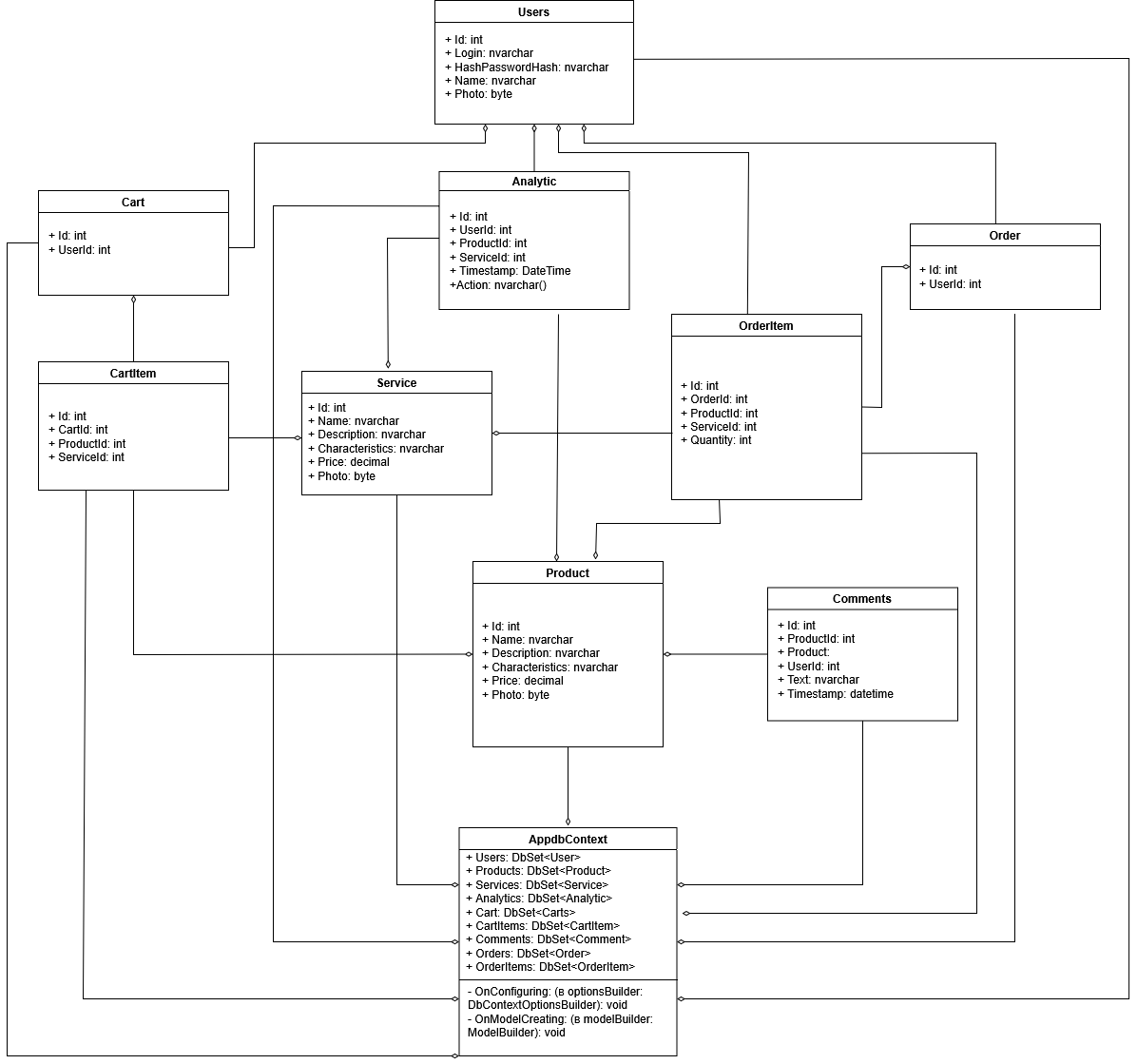
Таким образом, разработанное программное средство Printinvest соответствует заявленным требованиям, обеспечивает удобство работы клиента, администратора и менеджера, и готово к дальнейшему развитию и интеграции дополнительных модулей в соответствии с потребностями компании.

# Список использованных источников

1. Microsoft Visual Studio [Электронный ресурс] – https://ru.wikipedia.org/wiki/Microsoft\_Visual\_Studio – Дата доступа 23.04.2023
2. Полное руководство по языку программирования C# 7.0 и платформе .NET 4.7. Режим доступа: https://metanit.com/sharp/tutorial/ – Дата доступа: 23.04.2023
3. Пацей, Н.В. Курс лекций по языку программирования C# / Н. В. Пацей. – Минск: БГТУ, 2018. – 175 с.
4. Руководство по WPF // [Электронный ресурс]. – Режим доступа: https://metanit.com/sharp/wpf/ – Дата доступа: 25.04.2023
5. Руководство по XAML // [Электронный ресурс]. – Режим доступа: https://www.tutorialspoint.com/xaml/index.htm– Дата доступа: 25.04.2023

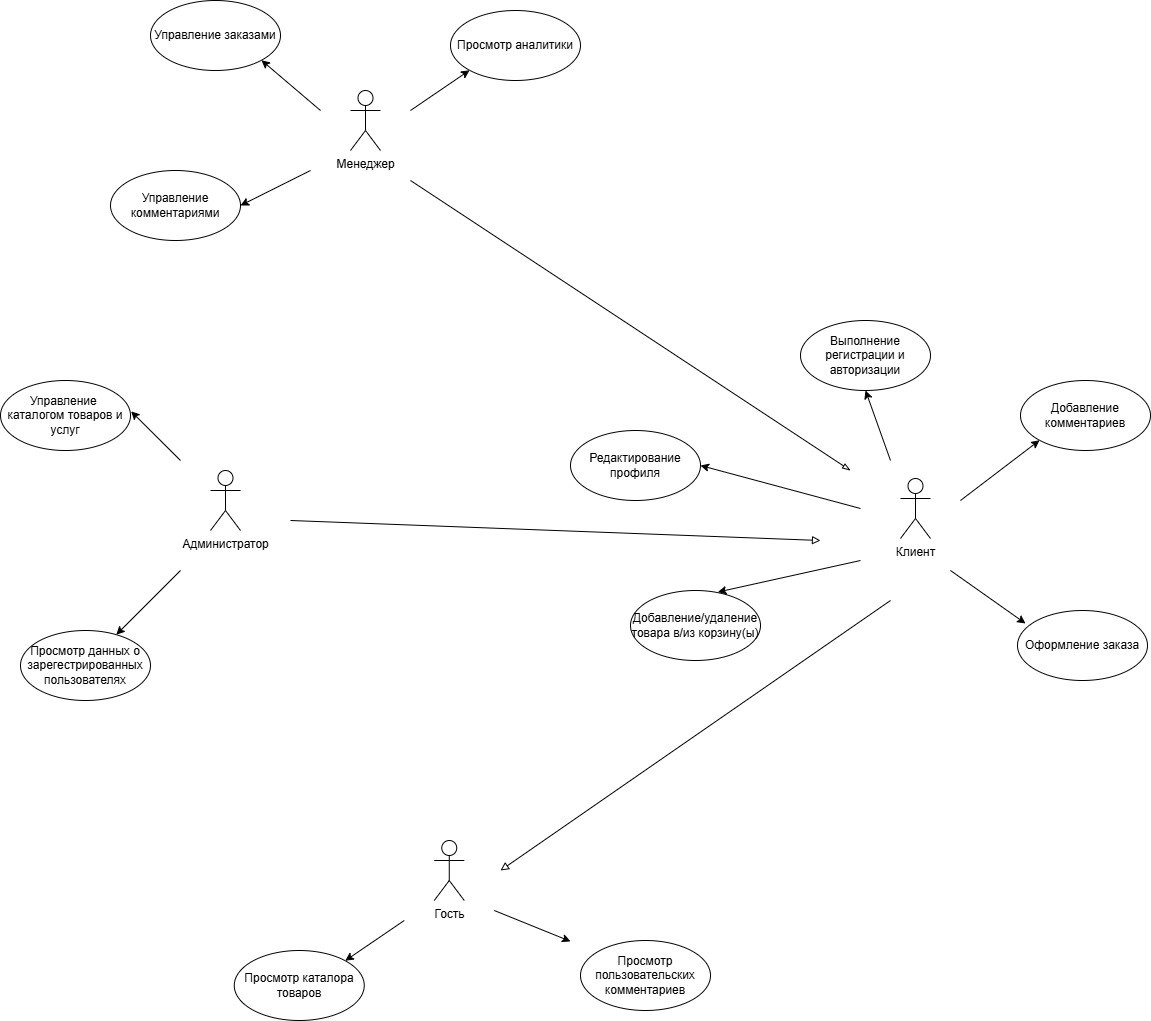
# ПРИЛОЖЕНИЕ А

Диаграмма классов



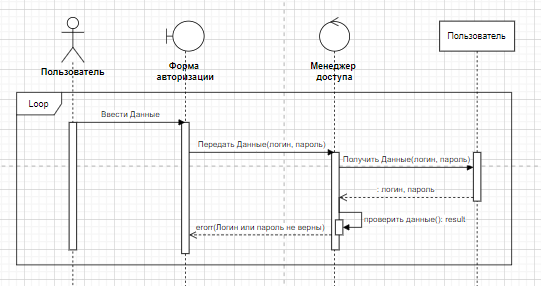
# ПРИЛОЖЕНИЕ Б

Диаграмма использования



# ПРИЛОЖЕНИЕ В

Диаграмма последовательности



# ПРИЛОЖЕНИЕ Г

Листинг метода ExecuteLogin

|  |
| --- |
| private void ExecuteLogin(object parameter)  {  User user = Repositories.RepositoryManager.Users.GetByLogin(Login);  if (user == null || !HashHelper.VerifyPassword(Password, user.HashPassword))  {  MessageBox.Show(Application.Current.TryFindResource("ErrorInvalidCredentials")?.ToString() ?? "Invalid login or password",  "Error", MessageBoxButton.OK, MessageBoxImage.Error);  return;  }  var mainViewModel = Application.Current.MainWindow?.DataContext as MainViewModel;  SessionManager.Login(user);  mainViewModel.CurrentPage = new ProfilePage();  } |

Листинг класса HashHelper

|  |
| --- |
| public static class HashHelper  {  /// <summary>  /// Создаёт хеш пароля с использованием BCrypt.  /// </summary>  /// <param name="password">Пароль для хеширования.</param>  /// <returns>Хеш пароля.</returns>  public static string HashPassword(string password)  {  if (string.IsNullOrWhiteSpace(password))  {  throw new ArgumentException("Пароль не может быть пустым.", nameof(password));  }  return BCrypt.Net.BCrypt.HashPassword(password);  }  /// <summary>  /// Проверяет, соответствует ли пароль указанному хешу.  /// </summary>  /// <param name="password">Пароль для проверки.</param>  /// <param name="hashedPassword">Хеш пароля для сравнения.</param>  /// <returns>True, если пароль соответствует хешу; иначе False.</returns>  public static bool VerifyPassword(string password, string hashedPassword)  {  if (string.IsNullOrWhiteSpace(password))  {  throw new ArgumentException("Пароль не может быть пустым.", nameof(password));  }  if (string.IsNullOrWhiteSpace(hashedPassword))  {  throw new ArgumentException("Хеш пароля не может быть пустым.", nameof(hashedPassword));  }  return BCrypt.Net.BCrypt.Verify(password, hashedPassword);  }  } |

Листинг метода RegisterExecute

|  |
| --- |
| private void RegisterExecute()  {  try  {  if (Password != ConfirmPassword)  {  ErrorMessage = Application.Current.Resources["ErrorPasswordsDoNotMatch"]?.ToString() ?? "Пароли не совпадают.";  return;  }  if (Password.Length < 6)  {  ErrorMessage = Application.Current.Resources["ErrorPasswordTooShort"]?.ToString() ?? "Пароль слишком короткий. Минимум 6 символов.";  return;  }  var existingUser = \_userRepository.GetByLogin(Login);  if (existingUser != null)  {  ErrorMessage = Application.Current.Resources["ErrorUserExists"]?.ToString() ?? "Пользователь с таким логином уже существует.";  return;  }  var newUser = new Models.User  {  Login = Login,  HashPassword = HashHelper.HashPassword(Password),  Name = Name,  Role = Models.UserRole.Client  };  \_userRepository.Add(newUser);  SessionManager.Login(newUser);  Navigate("Profile");  }  catch (Exception ex)  {  ErrorMessage = $"Ошибка: {ex.Message}";  }  } |

Листинг метода ExecuteAddToCart

|  |
| --- |
| private void ExecuteAddToCart(object parameter)  {  try  {  if (parameter is IItem item)  {  if (SessionManager.CurrentUser == null)  {  MessageBox.Show("Пожалуйста, войдите в систему для добавления в корзину.", "Авторизация требуется", MessageBoxButton.OK, MessageBoxImage.Warning);  var mainViewModel = Application.Current.MainWindow?.DataContext as MainViewModel;  mainViewModel.CurrentPage = new LoginPage();  return;  }  var cart = \_cartRepository.GetByUserId(SessionManager.CurrentUser.Id);  if (cart == null)  {  cart = new Cart  {  UserId = SessionManager.CurrentUser.Id,  User = SessionManager.CurrentUser,  Items = new List<CartItem>()  };  \_cartRepository.Add(cart);  }  var existingItem = cart.Items.FirstOrDefault(i =>  (item is Product && i.ProductId == item.Id) ||  (item is Service && i.ServiceId == item.Id));  if (existingItem != null)  {  existingItem.Quantity++;  }  else  {  CartItem cartItem;  if (item is Product)  {  cartItem = new CartItem  {  CartId = cart.Id,  ProductId = item.Id ,    Product = item is Product ? (Product)item : null,  Service = item is Service ? (Service)item : null,  Quantity = 1  };  }  else  {  cartItem = new CartItem  {  CartId = cart.Id,    ServiceId = item.Id,  Product = item is Product ? (Product)item : null,  Service = item is Service ? (Service)item : null,  Quantity = 1  };  }  \_cartRepository.AddItemToCart(cart.Id, cartItem);  }  \_cartRepository.Update(cart);  MessageBox.Show($"Товар '{item.Name}' добавлен в корзину!", "Успех", MessageBoxButton.OK, MessageBoxImage.Information);  }  }  catch (Exception ex)  {  MessageBox.Show($"Ошибка при добавлении в корзину: {ex.Message}", "Error", MessageBoxButton.OK, MessageBoxImage.Error);  }  } |

Листинг метода PlaceOrder

|  |
| --- |
| private void PlaceOrder()  {  try  {  if (SessionManager.CurrentUser == null)  {  MessageBox.Show("Пожалуйста, войдите в систему для оформления заказа.", "Авторизация требуется", MessageBoxButton.OK, MessageBoxImage.Warning);  return;  }  if (!CartItems.Any())  {  MessageBox.Show("Корзина пуста.", "Ошибка", MessageBoxButton.OK, MessageBoxImage.Warning);  return;  }  if (string.IsNullOrWhiteSpace(RecipientName) ||  string.IsNullOrWhiteSpace(DeliveryAddress) ||  string.IsNullOrWhiteSpace(PhoneNumber))  {  MessageBox.Show("Заполните все поля доставки.", "Ошибка", MessageBoxButton.OK, MessageBoxImage.Warning);  return;  }  if (string.IsNullOrWhiteSpace(SelectedPaymentMethod))  {  MessageBox.Show("Выберите способ оплаты.", "Ошибка", MessageBoxButton.OK, MessageBoxImage.Warning);  return;  }  var order = new Order  {  UserId = SessionManager.CurrentUser.Id,  User = SessionManager.CurrentUser,  Status = OrderStatus.Pending,  CreatedAt = DateTime.Now,  Items = new List<OrderItem>()  };  foreach (var cartItem in CartItems)  {  var orderItem = new OrderItem  {  ProductId = cartItem.ProductId,  ServiceId = cartItem.ServiceId,  Product = cartItem.Product,  Service = cartItem.Service,  Quantity = cartItem.Quantity  };  order.Items.Add(orderItem);  }  \_orderRepository.Add(order);  // Очистка корзины  \_cart.Items.Clear();  \_cartRepository.Update(\_cart);  CartItems.Clear();  MessageBox.Show("Заказ успешно оформлен!", "Успех", MessageBoxButton.OK, MessageBoxImage.Information);  }  catch (Exception ex)  {  MessageBox.Show($"Ошибка оформления заказа: {ex.Message}", "Error", MessageBoxButton.OK, MessageBoxImage.Error);  }  } |