МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

Учреждения образования «БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ

ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»

Факультет информационных технологий

Кафедра программной инженерии

Специальность 1-40 01 01 Программное обеспечение информационных технологий

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

**ПОЯСНИТЕЛЬНАЯ ЗАПИСКА**

**КУРСОВОГО ПРОЕКТ:**

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

Тема «Приложение магазина обуви»

Исполнитель

студент (ка) 2 курса группы 6 Ляшонок Матвей Михайлович

(Ф.И.О.)

Руководитель работы ст. препод. Сухорукова И. Г.

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

Курсовой проект защищен с оценкой

Председатель Мущук А. Н.

(подпись)

Минск 2024

Содержание

[Введение 3](#_Toc167181943)

[1. Анализ прототипов, литературных источников и формирование требований к проектируемому программному средству 4](#_Toc167181944)

[1.2 Постановка задачи 6](#_Toc167181945)

[1.3 Выводы к разделу 7](#_Toc167181946)

[2. Анализ требований к программному средству и разработка функциональных требований 8](#_Toc167181947)

[2.1 Спецификация функциональных требований к программному средству 8](#_Toc167181948)

[2.2 Выводы к разделу 9](#_Toc167181949)

[3. Проектирование программного средства 10](#_Toc167181950)

[3.1 Архитектура системы 10](#_Toc167181951)

[3.2 Взаимоотношения между классами 11](#_Toc167181952)

[3.3 Модель базы данных 11](#_Toc167181953)

[3.4 Проектирование последовательности взаимодействия 13](#_Toc167181954)

[3.5 Выводы к разделу 14](#_Toc167181955)

[4. Реализация программного средства 15](#_Toc167181956)

[4.1 Основные классы 15](#_Toc167181957)

[4.2 Реализация MVVM и других паттернов 16](#_Toc167181958)

[4.3 Выводы к разделу 17](#_Toc167181959)

[5. Тестирование, проверка работоспособности и анализ полученных результатов 18](#_Toc167181960)

[5.1 Тестирование регистрации и авторизации 18](#_Toc167181961)

[5.2 Тестирование панели пользователя 19](#_Toc167181962)

[5.3 Тестирование панели администратора 19](#_Toc167181963)

[5.4 Выводы к разделу 21](#_Toc167181964)

[6. Руководство по установке и использованию 22](#_Toc167181965)

[6.1 Выводы к разделу 28](#_Toc167181966)

[Заключение 29](#_Toc167181967)

[Список литературы 30](#_Toc167181968)

[Приложение А 31](#_Toc167181969)

[Приложение Б 32](#_Toc167181970)

[Приложение В 33](#_Toc167181971)

# Введение

В современном мире покyпка обуви стала необходимостью для каждого человека. С развитием технологий и ростом потребительского спроса, рынок обувных магазинов также претерпел значительные изменения. Многие традиционные розничные магазины перешли на онлайн-платформы, предлагая клиентам более удобный и простой способ покупки обуви. Однако, несмотря на эти достижения, потребители по-прежнему сталкиваются с проблемами при поиске идеальной пары обуви, соответствующей их предпочтениям и бюджету.

Данный курсовой проект посвящен разработке приложения для магазина обуви, которое позволит упростить процесс покупки и обеспечит клиентам более удобный и персонализированный опыт.

Цель курсового проектирования заключается в создании приложения, которое будет предлагать пользователям ассортимент обуви различных брендов и размеров, а также предоставит возможность персонализации и удобного поиска необходимой модели.

В основу проектирования приложения положен следующий функционал:

* блокирование аккаунта клиента и удаление;
* получение уведомления при появлении товара выбранного размера из списка желаемых;
* возможность добавить товар в список желаемых и корзину, а также заказать;
* возможность создать аккаунт;
* поиск товара по фильтрам.

Пояснительная записка курсового проекта включает следующие разделы:

* + анализ прототипов, литературных источников, формирование требований к проектируемому программному средству;
  + анализ требований к программному средству и разработка функциональных требований;
  + проектирование программного средства;
  + реализация программного средства;
  + тестирование, проверка работоспособности и анализ полученных результатов;
  + руководство по установке и использованию.

Разработка будет осуществляться на языке программирования C# с использованием принципов и методов объектно-ориентированного программирования (ООП). Для хранения данных будет задействована база данных MS SQL, а для создания пользовательского интерфейса будет применена технология Windows Presentation Foundation (WPF).

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

На данный момент большинство приложения для магазина обуви направлены на администрирование магазина, а не на использование пользователями. Зачастую приложения создаются крупными маркетплейсами, продажа на которых не сфокусирована на одной категории товаров, а обхватывает большое количество областей. Также доступны они на мобильных устройствах, а доступ с ПК к ним осуществляется с помощью веб-сайта. Из этого следует, что анализ будет происходить на основе веб-сайтов, которые направлены на продажу обуви.

Как один из примеров рассмотрим сайт superstep.ru. Этот интернет-ресурс предлагает не только обувь, но и одежду, аксессуары, но упор сделан в сторону обуви. Данный веб-сайт обладает удобным и интуитивно понятным интерфейсом. Этот магазин позволяет заказать товар без регистрации. В таком случае необходимо указать номер телефона для оформления заказа. Для регистрации также используется номер телефона, но ещё предоставляется возможность авторизации при помощи социальных сетей либо электронной почты. Для посика предлагается большое количество пунктов для фильтрации, такие как:

* по полу;
* цене;
* категории;
* бренду;
* размеру;
* цвету.
* и т.д.

Для демонстрации веб-сайта superstep.ru выбрана главная страница, представленная на рисунке 1.1.

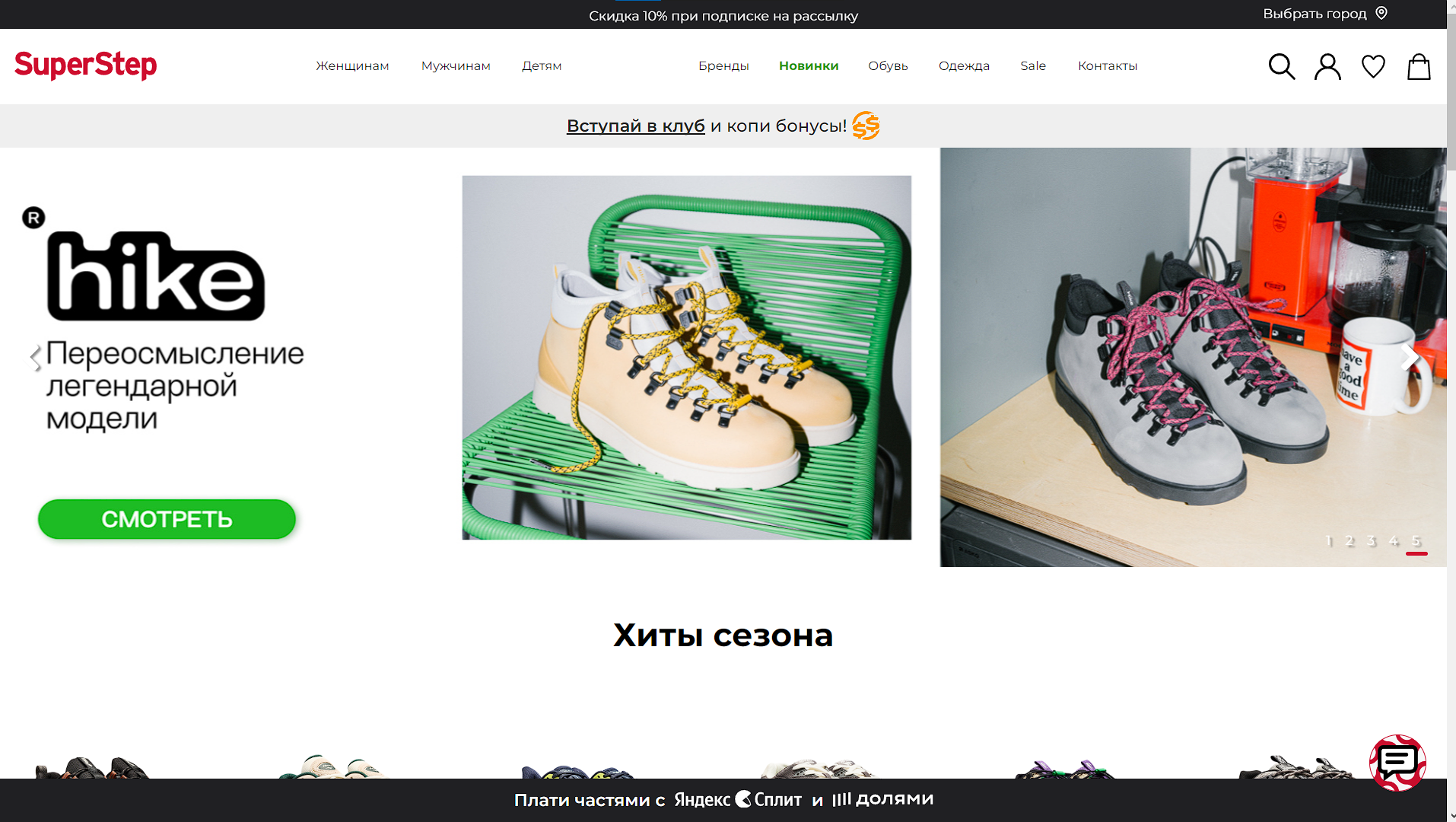


Рисунок 1.1 – Главная страница superstep.ru

В качестве недостатка можно отметить недостаточную оптимизацию в загрузке компанентов сайта, а также отсутствие отзывов о работе веб-сайта и товара.

В качестве другого примера будет приведён сайт opinions.by. Сайт имеет более сдержанный дизайн по сравнению с предыдущим рассмотренным аналогом. Также он имеет не так много опций для фильтрации товара вместе с небольшим количеством категорий товара. Ко все этому стоит упомянуть процесс заказа товара и регистрацию. Перед заказом товара необходимо добавить его в корзину. Далее заполнить форму личными данными. Для регистрации личного кабинета используется электронная почта. Главная страница представлена на рисунке 1.2.

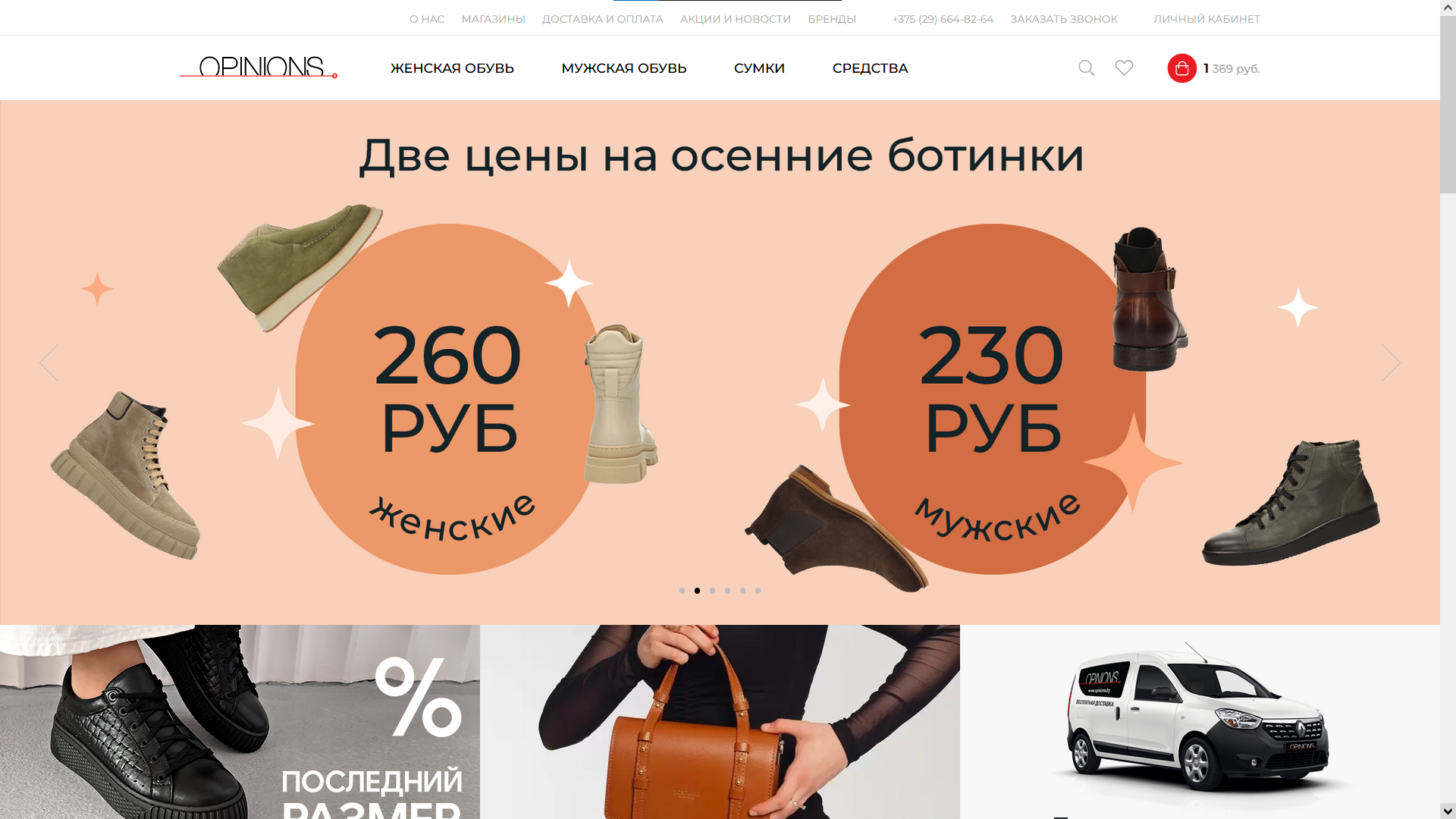


Рисунок 1.2 – Главная страница opinions.by

К преимуществам данного веб-сайта можно отнести быструю работу.

Далее на рассмотрение попадает веб-сайт marko.by. Дизайн данного интерне-ресурса в меру минималистичен, имеет большое количество плавных анимаций интерфейса. Интерфейс отзывчив, прост в освоении, ведь основной переход между категориями товара происходит засчёт навигационного меню, имеющего внушительное количество пунктов. Для регистрации на веб-сайте необходимо заполнить форму, введя личные данные, номер телефона, почту, пароль. После регистрации предоставляется возможность оставлять отзывы по товару, бронировать до 3-х товаров в сутки в фирменной сети магазинов «Марко», видеть статусы и истории всех броней. Бронирование происходит на один из предлагаемых физических магазинов, список которых появляется при выборе определённой пары. Главная страница веб-сайта marko.by представлена на рисунке 1.3.

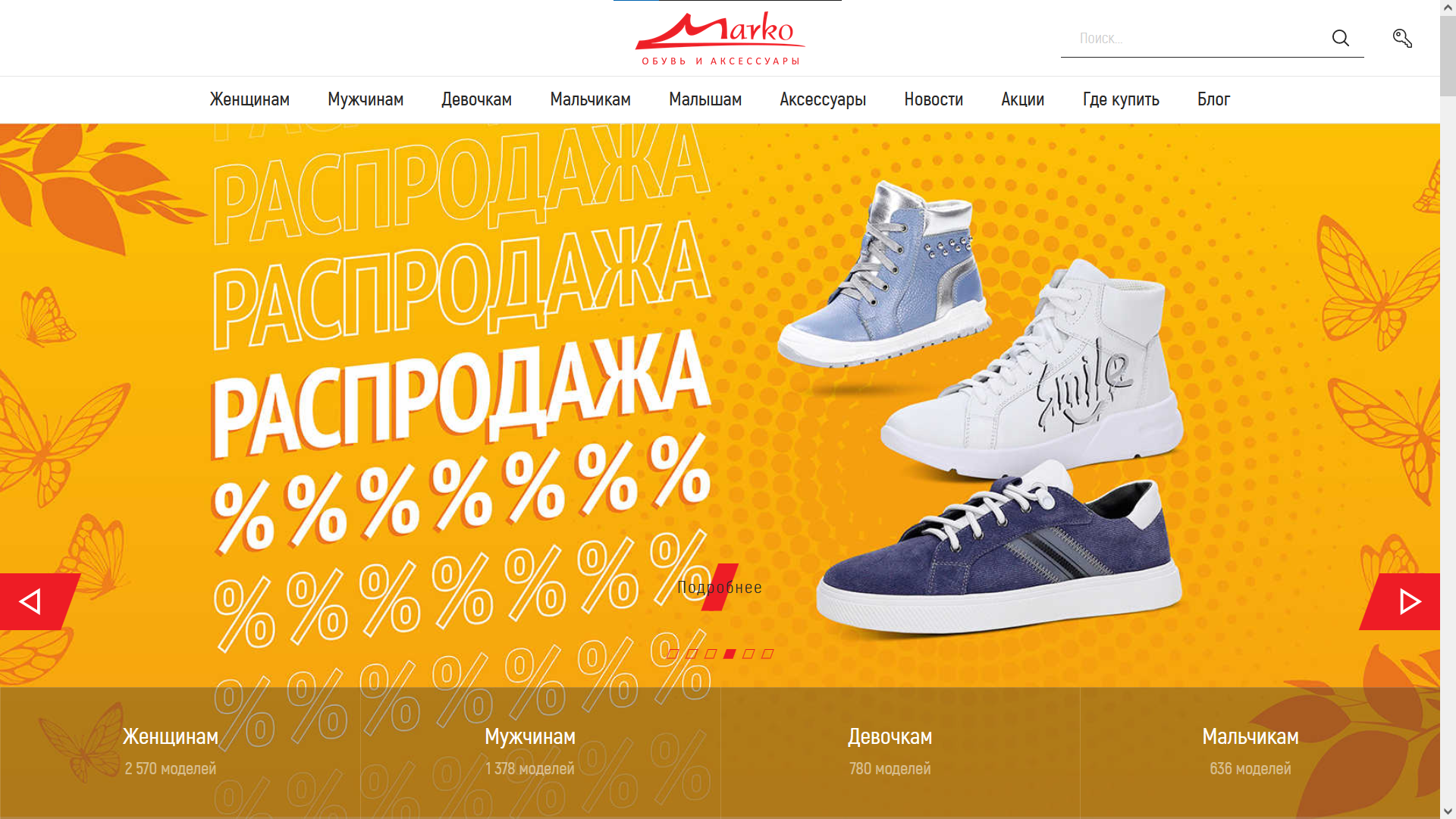


Рисунок 1.3 – Главная страница веб-сайта marko.by

На основании анализа веб-ресурсов можно сделать вывод о необходимых требованиях, таких как:

* удобный, отзывчивый интерфейс;
* наличие строки поиска;
* предоставления широкого выбора опций для фильтрации товара;
* простота заполнения форм регистрации/заказа;
* возможность оставить отзыв;
* оптимизация приложения.

(функциональные и нефункциональные требования дописать).

1.2 Постановка задачи

После анализа прототипов и аналогичных решений, выделение основных задач, которые должно решать программное средство "Магазин обуви", является ключевым шагом для определения функционала и требований к разрабатываемому продукту. Вот несколько основных задач, которые можно выделить на основе анализа:

Основные задачи:

1. Реализовать возможность поиска продуктов с помощью различных фильтров: типу обуви, полу для удобства пользователей и повышения эффективности поиска необходимых товаров.

2. Создать каталог обуви, который позволит пользователям просматривать товары с подробным описанием и изображением, а также добавлять интересующие их продукты в список желаемых и корзину для удобного доступа в будущем.

3. Разработать функционал для оформления заказов, обеспечивающий удобство и быстроту процесса покупки для клиентов магазина обуви.

4. Создать функционал для управления каталогом продуктов, что обеспечит возможность добавления, редактирования и удаления товаров для поддержания актуальности и разнообразия товарного ассортимента.

5. Осуществить тестирование программного средства для магазина продуктов с целью выявления и исправления возможных ошибок и недочетов, обеспечивая высокое качество работы приложения.

При разработке программного средства необходимо учитывать требования и пожелания пользователей, а также использовать современные технологии и методы программирования, чтобы обеспечить удобство и безопасность использования.

1.3 Выводы к разделу

В данном разделе подведены итоговые требования к разработке программного средства на основе веб-аналогов, такие как наличие поисковой строки, категорий для фильтрации, корзины, формы заказа.

2. Анализ требований к программному средству и разработка функциональных требований

Анализ требований – это процесс понимания, документирования, оценки и управления требованиями к системе. Целью анализа требований является создание четкого и понятного описания того, что должна делать система, каким образом и какие требования необходимы для того, чтобы система была полезной и эффективной для пользователей и бизнеса.

Анализ требований включает в себя такие шаги, как сбор и документирование требований, анализ требований на противоречивость, неполноту и загрузку, разработка модели требований, их оценка и проверка, управление требованиями в течение жизненного цикла проекта. Он является критическим и необходимым этапом в разработке любой системы, поскольку он устанавливает основу для дальнейшей работы разработчиков и гарантирует, что система будет соответствовать требованиям пользователей и бизнеса и будет успешно принята ими.

2.1 Спецификация функциональных требований к программному средству

Для выявления функциональных требований были определены следующие роли и группы пользователей:

*1. Пользователь.* Пользователь может просматривать товары, добавлять их в корзину/желаемое, оформлять и оплачивать заказ, фильтровать ассортимент обуви.

*2. Администратор.* Администратор отвечает за управление каталогом (обеспечивает модерацию контента), а также имеет возможность удалить и заблокировать пользователя.

В приложении А представлена диаграмма вариантов использования программного обеспечения разными группами пользователей.

Программное средство должно предоставлять следующие функциональные возможности:

Для пользователя:

регистрация;

авторизация;

просмотр каталога;

сортировка и поиск товаров;

добавление товара в корзину;

добавление товара в желаемое;

заказ товара;

просмотр информации о данных пользователя;

просмотр информации о товарах в корзине;

просмотр информации о товарах в избранном;

просмотр информации о товаре;

Для администратора:

авторизация;

просмотр каталога;

сортировка и поиск товаров;

добавление товаров;

загрузка изображения при добавлении обуви;

редактирование информации о обуви;

удаление товаров;

Диаграмма вариантов использования представлена в Приложении А.

2.2 Выводы к разделу

В данной главе были определены роли и группы пользователей, а также представлена диаграмма использования программного обеспечения разными группами пользователей. В целом, спецификация функциональных требований к программному средству позволяет определить функциональность и ожидания пользователей, а также задает рамки для разработки и тестирования программного продукта.

3. Проектирование программного средства

3.1 Архитектура системы

Программное средство «Магазин обуви» имеет следующую структуру, представленную на рисунке 3.1.

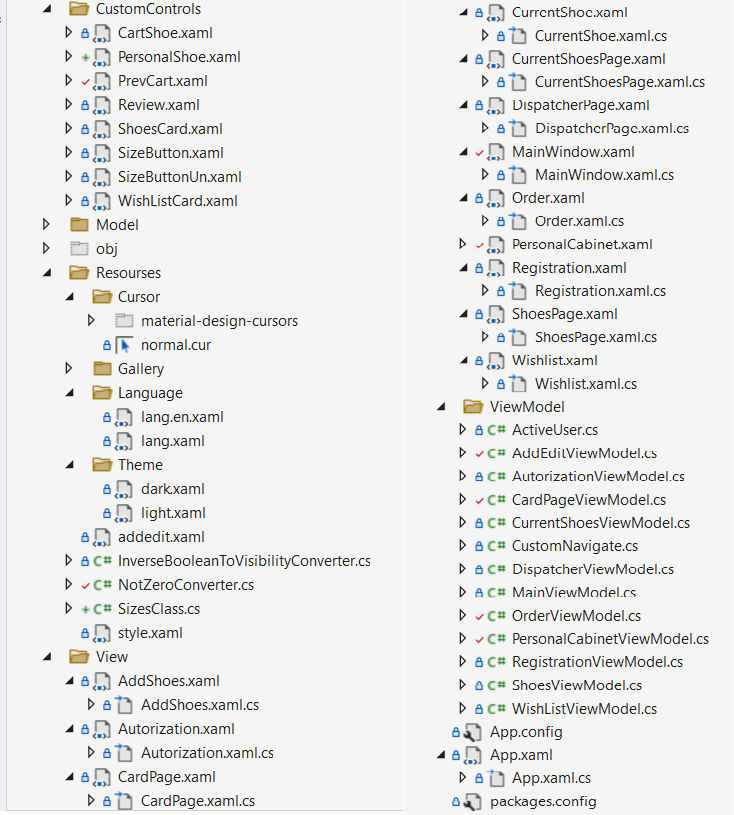


Рисунок 3.1 – Структура проекта

Описание структуры основных папок проекта представлено в таблице 3.1.

Таблица 3.1 – Описание структуры папок и файлов проекта

|  |  |
| --- | --- |
| Имя файла/папки | Содержание |
| CustomControls | Папка, содержащая пользовательские элементы управления |

Продолжение таблицы 3.1

|  |  |
| --- | --- |
| Model | Папка, содержащая сущности базы данных и логику работы с ней |
| Resourses | Дополнительные классы приложения, ресурсы, а также стили |
| View | Содержит внешние представления: окна, страницы |
| ViewModels | Классы, реализующие слой ViewModel паттерна MVVM. |

Таким образом, сформированная таблица помогает понять общую структуру проектируемого программного средства.

3.2 Взаимоотношения между классами

Для представления внутренней структуры программы в виде классов и связей между ними используется диаграмма классов. Приложение имеет многослойную архитектуру, поэтому классы разделяются по группам (слоям). Диаграмма классов модели данных представлена в Приложении Б.

3.3 Модель базы данных

Для реализации поставленной задачи была создана база данных dapperpath. Для ее создания использовалась система управления реляционными базами данных My SQL Server.

На рисунке 3.2 приведена схема одной из основных таблиц: а именно таблицы Users, содержащую информацию о пользователях приложения.

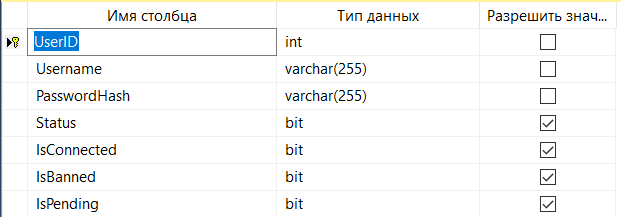


Рисунок 3.2 – Структура таблицы Users

Каждый столбец (поле) таблицы отвечает за хранение определенного типа данных, связанного с пользователем:

"UserID" - уникальный идентификатор пользователя, хранится как целое число (int).

"Username" - хранит текстовую информацию о пользователе, такую как логин.

"UserPasswordHash" - хранят данные о пароле пользователя, соответственно хэш (varchar(255)).

"IsConnected" - хранит информацию об активности пользователя (bit).

"IsBanned" - хранит информацию о доступе пользователя к аккаунту (bit).

"Status" - хранит информацию о статусе пользователя(bit).

"IsPending" - хранит информацию о статусе ожидания желаемого товара пользователя(bit).

База данных состоит из таблиц, представленных на рисунке 3.3.

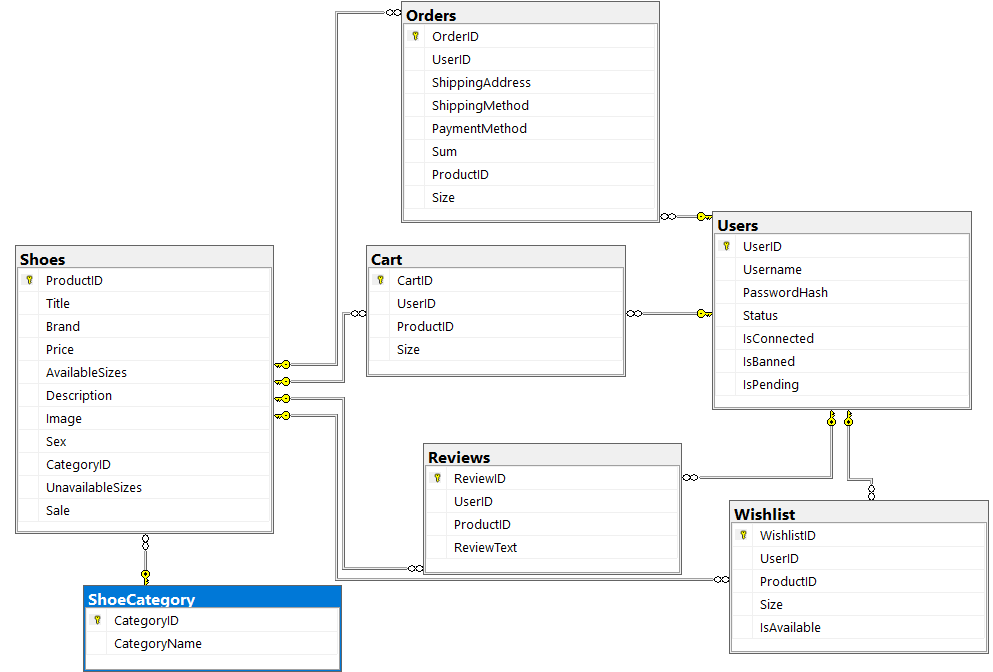


Рисунок 3.3 – Диаграмма модели базы данных dapperpath

На данной диаграмме показаны таблицы и связи между ними в базе данных приложения.

На рисунке 3.4 изображена структура ещё одной немаловажной таблицы Shoes, которая хранит информацию о продуктах.

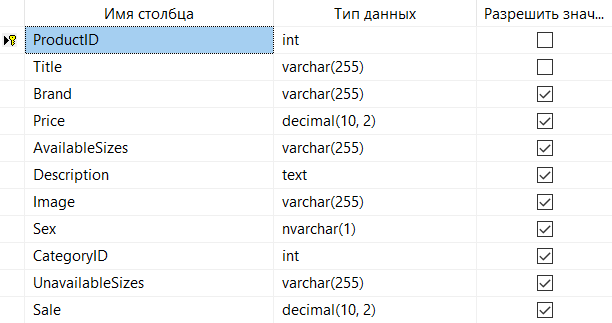


Рисунок 3.4 – Структура таблицы PRODUCTS

В таблице содержатся поля, которые хранят различную информацию о продукте, такую как его название, описание, цена, тип, количество и изображение.

Поля таблицы:

1. ProductId - уникальный идентификатор товара, хранится как целое число.
2. Title - название обуви на английском языке, хранится как varchar(255).
3. Brand - название обуви на русском языке, хранится как varchar(255).
4. Description - описание обуви на русском языке, хранится как text.
5. Price - цена пары обуви, хранится как decimal(10,2).
6. Sale - цена пары обуви с учётом скидки varchar(200).
7. CategoryID - номер типа продукта, хранится как целое число.
8. Image - имя файла или путь к изображению, связанному с продуктом, хранится как varchar(255).
9. AvailableSizes – строка доступных размеров обуви varchar(255).
10. UnavailableSizes – строка недоступных размеров обуви varchar(255).
11. Sex – целевой пол покупателя nvarchar(1)

3.4 Проектирование последовательности взаимодействия

На рисунке 3.5 была спроектирована диаграмма последовательности для авторизации и регистрации. На диаграмме последовательности отображаются только те объекты, которые непосредственно принимают участие во взаимодействии.



Рисунок 3.5 – Диаграмма последовательности   
для авторизации и регистрации

На диаграмме последовательности объекты представлены в виде вертикальных линий, а сообщения между объектами - в виде стрелок, направленных от одного объекта к другому. Каждое сообщение имеет свойство времени жизни, которое указывает, сколько времени занимает передача сообщения между объектами. Диаграмма последовательности позволяет описать не только последовательность сообщений между объектами, но и условия ветвления и циклов в процессе взаимодействия.

3.5 Выводы к разделу

Архитектура системы «Магазин обуви» демонстрирует четко структурированную и логически организованную проектную среду, обеспечивающую эффективное управление данными и взаимодействие между компонентами приложения. Разработка базы данных и проектирование последовательностей взаимодействий обеспечивают надежную и масштабируемую основу для реализации функционала магазина обуви.

4. Реализация программного средства

Для более детального изучения программного средства в данный момент остановимся подробнее на некоторых классах. Весь графический интерфейс определен в XAML-документах.

* 1. Основные классы

Класс AutorizationViewModel реализует методы, необходимые для авторизации пользователей и содержит следующие методы:

* SignInUser(): этот метод обрабатывает процесс входа пользователя в систему. Он проверяет наличие пользователя в базе данных, сравнивает введенный пароль с хешированным паролем, сохраненным в базе данных, и, если успешно, открывает соответствующее окно приложения (либо окно администратора, либо главное окно);
* HashPassword(string): данный метод выполняет хеширование строки пароля для создания ;
* ShowRegister(): данный метод открывает окно для регистрации нового пользователя.

Класс также содержит различные свойства, связанные с пользовательским интерфейсом, такие как Login, Password, AdminUser , которые используются для отображения и обработки данных в окне авторизации.

Класс AddEditViewModel предназначен для обработки полей страницы добавления и изменнения товара

* Back(): вернуться на страницу назад;
* SaveBoots():валидация свойств и сохранение товара в базу данных;
* OpenDialogFile(): открыть файловый диалог для добавления фотографии;

Класс PersonalCabinetViewModel предназначен смены текущего окна на окно личного кабинета:

Свойства Id, Name, Amount, AllOrdersSum, предназначены для отображения информации, а Orders – для отображения списка заказанных товаров.

Все упомянутые выше классы привязаны к модели представления, то есть разделена бизнес-логика и пользовательский интерфейс.

Класс ShoesViewModel предназначен для получения списка пар обуви, который присваивается в Shoes. Так же в данном классе реализованы методы, для фильтрации обуви по полу совместно с значением поисковой строкой. Код продемонстрирован в приложении В. Так же реализована фильтрация по категории обуви.

- FilterBySneakers(): сортирует коллекцию по номеру категории 11;

- FilterByBoots():сортирует коллекцию по номеру категории 12;

- FilterBySport(): сортирует коллекцию по номеру категории 13;

- FilterByPumps():сортирует коллекцию по номеру категории 14.

В выше перечисленных методах происходит уведомление об изменении свойства через паттерн Наблюдатель OnProtertyChanged(), реализация которого представлена в листинге 4.1

|  |
| --- |
| public event PropertyChangedEventHandler PropertyChanged;  protected virtual void OnPropertyChanged([CallerMemberName] string propertyName = null)  {  PropertyChanged?.Invoke(this, new PropertyChangedEventArgs(propertyName));  } |

Листинг 4.1 – Реализация паттерна Наблюдатель

Этот паттерн часто используется в WPF и других .NET-фреймворках пользовательского интерфейса, где он позволяет автоматически обновлять интерфейс пользователя, когда изменяется базовая модель данных. Реализуя INotifyPropertyChanged, класс может уведомлять связанные элементы интерфейса пользователя об изменении свойства, и интерфейс пользователя может соответственно обновить себя.

4.2 Реализация MVVM и других паттернов

Реализация паттерна MVVM основывалась на использовании библиотеки GalaSoft, где реализованы методы RaisePropertyChanged() и RelayCommand().

Так же в данной библиотеки реализован интерфейс ICommand для работы с паттерном Command, который использовался при разработке для привязки логики к кнопкам.

В проекте реализован паттерн Singtone для возможности обновления страницы обуви при изменении отслеживаемой коллекции, который представлен в листинге 4.2.

|  |
| --- |
| private static ShoesViewModel \_instance;  private static readonly object \_lock = new object();  public static ShoesViewModel Instance  {  get  {  if (\_instance == null)  {  lock (\_lock)  {  if (\_instance == null)  {  \_instance = new ShoesViewModel();  }  }  }  return \_instance;  }  } |

Листинг 4.2 Структура свойства Instance

Таким образом в данной реализации использовалось статическое свойство, за счёт которого появились новые возможности.

4.3 Выводы к разделу

В итоге, проектирование и реализация программного средства "Магазин обуви" включила применение различных паттернов проектирования, таких как MVVM, Command и Singtone.

Шаблон MVVM позволил разделить пользовательский интерфейс от бизнес-логики с помощью введения ViewModel, что способствует более гибкой и модульной архитектуре приложения.

Паттерн Command позволяет привязать логику ЭУ для создания широких возможностей.

Singtone предоставляет возможность вызовов методов из другого класса бещ создания экземпляра самого класса, что облегчает разработку проекта.

Такой подход создаёт хорошо структурированное и легко поддерживаемое приложение, где каждая часть системы имеет чётко определённые обязанности и взаимодействует с другими частями через строго определённые интерфейсы.

5. Тестирование, проверка работоспособности и анализ полученных результатов

5.1 Тестирование регистрации и авторизации

При авторизации и регистрации могут возникнуть ситуации, когда пользователь вводит некорректные данные, такие как неверный пароль, незарегистрированный логин. Программное средство обрабатывает такие исключения и уведомляет пользователя о некорректности введённых данных.

На рисунке 5.1 приведен пример обработки неверно заполненных или пустых полей в форме авторизации и авторизации.

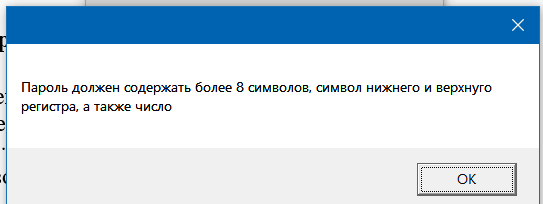


Рисунок 5.1 – Обработка формы авторизации

На рисунках 5.2, 5.3 приведены сценарии обработки ввода некорректных данных при заполнении формы регистрации.

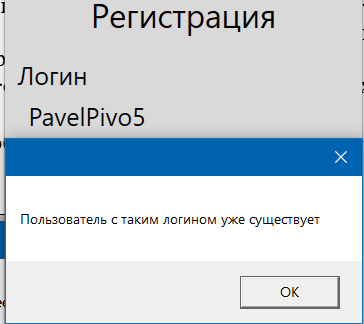


Рисунок 5.2 – Существующий логин

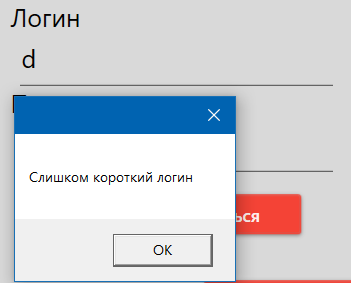


Рисунок 5.3 – Слишком короткий логин

Таким образом, приложение предотвращает поступление некорректных данных в базу данных и препятствует возникновению пользователей с одинаковыми логинами или адресами электронной почты.

5.2 Тестирование панели пользователя

На рисунке 5.6 представлен результат обработки ситуации, когда пользователь пытается оформить пустой заказ.

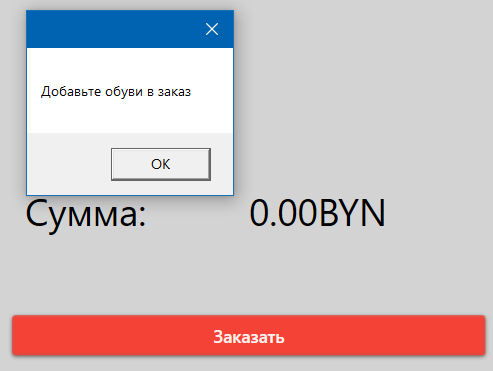


Рисунок 5.6 – Попытка совершить заказ

На рисунке 5.7 представлен результат обработки ситуации, когда пользователь пытается добавить товар с недоступным размером

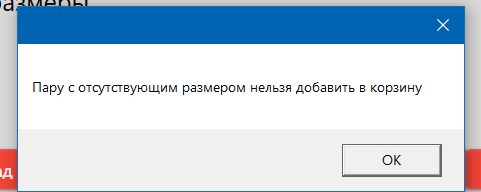


Рисунок 5.7 – Добавление в корзину отсутствующего товара

Таким образом разделена логика между добавлением товара в желаемое и корзину, что подразумевает под собой отличную друг от друга логику.

5.3 Тестирование панели администратора

Примеры обработки пустых полей формы добавления или изменения товара приведен на рисунке 5.9.

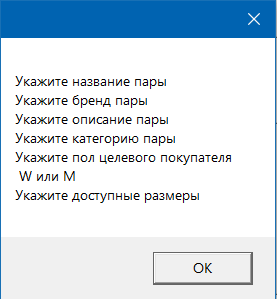


Рисунок 5.8 – Незаполненная форма товара

На рисунке 5.9 приведен сценарий, в котором обрабатывается возможность ввода цены со скидкой большей, чем без скидки.

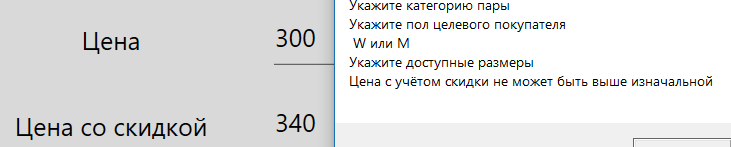


Рисунок 5.9 – Скидка больше Цены

На рисунке 5.10 приведен сценарий, в котором выводиться сообщение о сохранении информации.

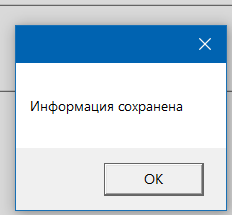


Рисунок 5.10 – Сохранение товара

При успешном удалении товара выводится сообщение, представленное на рисунке 5.11.

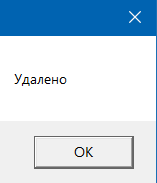


Рисунок 5.11 – Успешное выполнение удаления

Исходя из выше приложенных проверок следует вывод, что приложение покрывает широкий круг валидации информации, а также в понятной форме информирует об успешном окончании некоторого процесса.

5.4 Выводы к разделу

В данном разделе были рассмотрены результаты тестирований работоспособности программного средства. Тестирование основных функций, с которыми будет часто взаимодействовать пользователь, пройдено успешно.

6. Руководство по установке и использованию

При запуске программного средства «Магазин обуви» пользователь сразу попадает на страницу авторизации, которая содержит формы для входа, регистрации. Если у пользователя еще нет аккаунта, он может нажать на кнопку "Нет аккаунта?", чтобы перейти на форму для создания нового аккаунта. В случае ошибки пользователь может закрыть форму. Интерфейс стартового окна представлен на рисунке 6.1.

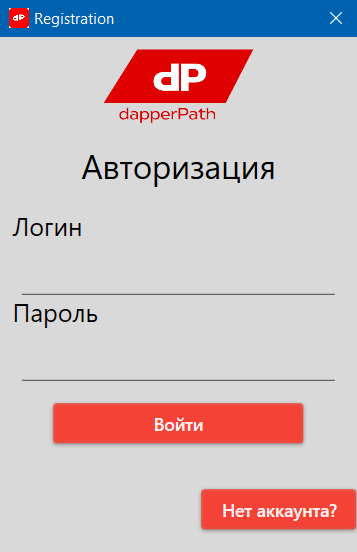


Рисунок 6.1 – Стартовое окно

Форма регистрации представлена на рисунке 6.2.

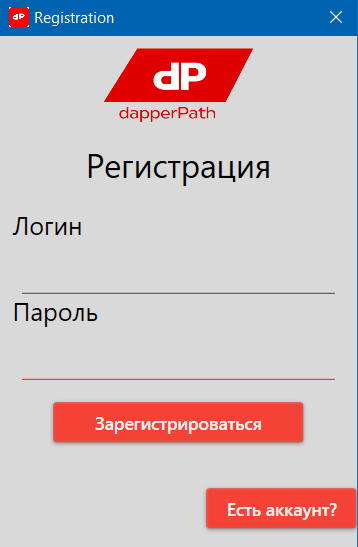


Рисунок 6.2 – Форма регистрации

После ввода необходимых данных на форме регистрации и нажатии на кнопку «Зарегистрироваться» в базу данных добавляется пользователь, а также сразу открывается окно авторизации в программное средство

После успешного входа в систему пользователь попадает на главную страницу. Здесь он может просматривать каталог, сортировать товары, использовать строку поиска для поиска товаров по названию, а также менять цветовую тему и язык приложения.

В зависимости от прав доступа, после входа в приложение пользователь попадает на главную страницу. Если пользователь не обладает правами администратора, то главная страница будет выглядеть так, как представлено на рисунке 6.3.

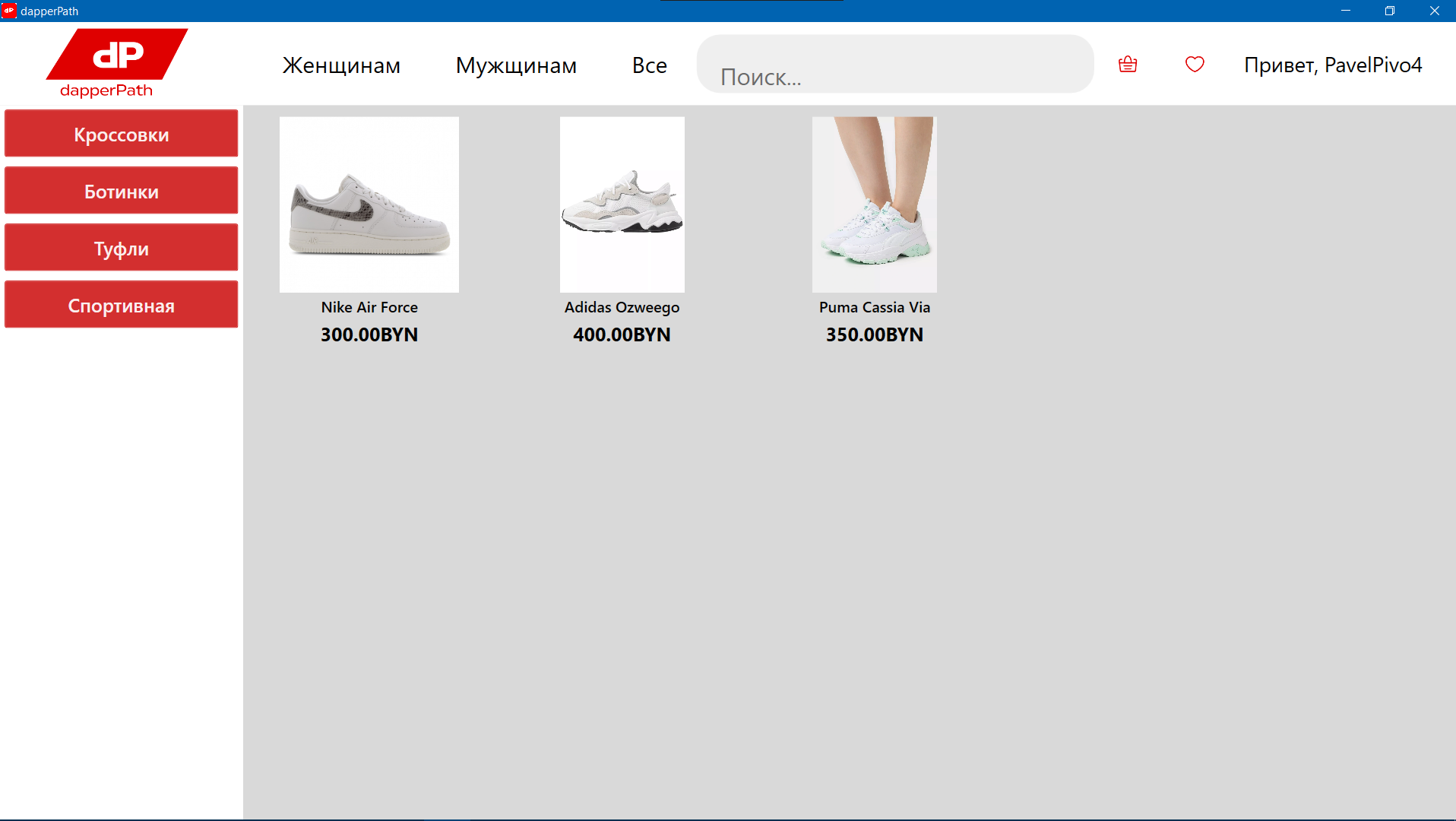


Рисунок 6.3 – Главная страница

На главной странице пользователь может просматривать каталог, сортировать товары согласно фильтрам, указанным слева от товаров. Одновременно пользователь может применить только один фильтр. В строке поиска пользователь может осуществить поиск товаров по их названию.

При нажатии на карточку товара будет произведён переход на страницу конкретного товара, где у пользователя будет возможность оставить отзыв на пару обуви, прочитать отзывы других пользователей. Пример представлен на рисунке 6.4.

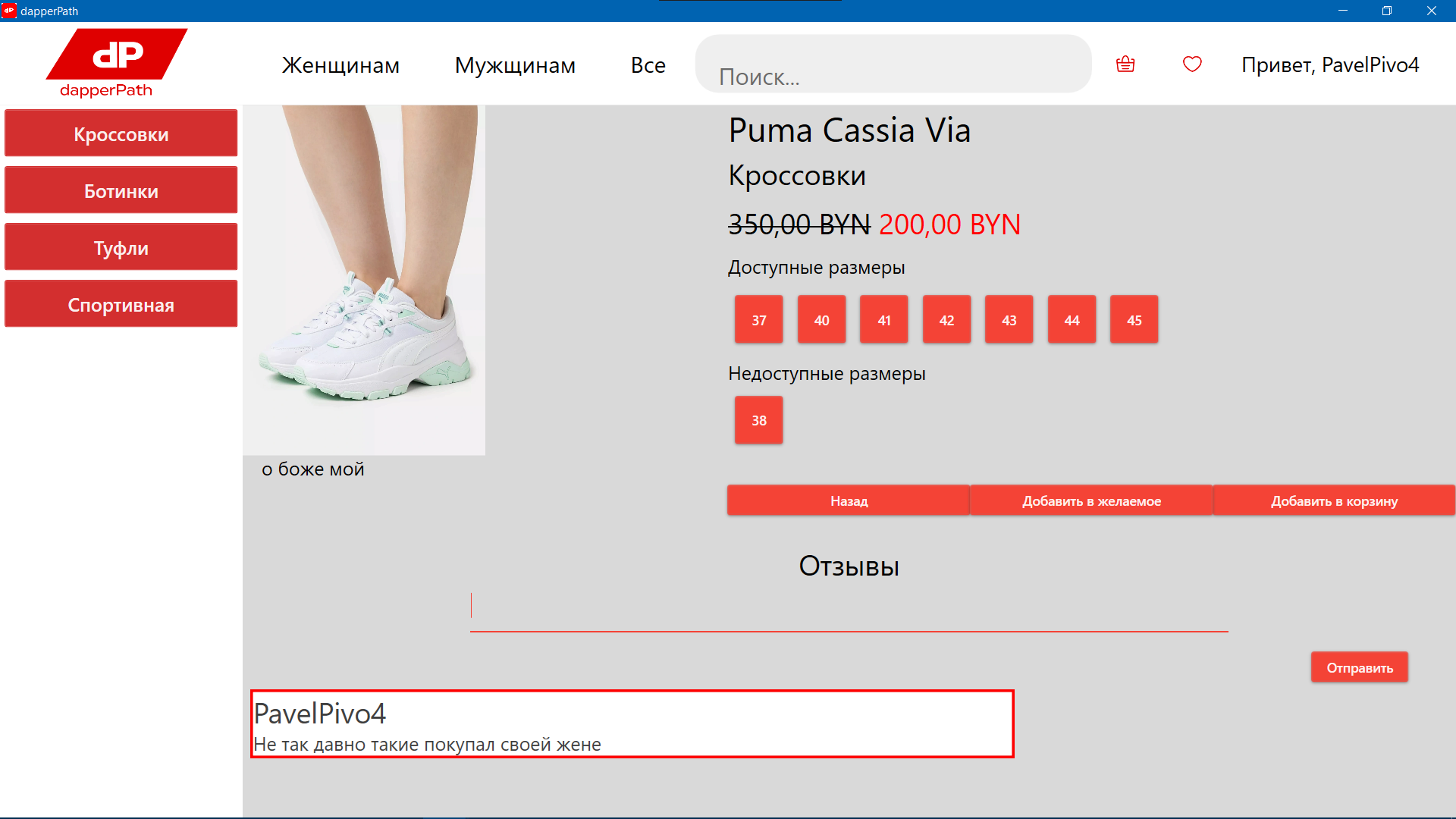


Рисунок 6.4 – Подробное описание товара

Также каждая карточка товара имеет кнопку «Добавить в корзину» или «Добавить в желаемое». При нажатии товары заносятся в соответствующие разделы, нажав перед этим на кнопку с номером размера. Просмотреть товары в корзине можно по нажатию на кнопку корзины в верхней панели. Интерфейс раздела «Корзина» Представлен на рисунке 6.5.

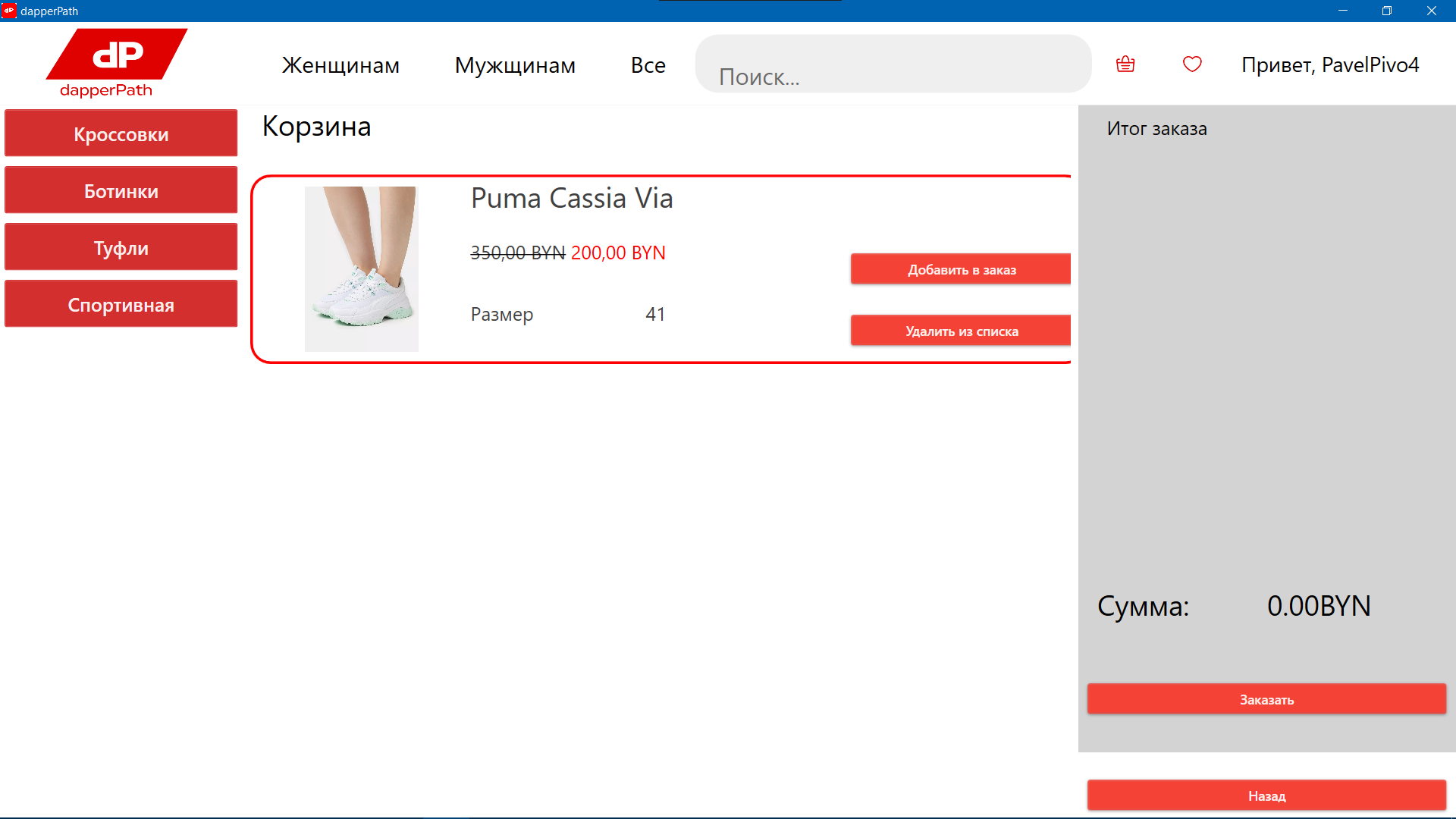


Рисунок 6.5. – Раздел «Корзина»

Чтобы добавить товар из корзины в заказ, необходимо нажать на кнопку «Добавить в заказ». При нажатии на кнопку «Удалить из списка» товар будет удалён из корзины. После добавления товара в заказ необходимо нажать на кнопку «Заказать».

Нажав на кнопку сердца, страница смениться на «Желаемое», предварительно указав размер обуви. В данную вкладку можно добавить обувь как доступного размера, так и недоступного. Так же как и на странице «Корзина», на странице «Желаемое» есть возможность удалить товар из списка, нажав на кнопку «Удалить из списка». Кнопка «Добавить в корзину» доступна только в том случае, если пара есть в наличии. Страница «Желаемое» изображена на рисунке 6.6.

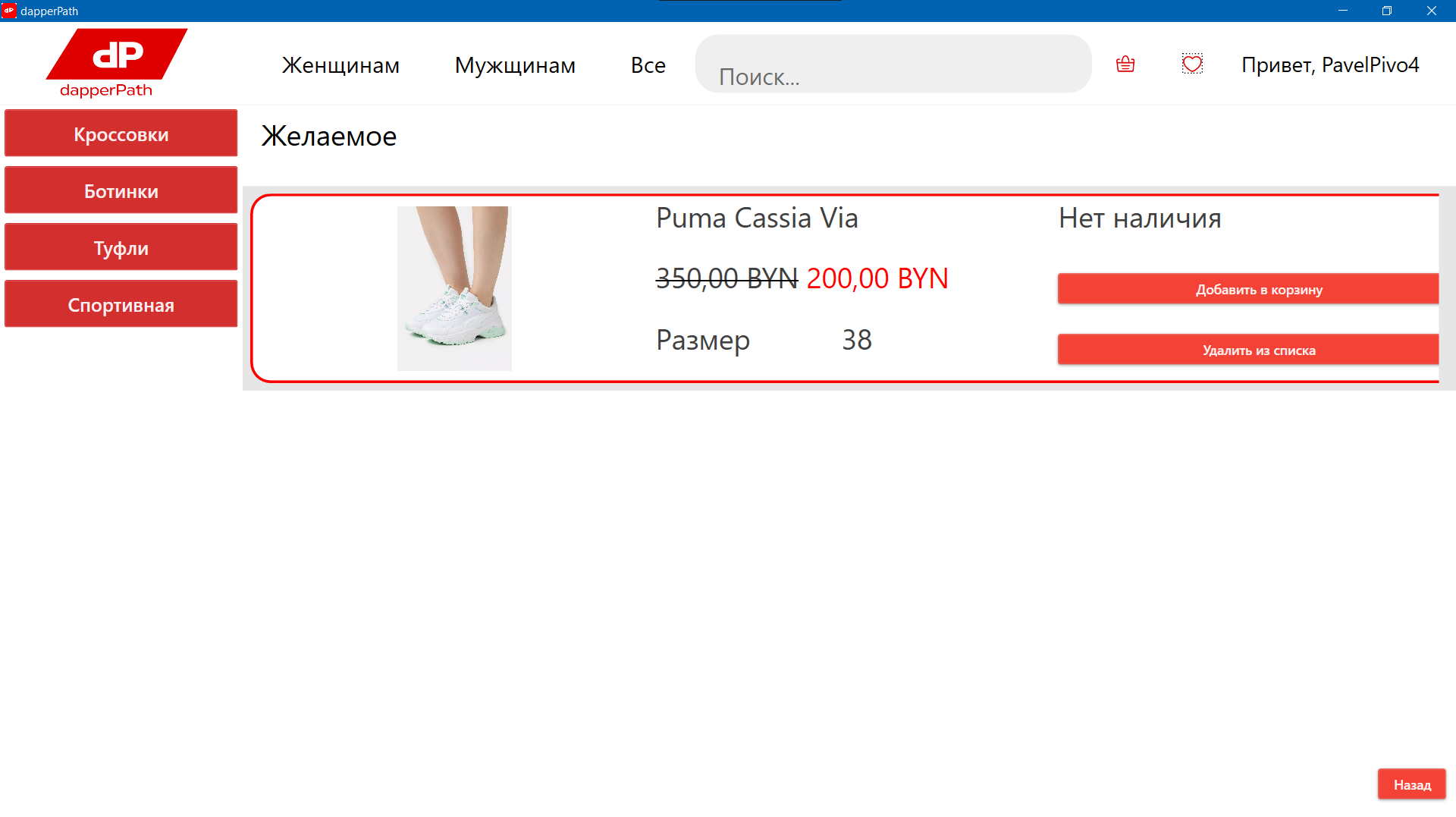


Рисунок 6.6 – Страница «Желаемое»

Также пользователю доступна возможность просмотра данных об аккаунте и о заказах. Для того, чтобы попасть на соответствующую страницу, пользователь должен нажать на кнопку «Привет, (пользователь)». Интерфейс страницы «Аккаунт» представлен на рисунке 6.7.

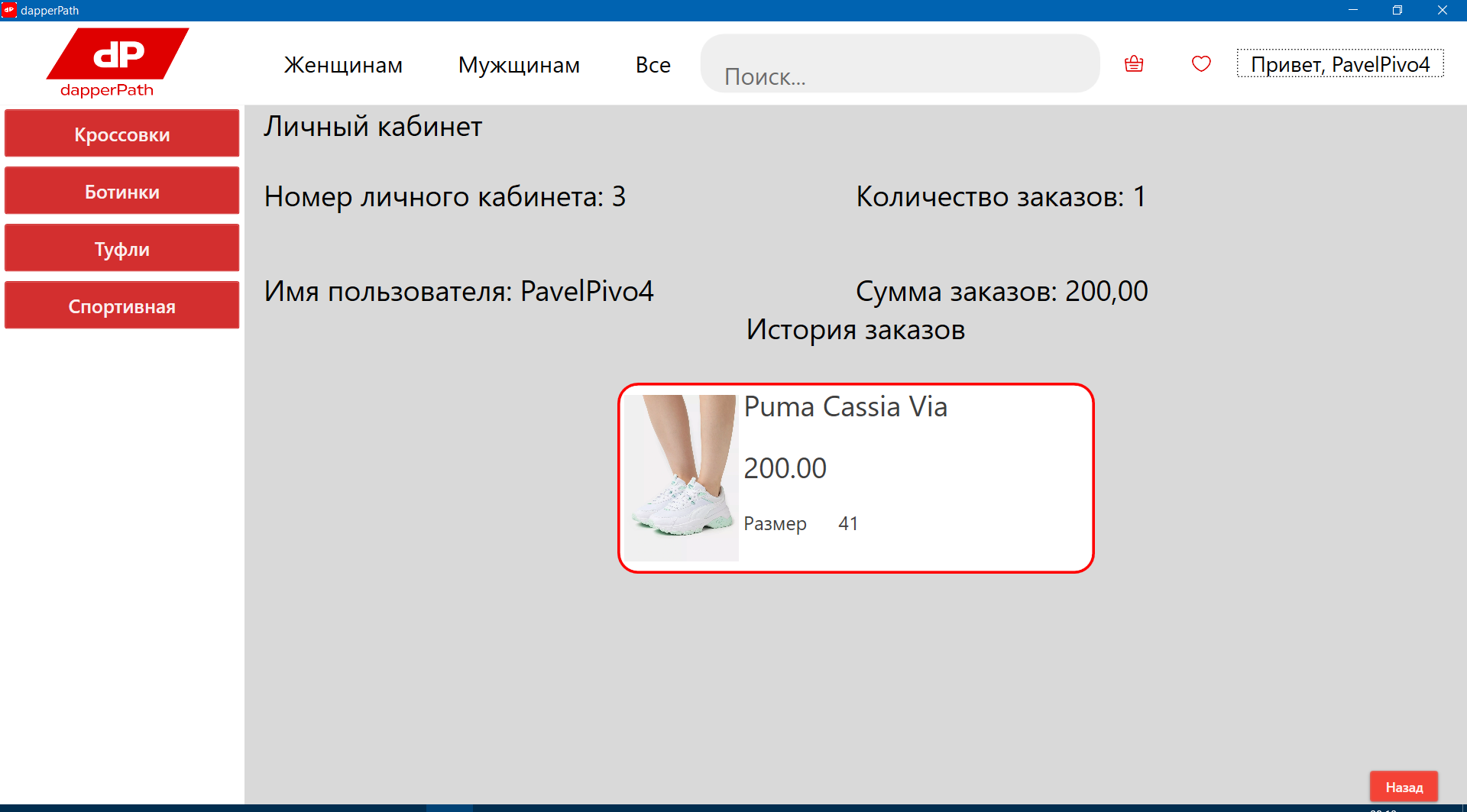


Рисунок 6.7 – Страница «Аккаунт»

В личном кабинете отображается имя пользователя, номер личного кабинета, количество заказов, а также сумма всех заказов.

Если пользователь обладает правами администратора, то главное окно будет выглядеть, как показано на рисунке 6.8.

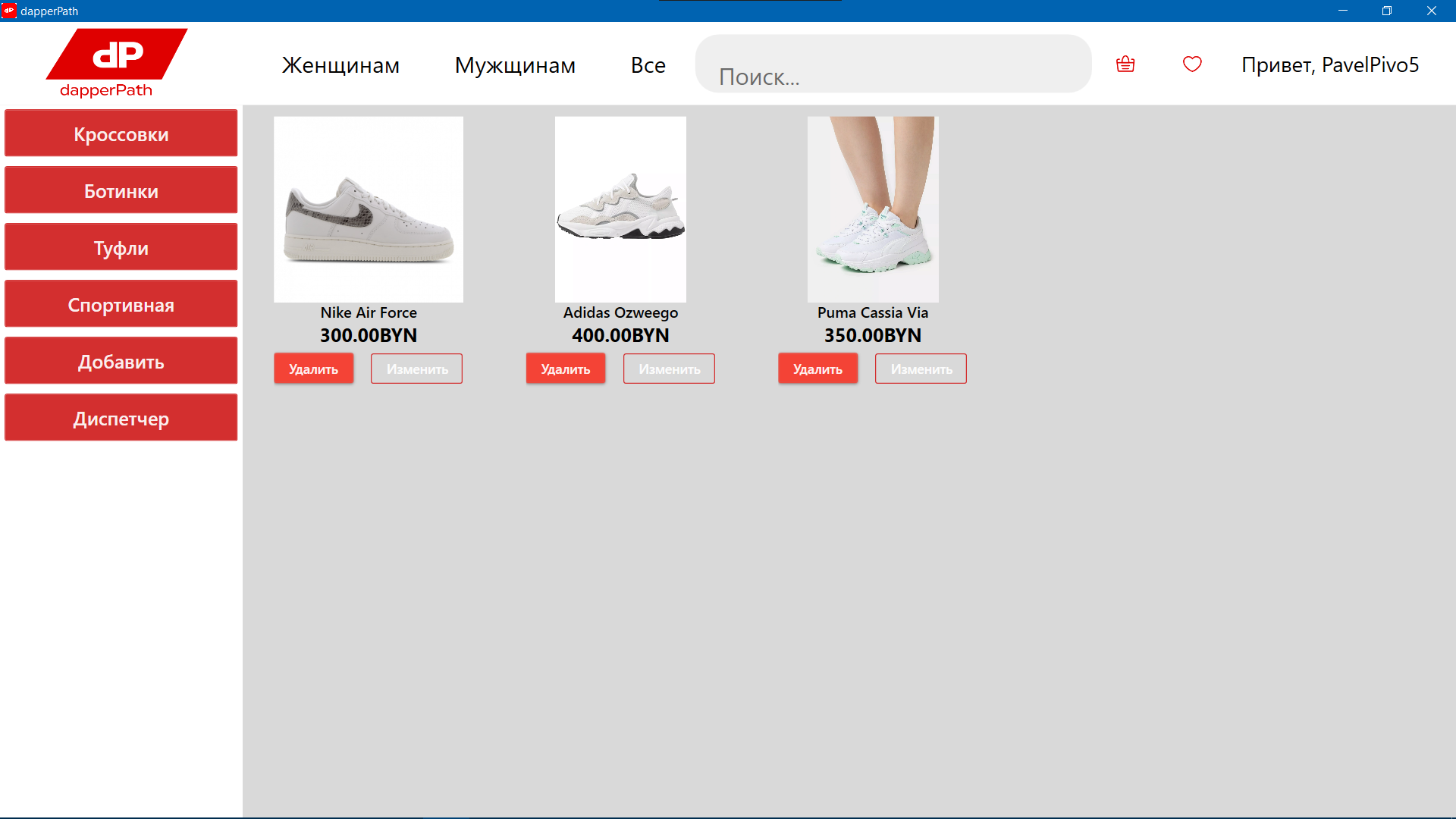


Рисунок 6.7 – Главное окно администратора

Администратор также может просматривать каталог и выполнять сортировку и поиск товаром по заданным критериям. Кроме этого, по нажатию на кнопки «Добавить», «Изменить», «Удалить» администратор может осуществить соответствующие операции.

Для изменения товара необходимо нажать на кнопку «Изменить». Откроется форма, аналогичная форме добавления, только с заполненной информацией, которую можно изменять. Для деактивации необходимо нажать на кнопку «Удалить».

При добавлении товара откроется пустая форма, интерфейс которой представлен на рисунке 6.9.

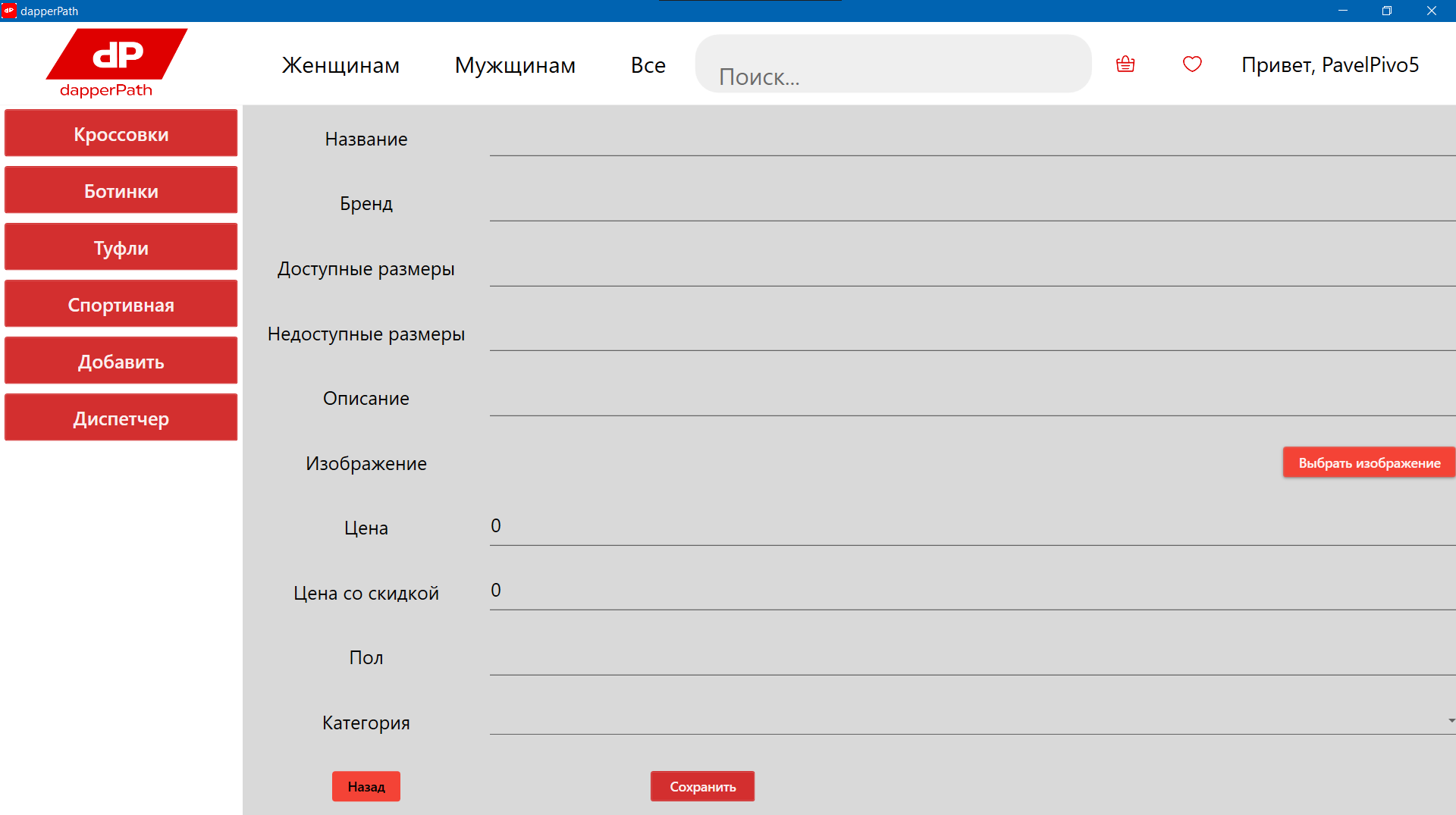


Рисунок 6.9 – Форма добавления товара

Администратор имеет возможность манипулировать аккаунтами пользователей на странице «Диспетчер», а именно удалять и изменять доступ. Если у пользователя будет запрет, то он не сможет воспользоваться программным средством. Страница «Диспетчер» представлена на рисунке 6.10.

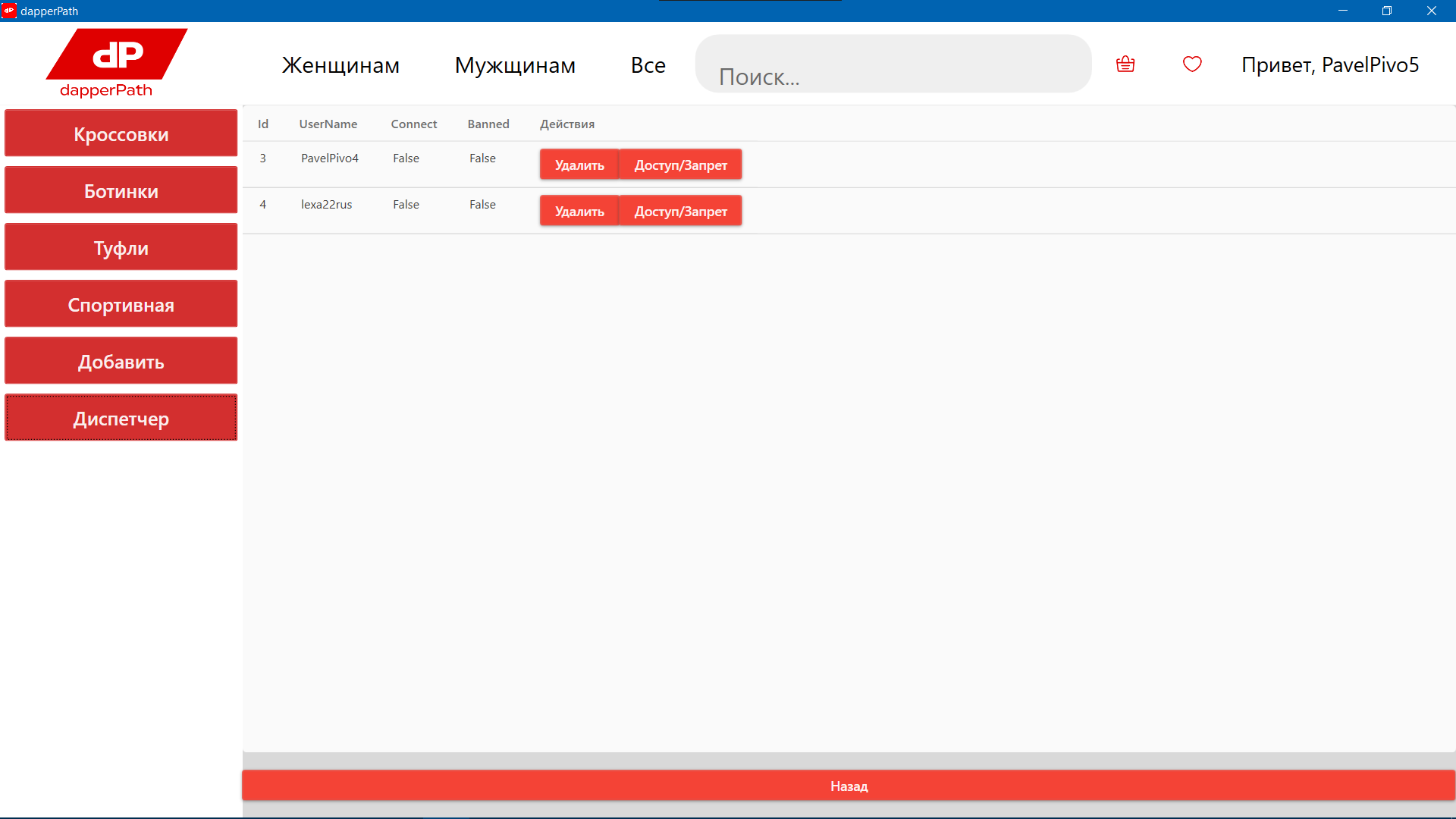


Рисунок 6.10 – Страница «Диспетчер»

6.1 Выводы к разделу

В данном разделе описано использование программного средствами как пользователем, так и администратором. Приложение обладает функционалом, свойственным для магазина обуви.

Заключение

В заключение можно отметить, что разработка курсового проекта "Магазин обуви" с использованием языка C#, WPF, Entity Framework и MaterialDesign продемонстрировала высокую эффективность и удобство современных технологий и паттернов проектирования. Интеграция паттернов Singtone и Command обеспечила надежное и гибкое управление данными, способствуя чистоте и удобству работы при разработке. Паттерн Command позволил организовать обработку команд, улучшив читаемость и поддержку кода, а паттерн MVVM обеспечил четкое разделение логики представления и бизнес-логики.

Использование WPF в сочетании с библиотекой MaterialDesign позволило создать современный и привлекательный пользовательский интерфейс, который улучшает пользовательский опыт и делает взаимодействие с приложением интуитивно понятным. Применение Entity Framework упростило работу с базой данных, обеспечивая безопасность, удобные средства для выполнения запросов и управления данными.

В ходе работы над проектом был также реализован ряд дополнительных функций, таких как система регистрациии авторизации пользователей. Эти функции значительно расширяют возможности приложения, делая его более гибким и универсальным.

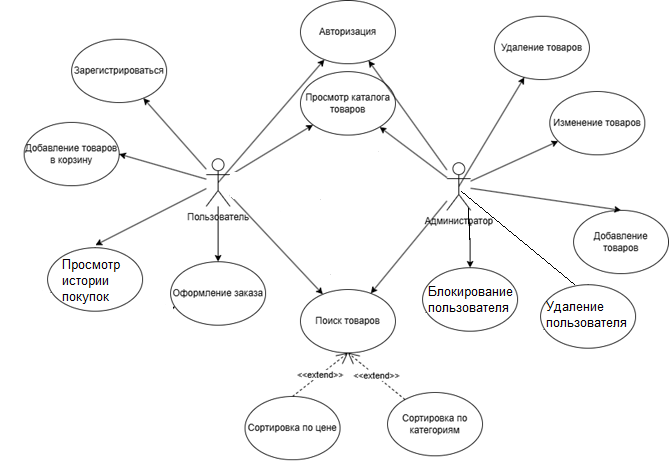
Кроме того, особое внимание было уделено тестированию и отладке кода, что позволило обеспечить высокое качество и стабильность работы приложения. Были применены методы модульного и интеграционного тестирования, что позволило выявить и устранить потенциальные ошибки на ранних этапах разработки.

В результате, курсовой проект "Магазин обуви" представляет собой хорошо структурированное, масштабируемое и легко поддерживаемое приложение, в котором применены передовые практики разработки ПО. Это позволяет не только достигнуть поставленных целей, но и заложить прочную основу для дальнейшего развития и улучшения функциональности приложения. Проект демонстрирует возможности современных технологий и подходов к разработке программного обеспечения, обеспечивая удобство и надежность конечного продукта.

Список литературы

1. Интернет-ресурс «Superstep» [Электронный ресурс] –  [https://superstep.ru/](%20https://superstep.ru/) – Дата доступа: 14.04.2024.
2. Интернет-ресурс «Opinions» [Электронный ресурс] –https://opinions.by/ – Дата доступа: 12.05.2024.
3. Интернет-ресурс «Marko.» [Электронный ресурс] – https://www.marko.by/ – Дата доступа: 15.04.2024.
4. Троелсен, Э. Язык программирования C# 7 и платформы .NET и .NET Core / Э. Троелсен, Ф. Джепикс. – М. : Вильямс, 2018. – 1376 с.
5. Нэйджел, И. C# 6.0 и платформы .NET 4.6: полное руководство / И. Нэйджел, Дж. Глиндс. – СПб. : Питер, 2016. – 1152 с.
6. Рихтер, Д. CLR via C#. Программирование на платформе Microsoft .NET Framework 4.5 на языке C# / Д. Рихтер. – СПб. : Питер, 2014. – 896 с.
7. Макдональд, М. Pro WPF in C# 2010: Windows Presentation Foundation in .NET 4 / М. Макдональд. – Н. : Apress, 2010. – 1200 с

Приложение А



Приложение Б



Приложение В

Код фильтрации товара

|  |
| --- |
| public enum FilterMode  {  All,  Female,  Male  }  public string \_filterText;  private FilterMode \_filterMode = FilterMode.All;  public FilterMode filterMode  {  get { return \_filterMode; }  set  {  \_filterMode = value;  RaisePropertyChanged(nameof(FilterMode));  FilterShoes(\_filterText);  }  }  public void FilterByFemale()  {  Shoes = new ObservableCollection<Shoes>(\_totalShoes.Where(shoe => shoe.Sex == "W"));  }  public void FilterByMale()  {  Shoes = new ObservableCollection<Shoes>(\_totalShoes.Where(shoe => shoe.Sex == "M"));  }  public void ShowAll()  {  Shoes = new ObservableCollection<Shoes>(currentStuff);  } public void FilterShoes(string filterText)  {  \_filterText = filterText;  switch (filterMode)  {  case FilterMode.All:  if (\_filterText == null)  {  ShowAll();  }  else  {  Shoes = new ObservableCollection<Shoes>(currentStuff.Where(s => s.Title.ToLower().Contains(filterText.ToLower())).ToList());  }  break;  case FilterMode.Female:  if (\_filterText == null)  {  FilterByFemale();  }  else  {  Shoes = new ObservableCollection<Shoes>(currentStuff.Where(s => s.Sex == "W" && s.Title.ToLower().Contains(filterText.ToLower())).ToList());  }  break;  case FilterMode.Male:  if (\_filterText == null)  {  FilterByMale();  }  else  {  Shoes = new ObservableCollection<Shoes>(currentStuff.Where(s => s.Sex == "M" && s.Title.ToLower().Contains(filterText.ToLower())).ToList());  }    break;  }  OnPropertyChanged(nameof(Shoes));  } |