Министерство образования и науки Республики Башкортостан

Государственное автономное профессиональное образовательное учреждение

Уфимский колледж статистики, информатики и вычислительной техники

Тема проекта «»

Практическая работа №3 «Разработка тестовых сценариев для аутентификации»

МДК.01.02 Поддержка и тестирование программных модулей

|  |  |
| --- | --- |
|  | Преподаватель  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Л.Р. Набиева  «\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2024 г. |
|  | Студент гр. 22П-3  А. А. Ахмадуллин  «12» ноября 2024 г. |

2024

***Цель работы:***

Разработать функционал в модулях выводящий товары в UserControl из JSON файла. В приложении должны быть реализованны добавление/удаление/редактирование товара, так же должна быть реализована сортировка/поиск/фильтрация товаров.

***Описание работы:***

**1. Вывод товаров в окно:**

При инициализации окна главного меню вызывается функция

updateListView, которая получает товары из функции deserialize класса Offer

создает новые UserCard на основе данных товаров из листа и присваивает эти UserCard в ListView

private void updateListView()

{

List<Offer> catalog = Offer.deserialize();

catalog = chekFilters(catalog);

offers\_ListView.Items.Clear();

foreach (Offer offer in catalog)

{

OfferCard currentOfferCard = new OfferCard();

currentOfferCard.DataContext = offer;

currentOfferCard.deleteEventeChanged += UserControl\_deleteChanged;

currentOfferCard.editingEventeChanged += UserControl\_editChanged;

offers\_ListView.Items.Add(currentOfferCard);

}

if (offers\_ListView.Items.Count != 0)

offersWereRunningOutLabel.Visibility = Visibility.Hidden;

else

offersWereRunningOutLabel.Visibility = Visibility.Visible;

}

public static List<T> Deserialize<T>(string path)

{

if (!File.Exists(path))

{

using (File.Create(path)) { }

}

string stringItemsList = File.ReadAllText(path);

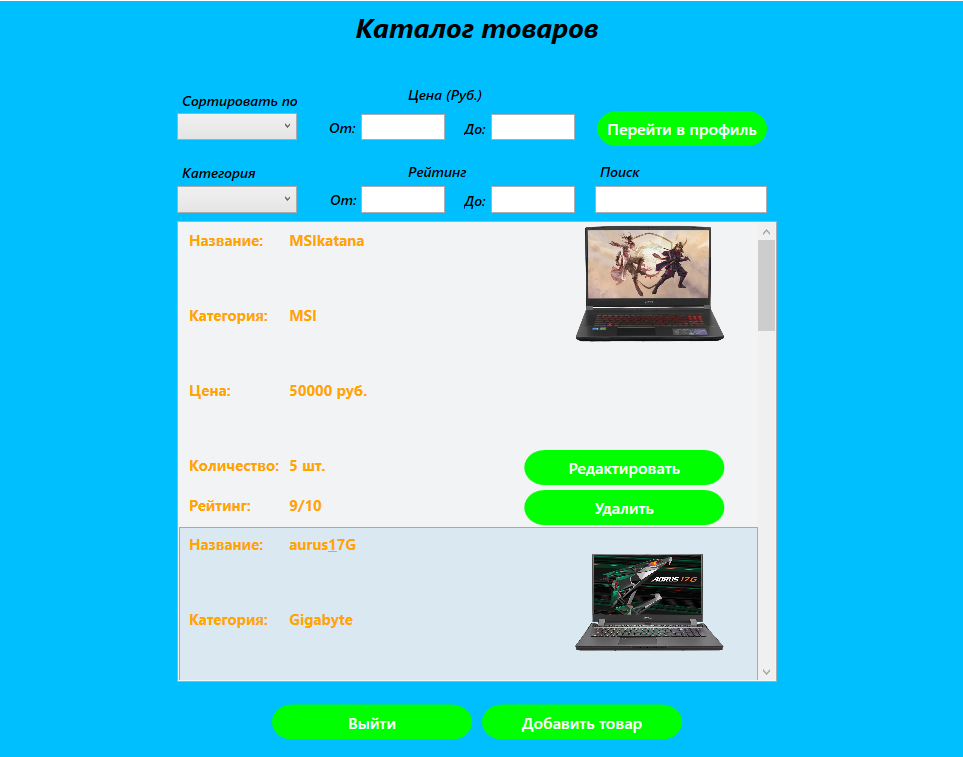
List<T> itemList = JsonConvert.DeserializeObject<List<T>>(stringItemsList);

if (itemList == null)

itemList = new List<T>();

return itemList;

}



**2. Добавление товара:**

При нажатии на кнопку “Добавить товар” вызывается функция addOfferFunction открывающая окно добавления и редактирования товаров:

private void addOfferFunction()

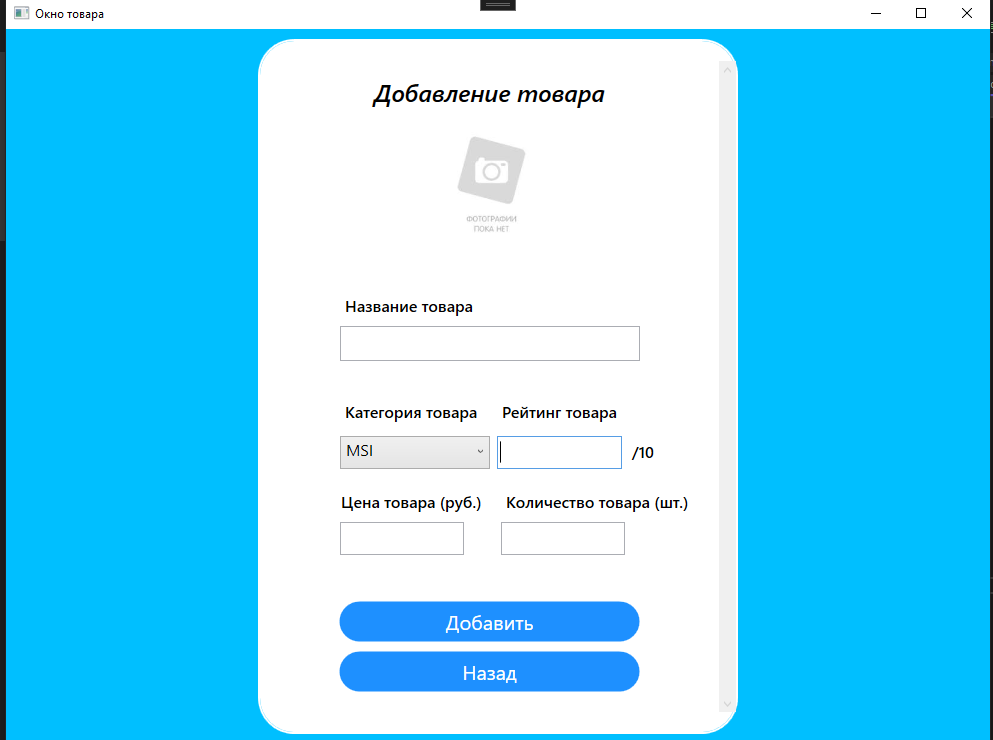
{

AddAndEditiongOfferWindow addAndEditiongOfferWindow = new AddAndEditiongOfferWindow(currentPerson);

addAndEditiongOfferWindow.Show();

this.Close();

}



После того, как пользователь заполнил поля и нажал кнопку вызвается функция save\_Click, которая проверяет данные на пустые поля и некорректные данные, и если все верно то доболяется новый товар:

private void save\_Click(object sender, RoutedEventArgs e)

{

string response = validation();

if (response != "")

{

MessageBox.Show(response);

return;

}

newOffer();

}

private string validation()

{

if (categoryCB.SelectedItem == null)

{

return "Невыбрана категория!";

}

return Offer.validationOfferData(nameTextBox.Text, (categoryCB.SelectedItem as ComboBoxItem).Content.ToString(),countTextBox.Text, priceTextBox.Text, retingTextBox.Text);

}

public static string validationOfferData(string name, string category, string count, string price, string rating)

{

if (name == "" || category == null || category == "" || count == "" || price == "" || rating == "")

{

return "Введены не все данные!";

}

if (Validation.nameRegex.IsMatch(name))

{

return "Некорректное название товара!";

}

int count1;

int price1;

int rating1;

bool suscessCount = int.TryParse(count, out count1);

bool suscessPrice = int.TryParse(price, out price1);

bool suscessRating = int.TryParse(rating, out rating1);

if (!suscessCount)

{

return "Некорректное колличество товаров!";

}

if (count1<1 || count1 > 9999999)

{

return "Колличество товаров должно быть от 1 до 9999999 шт!";

}

if (!suscessPrice)

{

return "Некорректное цена товара!";

}

if (price1 < 1 || price1 > 9999999)

{

return "Цена товара должна быть от 1 до 9999999 руб!";

}

if (!suscessRating)

{

return "Некорректный рейтинг";

}

if (rating1 < 1 || rating1 > 10)

{

return "Рейтинг товара должен быть от 1 до 10!";

}

return "";

}

функция newOffer добовляет довар или увеличивает его колличтество в каталоге, если товар с такими данными уже существует:

private void newOffer()

{

Offer offer = new Offer(Offer.getId(), nameTextBox.Text, (categoryCB.SelectedItem as ComboBoxItem).Content.ToString(),

countTextBox.Text + " шт.", priceTextBox.Text + " руб.", retingTextBox.Text + "/10", imagePath);

if (currentOffer == null)

{

bool hava = false;

Offer haveOffer = Offer.offerExists(nameTextBox.Text, (categoryCB.SelectedItem as ComboBoxItem).Content.ToString(),

retingTextBox.Text + "/10", priceTextBox.Text + " руб.");

if ( haveOffer != null)

{

offer = haveOffer;

offer.Count = (int.Parse(offer.Count.Substring(0, offer.Count.Length-4)) + int.Parse(countTextBox.Text)) +" шт.";

Offer.EditingOffer(offer.Id, offer);

MessageBox.Show("Товар добавлен");

backFunction();

return;

}

Offer.AddOffer(offer);

MessageBox.Show("Товар добавлен");

backFunction();

return;

}

offer.Id = currentOffer.Id;

Offer.EditingOffer(currentOffer.Id, offer);

MessageBox.Show("Товар изменен");

backFunction();

}

public static void AddOffer(Offer newOffer)

{

CastomSerialize.SerializationJson<Offer>(Path, newOffer);

}

public static void SerializationJson<T>(string path, T item)

{

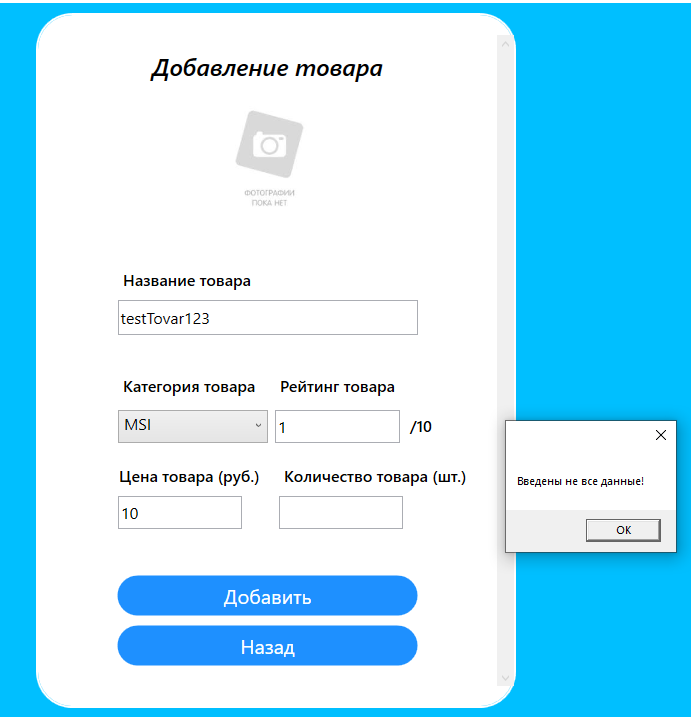
List<T> itemList = Deserialize<T>(path);

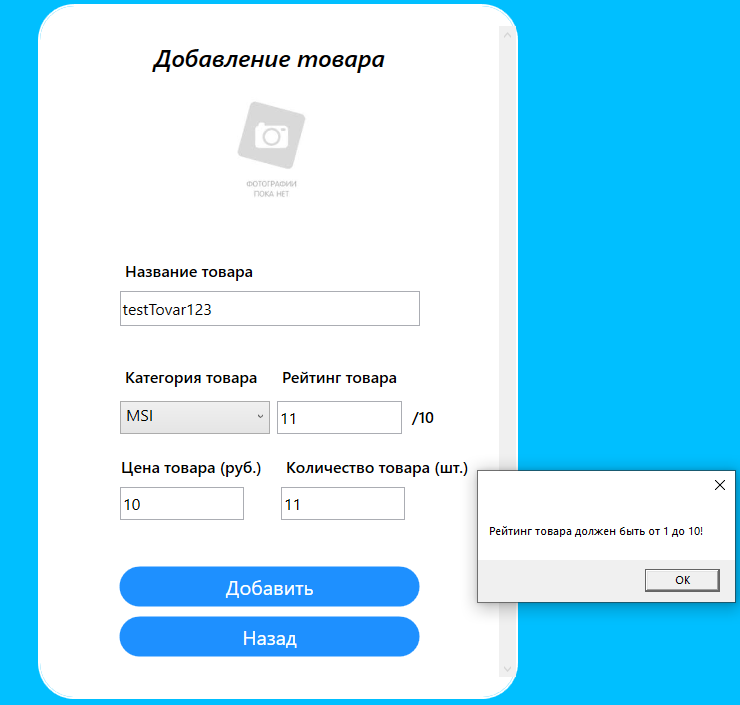
itemList.Add(item);

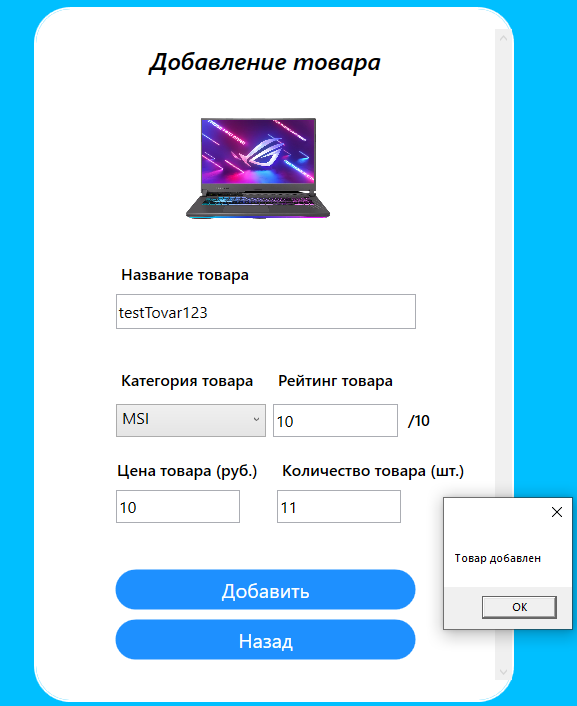
string itemsListToJson = JsonConvert.SerializeObject(itemList, Formatting.Indented);

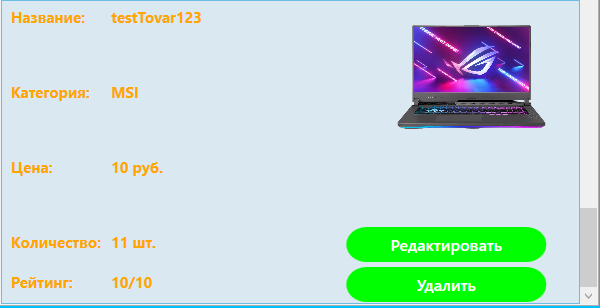
File.WriteAllText(path, itemsListToJson);

}









**3. Удаление товара:**

При нажатии на кнопку удалить вызывается функция UserControl\_deleteChanged, которая отображает диалоговое окно с тесткстом “Вы уверены, что хотите удалить това?”. Если мы нажамаем кнопку “Ok” в диалоговом окне, то товар удаляется:

private void UserControl\_deleteChanged(object sender, EventArgs e)

{

OkOrCancel okOrCancel = new OkOrCancel();

bool? result = okOrCancel.ShowDialog();

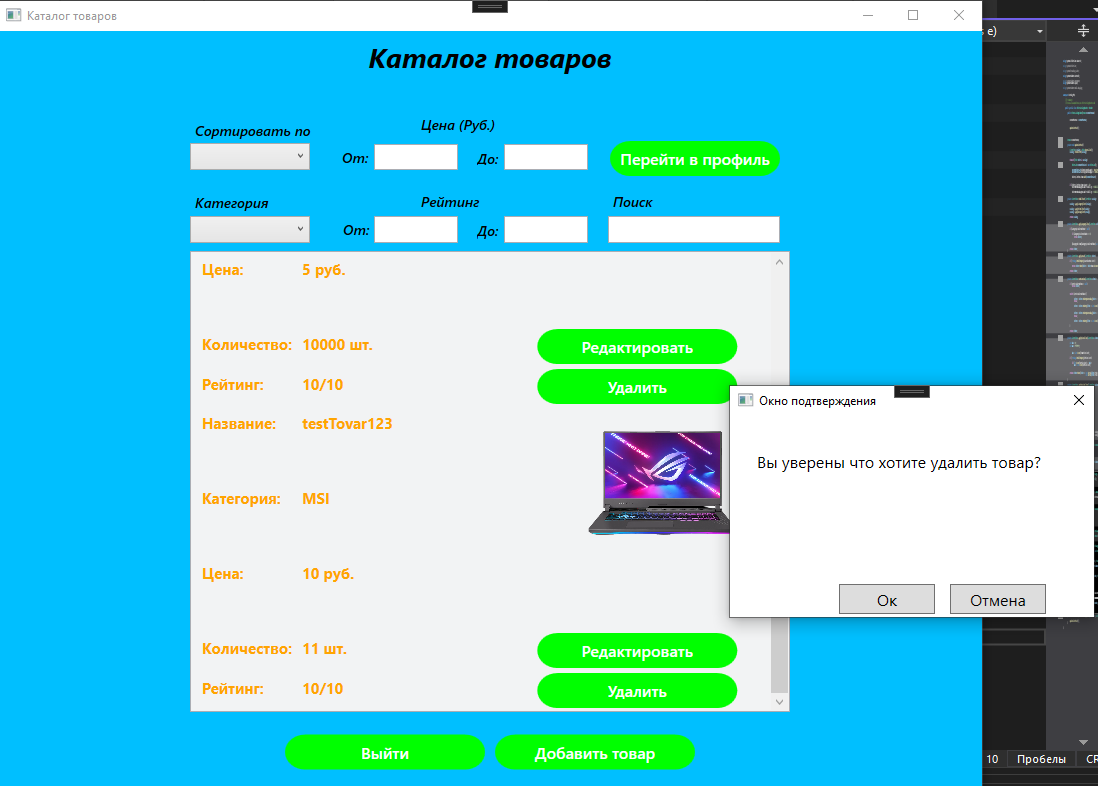
if (result == false)

return;

Offer.DeleteOffer(((sender as UserControl).DataContext as Offer).Id);

updateListView();

}



Код диалогового окна:

public OkOrCancel()

{

InitializeComponent();

this.Top = 500;

this.Left = 500;

}

private void Cansel\_Click(object sender, RoutedEventArgs e)

{

this.DialogResult = false;

this.Close();

}

private void Ok\_Click(object sender, RoutedEventArgs e)

{

this.DialogResult = true;

this.Close();

}

Функция удаления товара по id в классе Offer:

public static void DeleteOffer(long id)

{

List<Offer> offers = deserialize();

offers = offers.Where(o => o.Id != id).ToList();

CastomSerialize.SerializationListToJson<Offer>(Path, offers);

}

**4. Редактирование товара:**

При нажатии на кнопку Редактировать товар открывается тоже окно, что и при добавлении товара:

private void UserControl\_editChanged(object sender, EventArgs e)

{

AddAndEditiongOfferWindow addAndEditiongOfferWindow = new AddAndEditiongOfferWindow(currentPerson, (sender as UserControl).DataContext as Offer);

addAndEditiongOfferWindow.Show();

this.Close();

}

При инициализации окна вызывается функция loadData, которая автоматически подстовляет данные товара в поля:

private void loadData(Offer offer)

{

titleLabel.Content = "Редактирование товара";

addOrEditingButton.Content = "Сохранить";

nameTextBox.Text = offer.Name;

priceTextBox.Text = offer.Price;

countTextBox.Text = offer.Count;

retingTextBox.Text = offer.Rating.Substring(0, offer.Rating.Length - 3);

if (offer.Image != "")

{

avatarImage.Source = new BitmapImage(new Uri($"{offer.Image}"));

imagePath = offer.Image;

}

if (offer.Category.ToLower() == "msi")

{

categoryCB.SelectedIndex = 0;

}

else if (offer.Category.ToLower() == "gigabyte")

{

categoryCB.SelectedIndex = 1;

}

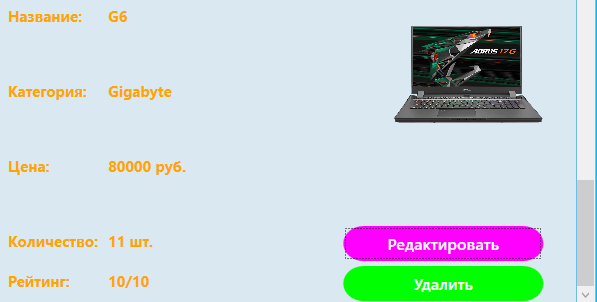
else

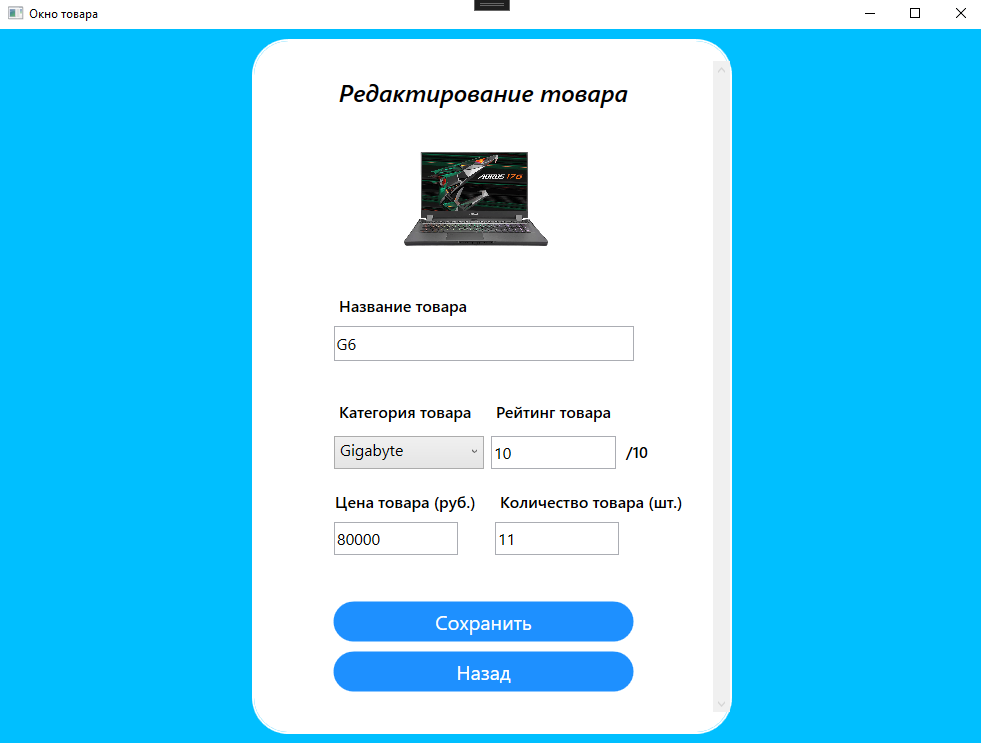
{

categoryCB.SelectedIndex = 2;

}

}





При нажатии на кнопку вызывается функция saveData которая проверяет новые данные на валидацию и вызывает функцию newOffer

private void save\_Click(object sender, RoutedEventArgs e)

{

string response = validation();

if (response != "")

{

MessageBox.Show(response);

return;

}

newOffer();

}

Функция newOffer создает новый товар на основе данных из полей и вызываетс функцию EditingOffer

private void newOffer()

{

Offer offer = new Offer(Offer.getId(), nameTextBox.Text, (categoryCB.SelectedItem as ComboBoxItem).Content.ToString(),

countTextBox.Text + " шт.", priceTextBox.Text + " руб.", retingTextBox.Text + "/10", imagePath);

if (currentOffer == null)

{

bool hava = false;

Offer haveOffer = Offer.offerExists(nameTextBox.Text, (categoryCB.SelectedItem as ComboBoxItem).Content.ToString(),

retingTextBox.Text + "/10", priceTextBox.Text + " руб.");

if ( haveOffer != null)

{

offer = haveOffer;

offer.Count = (int.Parse(offer.Count.Substring(0, offer.Count.Length-4)) + int.Parse(countTextBox.Text)) +" шт.";

Offer.EditingOffer(offer.Id, offer);

MessageBox.Show("Товар добавлен");

backFunction();

return;

}

Offer.AddOffer(offer);

MessageBox.Show("Товар добавлен");

backFunction();

return;

}

offer.Id = currentOffer.Id;

Offer.EditingOffer(currentOffer.Id, offer);

MessageBox.Show("Товар изменен");

backFunction();

}

Функция EditingOffer ищет товар по id и заменят его на передоваемый товар, после измененный лист сериализуется в файл

public static void EditingOffer(long id, Offer newOffer)

{

List<Offer> offers = deserialize();

for (int i = 0; i < offers.Count; i++)

{

if (offers[i].Id == id)

{

offers[i] = newOffer;

break;

}

}

CastomSerialize.SerializationListToJson<Offer>(Path, offers);

}

**5. Поиска товара:**

При вводе текста в поле поиска вызывается функция updateListView, которая вызывает функцию chekFilters:

private void updateListView()

{

List<Offer> catalog = Offer.deserialize();

catalog = chekFilters(catalog);

offers\_ListView.Items.Clear();

foreach (Offer offer in catalog)

{

OfferCard currentOfferCard = new OfferCard();

currentOfferCard.DataContext = offer;

currentOfferCard.deleteEventeChanged += UserControl\_deleteChanged;

currentOfferCard.editingEventeChanged += UserControl\_editChanged;

offers\_ListView.Items.Add(currentOfferCard);

}

if (offers\_ListView.Items.Count != 0)

offersWereRunningOutLabel.Visibility = Visibility.Hidden;

else

offersWereRunningOutLabel.Visibility = Visibility.Visible;

}

private List<Offer> chekFilters(List<Offer> catalog)

{

catalog = applyCategoryFilter(catalog);

catalog = applySerach(catalog);

catalog = applyPriceFilter(catalog);

catalog = applyRatingFilter(catalog);

catalog = sortFunction(catalog);

return catalog;

}

Функция chechFilter вызывает функции поиска, сортировки и фильтрации, оставляя только те товары, которые соответствуют критериям. Функция applySerarch ищет совпадения названий товаров в списке и текста в поле поиска через метод метод Contains и LINQ:

private List<Offer> applySerach(List<Offer> offers)

{

if (!string.IsNullOrEmpty(searchTextBox.Text))

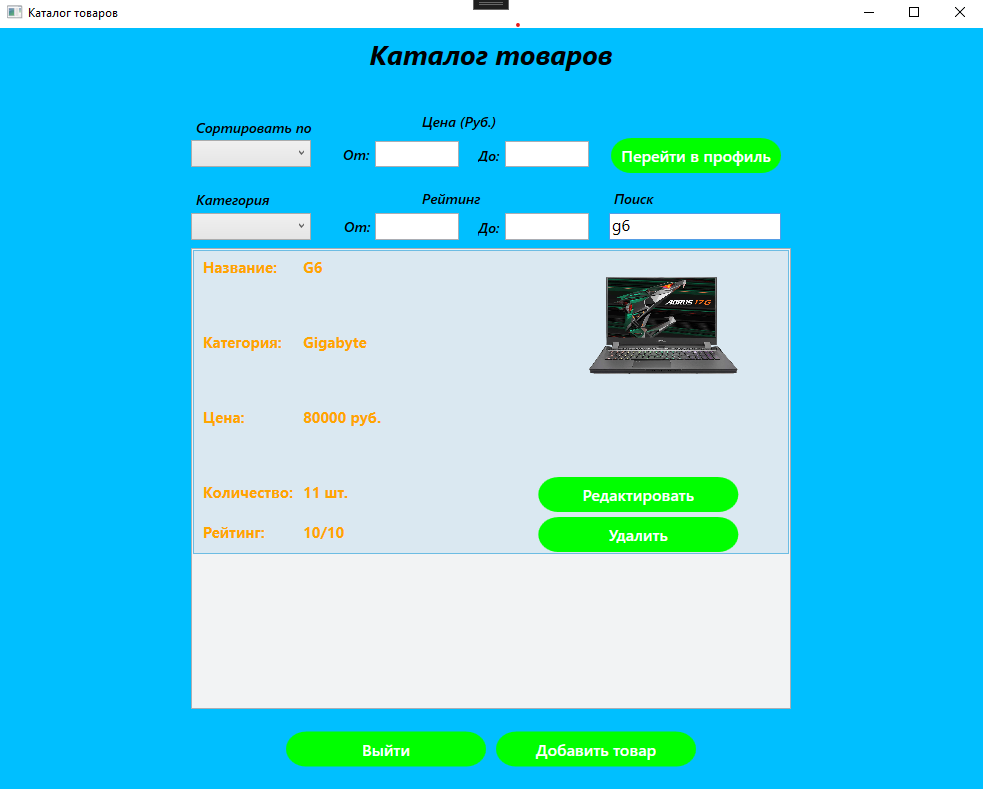
{

return offers.Where(offer => offer.Name.ToLower().Contains(searchTextBox.Text.ToLower())).ToList();

}

return offers;

}



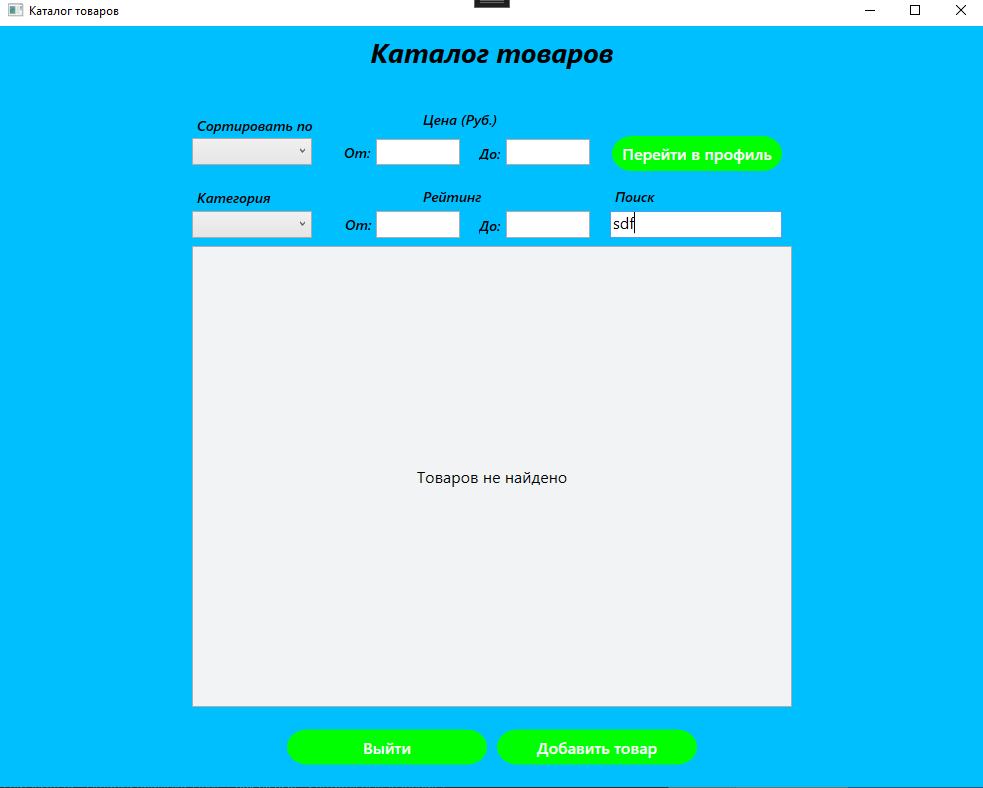
Если кол-во соответствующих товаров равно 0, то в центре экрана будет отображаться лейбл offersWereRunningOutLabel c текстом: “Товаров не найдено”

if (offers\_ListView.Items.Count != 0)

offersWereRunningOutLabel.Visibility = Visibility.Hidden;

else

offersWereRunningOutLabel.Visibility = Visibility.Visible;



**6. Cортировка товаров:**

Функция checkFilters вызывает функцию sortFunction, которая изменяет порядок элементов в списке через метод OrderBy и OrderByDescending по полям цена и рейтинг и возвращает отсортированный список:

private List<Offer> sortFunction(List<Offer> offers)

{

if (sortCB.SelectedItem == null)

return offers;

switch (sortCB.SelectedIndex) {

case 0:

offers = offers.OrderByDescending(offer => int.Parse(offer.Rating.Substring(0, offer.Rating.Length - 3))).ToList();

break;

case 1:

offers = offers.OrderBy(offer => int.Parse(offer.Rating.Substring(0, offer.Rating.Length - 3))).ToList();

break;

case 2:

offers = offers.OrderByDescending(offer => int.Parse(offer.Price.Substring(0, offer.Price.Length - 4))).ToList();

break;

case 3:

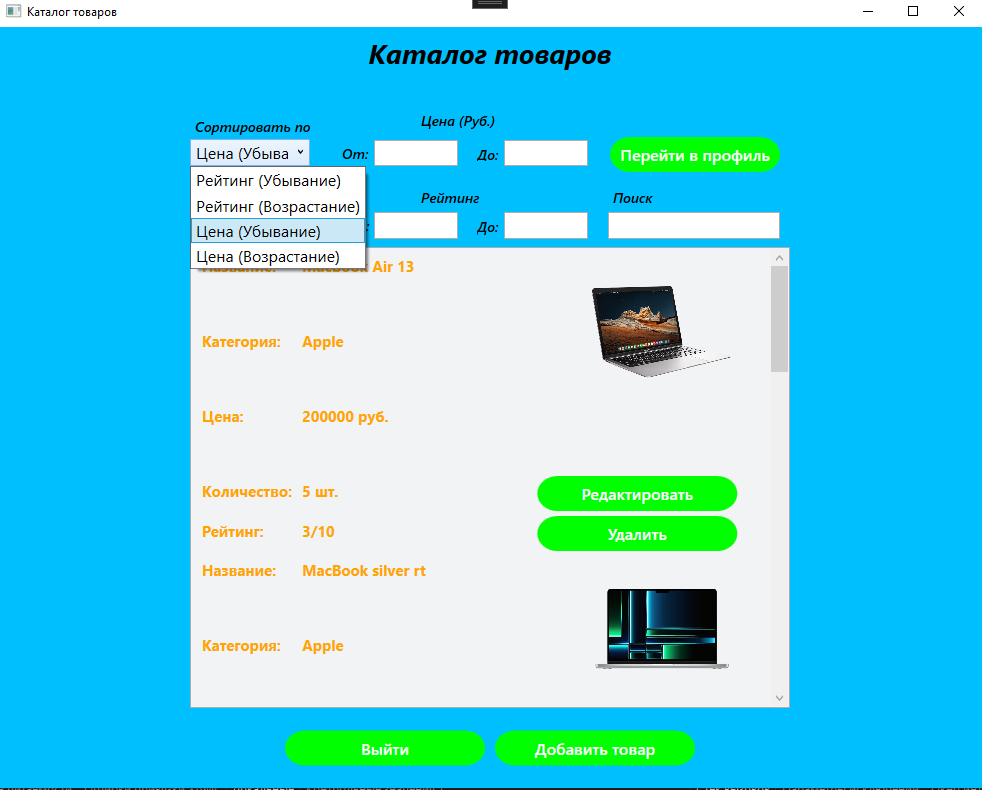
offers = offers.OrderBy(offer => int.Parse(offer.Price.Substring(0, offer.Price.Length - 4))).ToList();

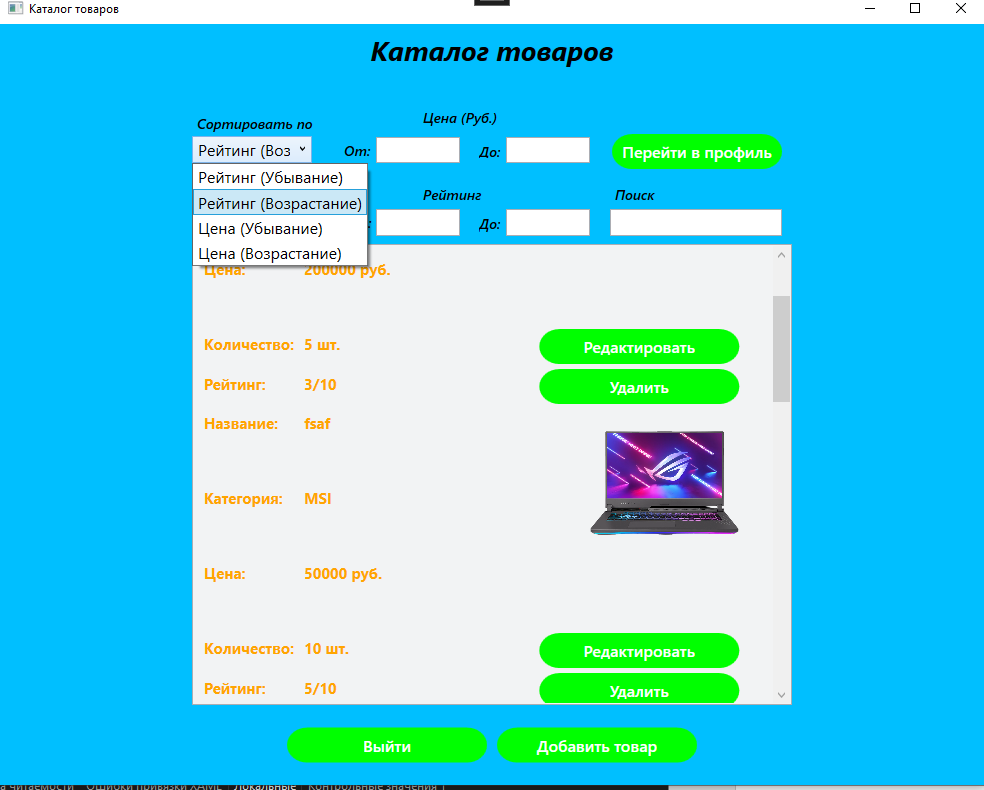
break;

}

return offers;

}





**7. Фильтрация товаров:**

**1) По категории**

Функция checkFilters вызывает функцию applyCategoryFilter, которая остовляет только те элементы в списке, которые соответствую выбранной категори в ComboBox через метод where и LINQ и возвращает отсортированный список:

private List<Offer> applyCategoryFilter(List<Offer> offers)

{

if (kategoryCB.SelectedItem != null)

{

if (kategoryCB.SelectedIndex == 0)

return offers;

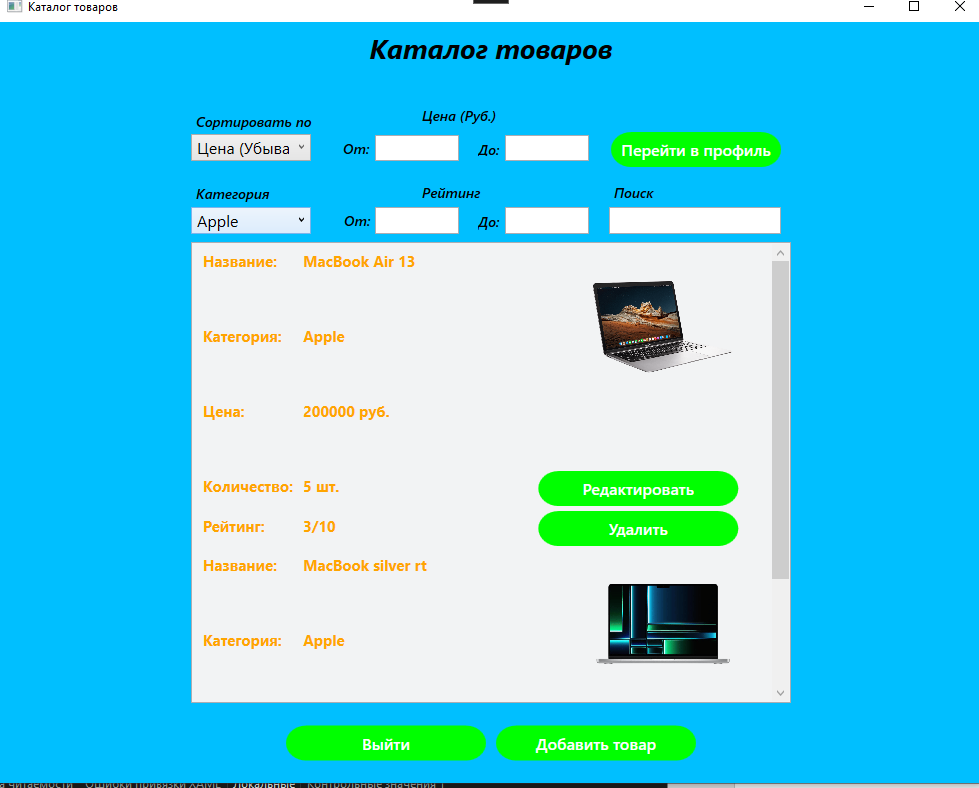
return offers.Where(offer => offer.Category.ToLower() ==

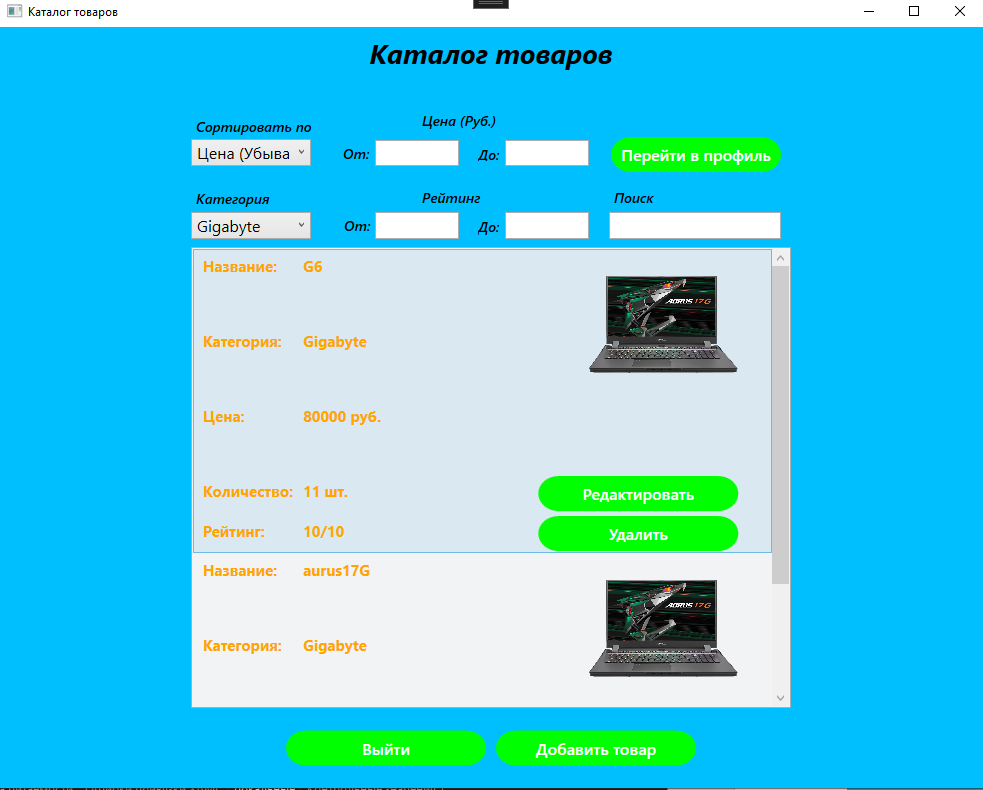
(kategoryCB.Items[kategoryCB.SelectedIndex] as ComboBoxItem).Content.ToString().ToLower()).ToList();

}

return offers;

}

****

****

**2) По рейтингу (диапазон)**

Функция checkFilters вызывает функцию applyRaringFilter, которая

получает нижнею и верхнею границу по рейтингу и ищит элементы у которых рейтинг в находиться в этом диапазоне и возврощает отсортированый список

private List<Offer> applyRatingFilter(List<Offer> offers)

{

int min = 1;

int max = 10;

if (!string.IsNullOrEmpty(fromRating.Text))

{

min = int.Parse(fromRating.Text);

}

if (min < 1)

min = 1;

if (!string.IsNullOrEmpty(toRating.Text))

{

if (int.Parse(toRating.Text) >= min)

max = int.Parse(toRating.Text);

}

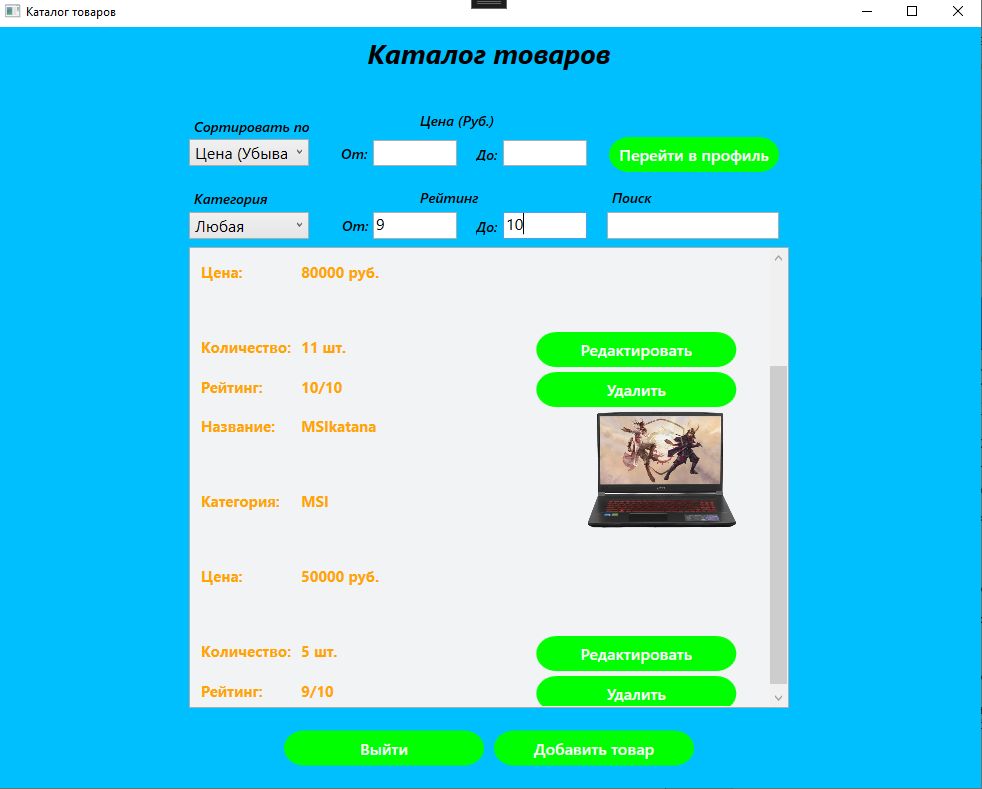
if (max > 10)

max = 10;

return offers.Where(offer => int.Parse(offer.Rating.Substring(0, offer.Rating.Length - 3)) >= min &&

int.Parse(offer.Rating.Substring(0, offer.Rating.Length - 3)) <= max).ToList();

}

****

**3) По цене (диапазон)**

Функция checkFilters вызывает функцию applyPriceFilter, которая

получает нижнею и верхнею границу по цене и ищит элементы у которых цена в находиться в этом диапазоне и возврощает отсортированый список

private List<Offer> applyPriceFilter(List<Offer> offers)

{

int min = 0;

int max = 9999999;

if (!string.IsNullOrEmpty(fromPrice.Text)){

min = int.Parse(fromPrice.Text);

}

if (!string.IsNullOrEmpty(toPrice.Text))

{

if (int.Parse(toPrice.Text) >= min)

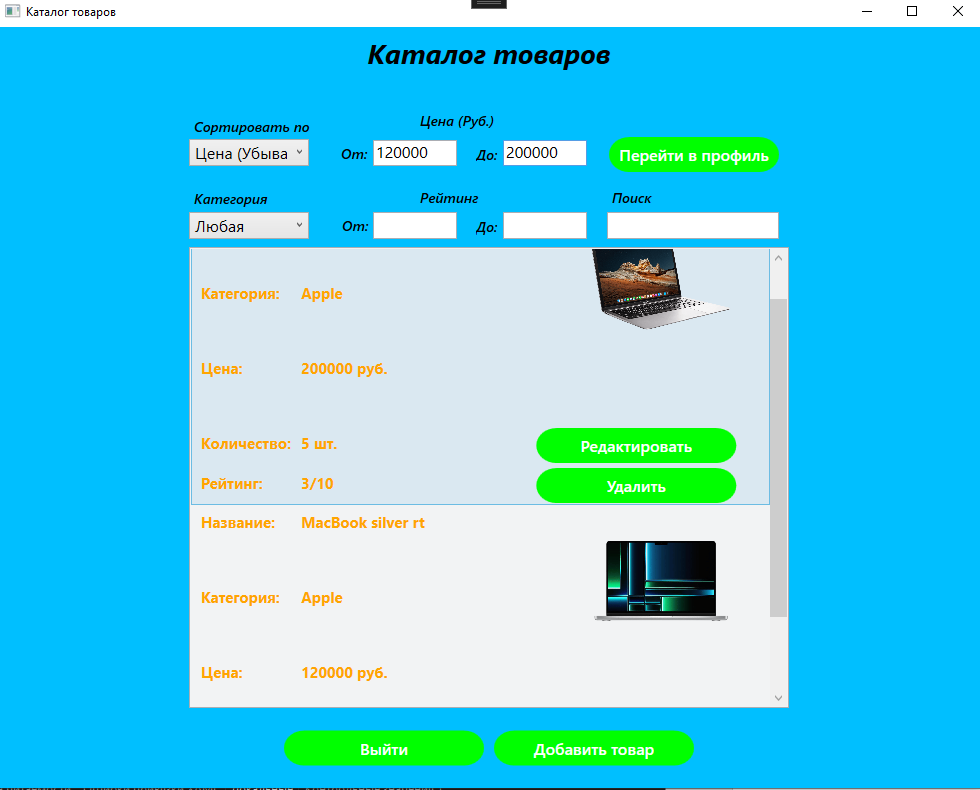
max = int.Parse(toPrice.Text);

}

return offers.Where(offer => int.Parse(offer.Price.Substring(0, offer.Price.Length - 4)) >= min &&

int.Parse(offer.Price.Substring(0, offer.Price.Length - 4)) <= max).ToList();

}



**8. Обновление списка товаров:**

Любые изменения в полях в окне каталога товаров вызывают функцию updateListView динамически обновляя каталог товаров

private void sortCB\_SelectionChanged(object sender, SelectionChangedEventArgs e)

{

updateListView();

}

***Вывод:***

Научился создавать приложения для вывода товаров из JSON файла с использованием UserControl, а так же применять сортировку, фильтрацию и поиск к товарам