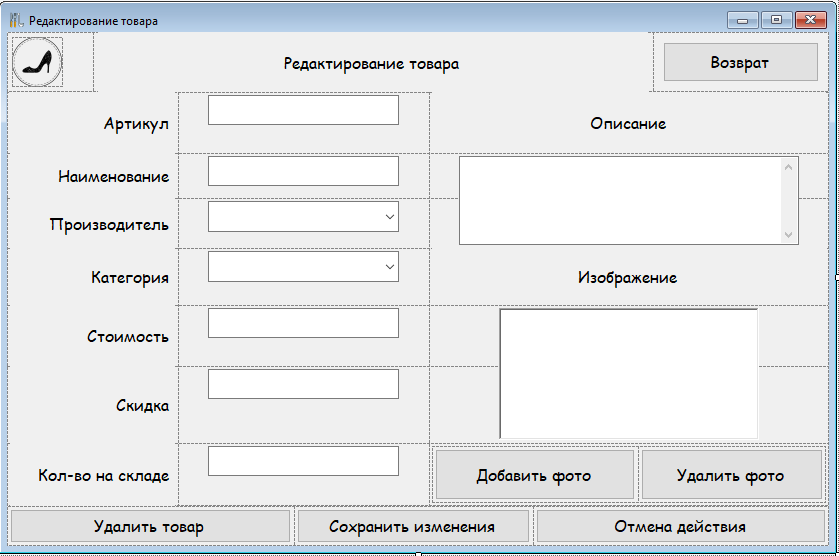
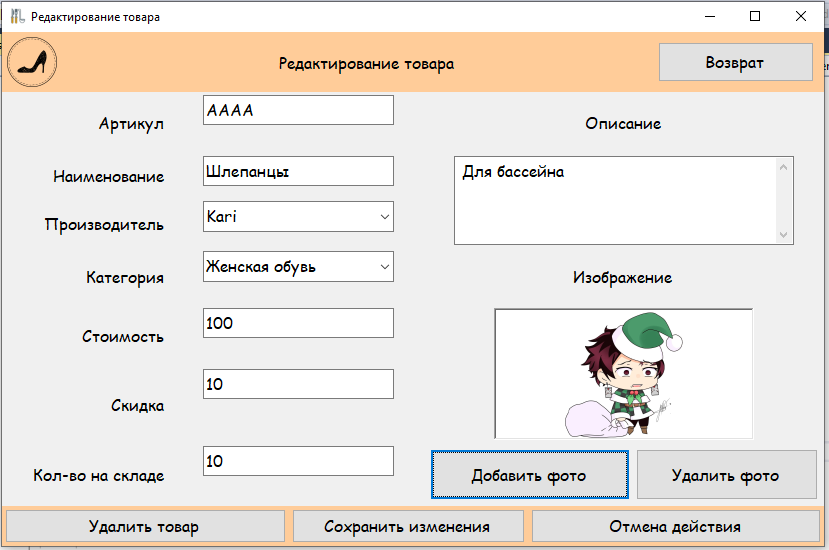
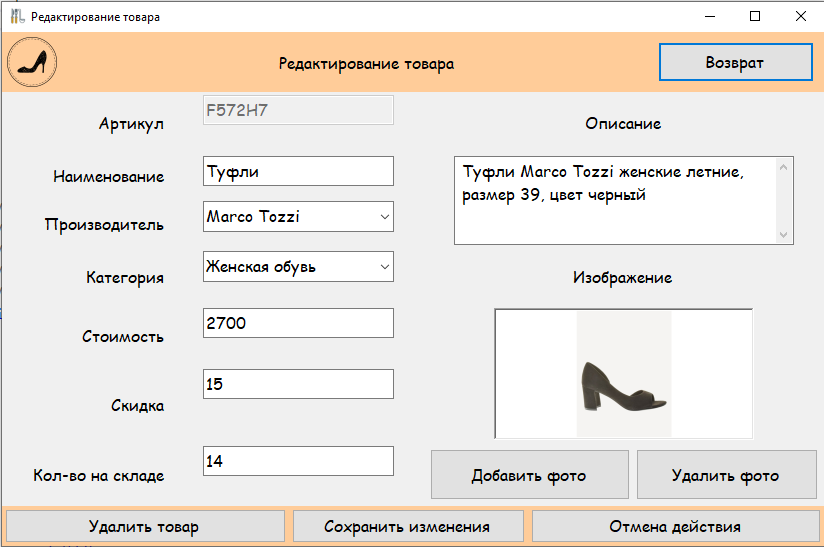
**Реализация редактирование товаров с применением платформы Entity Framework**



Форма в режиме Конструктора



Форма в процессе добавление нового товара



Форма в процессе редактирования товара

**Фрагменты кода**

using System.IO;

namespace Shoes

{

public partial class EditProduct : Form

{

string productArticle; //Переданный артикул товара для редактирования

string photo; //Только имя файла

string pathExe = Application.StartupPath; //Путь к exe без имени файла

string fileSource = ""; //Путь к файлу источнику картинки

/// Конструктор с параметром:

/// пусто - для добавления нового товара

/// артикул - для редактирования существующего

/// <param name="idProduct"></param>

public EditProduct(string **productArticle**)

{

InitializeComponent();

this.productArticle = productArticle;

}

/// Начальные настройки

private void Authorizations\_Load(object sender, EventArgs e)

{

tableLayoutPanelTop.BackColor = Color.FromArgb(255, 204, 153);

tableLayoutPanelBottom.BackColor = Color.FromArgb(255, 204, 153);

//Настройка списка ComboBox для производителя

List<Entities.Manufacturer> manufacturers = Helper.DB.Manufacturer.ToList();

comboBoxManufacture.DataSource = manufacturers;

comboBoxManufacture.DisplayMember = "ManufacturerName";

comboBoxManufacture.ValueMember = "ManufacturerId";

//Настройка списка ComboBox для категории

List<Entities.Category> categories = Helper.DB.Category.ToList();

comboBoxCategory.DataSource = categories;

comboBoxCategory.DisplayMember = "CategoryName";

comboBoxCategory.ValueMember = "CategoryId";

openFileDialogPhoto.Filter = "Только JPG|\*.jpg";

//Настройка компонентов формы в зависимости от режима

if (String.IsNullOrEmpty(productArticle)) //Добавление нового

{

textBoxArticle.Text = textBoxName.Text = textBoxDescription.Text = "";

textBoxCost.Text = textBoxDiscount.Text = textBoxInStock.Text = "0";

pictureBoxPhoto.Image = Properties.Resources.picture;

photo = null;

}

else //Редактирование имеющего товара

{

//Получить данные о товаре по артиклю

Entities.Product product = Helper.DB.Product.

Where(x => x.ProductArticle == productArticle).FirstOrDefault();

//Вывести информацию о товаре

textBoxArticle.Text = product.ProductArticle;

textBoxArticle.Enabled = false;

textBoxName.Text = product.ProductName;

comboBoxManufacture.SelectedValue = product.ManufacturerId;

comboBoxCategory.SelectedValue = product.CategoryId;

textBoxCost.Text = product.ProductCost.ToString();

textBoxDiscount.Text = product.ProductMaxSale.ToString();

textBoxInStock.Text = product.ProductCount.ToString();

textBoxDescription.Text = product.ProductDescription;

photo = product.ProductPhoto;

if (String.IsNullOrEmpty(photo)) //Отсутствие фото у товара

{

pictureBoxPhoto.Image = Properties.Resources.picture;

}

else //Есть фото у товара

{

string s = pathExe + @"\Товар\_import\" + photo;

pictureBoxPhoto.Load(s);

}

}

}

/// Удалить фото

private void buttonDelPhoto\_Click(object sender, EventArgs e)

{

pictureBoxPhoto.Image = Properties.Resources.picture;

photo = null;

}

/// Добавить фото

private void buttonAddPhoto\_Click(object sender, EventArgs e)

{

if(openFileDialogPhoto.ShowDialog()==DialogResult.OK)

{

fileSource = openFileDialogPhoto.FileName; //От куда копировать

//имя файла совпадает с артиклем (Имя будущего файла с картинкой)

photo = textBoxArticle.Text+".jpg";

pictureBoxPhoto.Load(fileSource);

}

}

/// Сохранить изменения в БД

private void buttonFixed\_Click(object sender, EventArgs e)

{

//Получить новые значения полей с валидацией

double cost; //Стоимость

try

{

cost = Convert.ToDouble(textBoxCost.Text);

}

catch (FormatException)

{

MessageBox.Show("В качестве стоимости введено нечисловое значение");

textBoxCost.Clear();

return;

}

if (cost < 0)

{

MessageBox.Show("Стоимость не может быть отрицательным значением");

textBoxCost.Clear();

return;

}

double discount; //Скидка

try

{

discount = Convert.ToDouble(textBoxDiscount.Text);

}

catch (FormatException)

{

MessageBox.Show("В качестве скидки введено нечисловое значение");

textBoxDiscount.Clear();

return;

}

if (discount < 0 || discount > 100)

{

MessageBox.Show("Скидка должна быть из [1; 100] ");

textBoxDiscount.Clear();

return;

}

int count; //Количество

try

{

count = Convert.ToInt32(textBoxInStock.Text);

}

catch (FormatException)

{

MessageBox.Show("В качестве количества введено нечисловое значение");

textBoxInStock.Clear();

return;

}

if (count < 0)

{

MessageBox.Show("Количество не может быть отрицательным");

textBoxInStock.Clear();

return;

}

string artice = textBoxArticle.Text; //Артикул для поиска или добавления

Entities.Product product= null; //Обрабатываемый товар

if (String.IsNullOrEmpty(productArticle)) //Добавление нового товара

{

//Поиск артикля для отсутствия повторения

Entities.Product productFind = Helper.DB.Product.**Find**(artice);

if (productFind != null)

{

MessageBox.Show("Такой артикул есть в БД");

return;

}

//Такого артикля нет, значит добавляем новый товар

**product = new Entities.Product();** //Создали новый товар

}

else //Редактирование товара

{

product = Helper.DB.Product.**Find**(artice); //Найти товар по артиклю

}

//Заполняем все поля товара

product.ProductArticle = artice;

product.ProductName = textBoxName.Text;

product.UnitId = 1;

product.ProductCost = cost;

product.ProductMaxSale = discount;

product.ManufacturerId = (int)comboBoxManufacture.SelectedValue;

product.ProviderId = 1;

product.CategoryId = (int)comboBoxCategory.SelectedValue;

product.ProductSale = discount;

product.ProductCount = count;

product.ProductDescription = textBoxDescription.Text;

product.ProductPhoto = photo;

if (String.IsNullOrEmpty(productArticle)) //Добавление нового товара

{

Helper.DB.Product.**Ad**d(product);

}

try

{

Helper.DB.**SaveChanges**();

MessageBox.Show("Информация в БД успешно обновлена");

}

catch

{

MessageBox.Show("При обновлении данных в БД возникла ошибка");

return;

}

this.Close();

//Работа с папкой фото

if (!String.IsNullOrEmpty(photo)) //Изображение выбрано

{

string fileDesc = pathExe + @"\Товар\_import\" + photo; //Куда копировать

if (File.Exists(fileDesc)) //В случае редактирования такой файл уже может быть

{

File.Delete(fileDesc); //Удаляем его

}

File.Copy(fileSource, fileDesc); //Переносим в папку с фото выбранное изображение

}

}

/// Удалить товар

private void buttonDelProduct\_Click(object sender, EventArgs e)

{

if(MessageBox.Show("Вы действительно хотите удалить этот товар","Контроль",

MessageBoxButtons.OKCancel)==DialogResult.OK)

{

//Поиск товара по ключевому полю (артикль) для удаления

Entities.Product productDelete = Helper.DB.Product.**Find**(productArticle);

Helper.DB.Product.**Remove**(productDelete); //Удаление из виртуальной таблицы

try

{

Helper.DB.**SaveChanges**(); //Сохранение изменений в БД

MessageBox.Show("Товар удален успешно");

}

catch

{

MessageBox.Show("Товар нельзя удалить, т.к. он есть в заказах");

return;

}

}

this.Close();

}

}

}