Министерство науки и высшего образования Российской Федерации

**Федеральное государственное бюджетное образовательное учреждение высшего образования**

«Владимирский государственный университет

имени Александра Григорьевича и Николая Григорьевича Столетовых»

**(ВлГУ)**

**Кафедра информационных систем и программной инженерии**

Лабораторная работа №7

по дисциплине «Технологии программирования»

Тема: «Язык интегрированных запросов LINQ to Entities»

Выполнил:

студент гр. ПРИ-123

А.Ц. Нямаа

Принял:

Ст. пр. кафедры ИСПИ

В.В. Данилов

Владимир, 2025 г.

ЦЕЛЬ РАБОТЫ:

Познакомиться с технологией использования языка интегрированных запросов LINQ to Entities.

ЗАДАНИЕ (вариант 12).

В созданной в предыдущих лабораторных работах программе добавить следующие возможности: при выборе определенной группы должен производиться показ контактов в выбранной группе, на странице Details.aspx добавить вывод группы, к которой принадлежит контакт и создать раздел в веб-приложении, который позволит управлять группами контактов. Работу с данными организовать через LINQ to Entities.

ВЫПОЛНЕНИЕ РАБОТЫ:

Для выполнения данного задания в ранее написанном веб-приложение все прямые SQL-запросы были заменены на работу с LINQ to Entities. Также был реализован вывод продуктов по группам, в которых они состоят. На странице Details добавил вывод групп, к которым принадлежит товар. Была реализована работа с группами: изменение цены выбранной группы.

Листинг ApplicationContext.cs:

using Microsoft.EntityFrameworkCore;

using ProductModel.Models;

using OrderModel.Models;

namespace task\_beta.Models

{

public class ApplicationContext : DbContext

{

public DbSet<Product> products { get; set; }

public DbSet<Order> orders { get; set; }

public ApplicationContext(DbContextOptions<ApplicationContext> options) : base(options)

{

Database.EnsureCreated();

}

protected override void OnModelCreating(ModelBuilder modelBuilder)

{

modelBuilder.Entity<Product>().ToTable("product");

modelBuilder.Entity<Order>().ToTable("orderinfo");

modelBuilder.Entity<Order>(entity =>

{

entity.Property(o => o.OrderEmail).IsRequired(false);

});

}

}

}

Листинг homeController.cs:

using Microsoft.AspNetCore.Mvc;

using OrderModel.Models;

using ProductModel.Models;

using task\_beta.Models;

namespace task\_beta.Controllers

{

public class HomeController : Controller

{

private readonly ApplicationContext db;

public HomeController(ApplicationContext db)

{

this.db = db;

}

private List<Product> products = new List<Product>();

[BindProperty]

public Order order { get; set; } = new Order();

[HttpGet]

public async Task<IActionResult> Index(string selectType = "All")

{

products.Clear();

if (selectType == "All")

products = db.products.ToList();

else

products = db.products.Where(p => p.ProductType == selectType).ToList();

ViewBag.Products = products;

ViewBag.SelectedType = selectType;

ViewBag.succesValid = successValid;

ViewBag.errorValid = errorValid;

return View();

}

private string successValid = "";

private string errorValid = "";

[HttpPost]

public async Task<IActionResult> Order(Order order)

{

if (!ModelState.IsValid)

{

errorValid = "Некоторые данные были введены неправильно!";

return RedirectToAction("Index");

}

db.orders.Add(new Order()

{

OrderName = order.OrderName,

OrderSurname = order.OrderSurname,

OrderAddress = order.OrderAddress,

OrderQuantity = order.OrderQuantity

});

await db.SaveChangesAsync();

successValid = $"Спасибо {order.OrderName} {order.OrderSurname}, заказ выполнен успешно!{Environment.NewLine}" +

$"Доставим товар по адресу {order.OrderAddress} в количестве {order.OrderQuantity} шт.{Environment.NewLine}" +

$"Электронный чек придет в ближайшее время на почту {order.OrderEmail}";

return RedirectToAction("Index");

}

}

}

Листинг observerController.cs:

using Microsoft.AspNetCore.Mvc;

using task\_beta.Models;

using ProductModel.Models;

namespace task\_beta.Controllers

{

public class watchController : Controller

{

private readonly ApplicationContext db;

public watchController(IConfiguration configuration, ApplicationContext db)

{

this.db = db;

}

private static List<Product> products = new List<Product>();

public IActionResult Observer()

{

products.Clear();

products = db.products.ToList();

ViewBag.Products = products;

return View();

}

}

}

Листинг groupWorkController.cs:

using Microsoft.AspNetCore.Mvc;

using ProductModel.Models;

using task\_beta.Models;

namespace task\_beta.Controllers

{

public class groupWorkController : Controller

{

private readonly ApplicationContext db;

public groupWorkController(ApplicationContext db)

{

this.db = db;

}

[HttpGet]

public IActionResult GroupWork()

{

return View();

}

public Product product;

[HttpPost]

public IActionResult ChangePrice(string type, int percent)

{

List<Product> productsToUpdate;

if (type == "all")

{

productsToUpdate = db.products.ToList();

}

else

{

productsToUpdate = db.products.Where(p => p.ProductType == type).ToList();

}

if (productsToUpdate.Any())

{

foreach (var product in productsToUpdate)

{

product.ProductCost = (int)Math.Round(product.ProductCost \* (1 + percent / 100.0));

}

db.SaveChanges();

}

return RedirectToAction("GroupWork", "GroupWork");

}

}

}

Листинг editController.cs:

using Microsoft.AspNetCore.Mvc;

using OrderModel.Models;

using task\_beta.Models;

namespace task\_beta.Controllers

{

public class editController : Controller

{

private readonly ApplicationContext db;

public editController(ApplicationContext db)

{

this.db = db;

}

public Order order { get; set; } = new Order();

[HttpGet]

public IActionResult Editor()

{

return View();

}

[HttpPost]

public IActionResult EditorPost(Order order)

{

var currentOrder = db.orders.Where(o => o.OrderId == order.OrderId).FirstOrDefault();

if (currentOrder != null)

{

currentOrder.OrderName = order.OrderName;

currentOrder.OrderSurname = order.OrderSurname;

currentOrder.OrderEmail = order.OrderEmail;

currentOrder.OrderAddress = order.OrderAddress;

currentOrder.OrderQuantity = order.OrderQuantity;

db.orders.Update(currentOrder);

db.SaveChanges();

}

return RedirectToAction("Editor", "Edit");

}

}

}

Листинг deleteController.cs:

using Microsoft.AspNetCore.Mvc;

using OrderModel.Models;

using task\_beta.Models;

namespace task\_beta.Controllers

{

public class deleteController : Controller

{

private readonly ApplicationContext db;

public deleteController(ApplicationContext db)

{

this.db = db;

}

public Order order = new Order();

[HttpGet]

public IActionResult Deleter()

{

return View();

}

[HttpPost]

public IActionResult DeleterPost(Order order)

{

var currentOrder = db.orders.Where(o => o.OrderId == order.OrderId).FirstOrDefault();

db.orders.Remove(currentOrder);

db.SaveChanges();

return RedirectToAction("Deleter", "Delete");

}

}

}

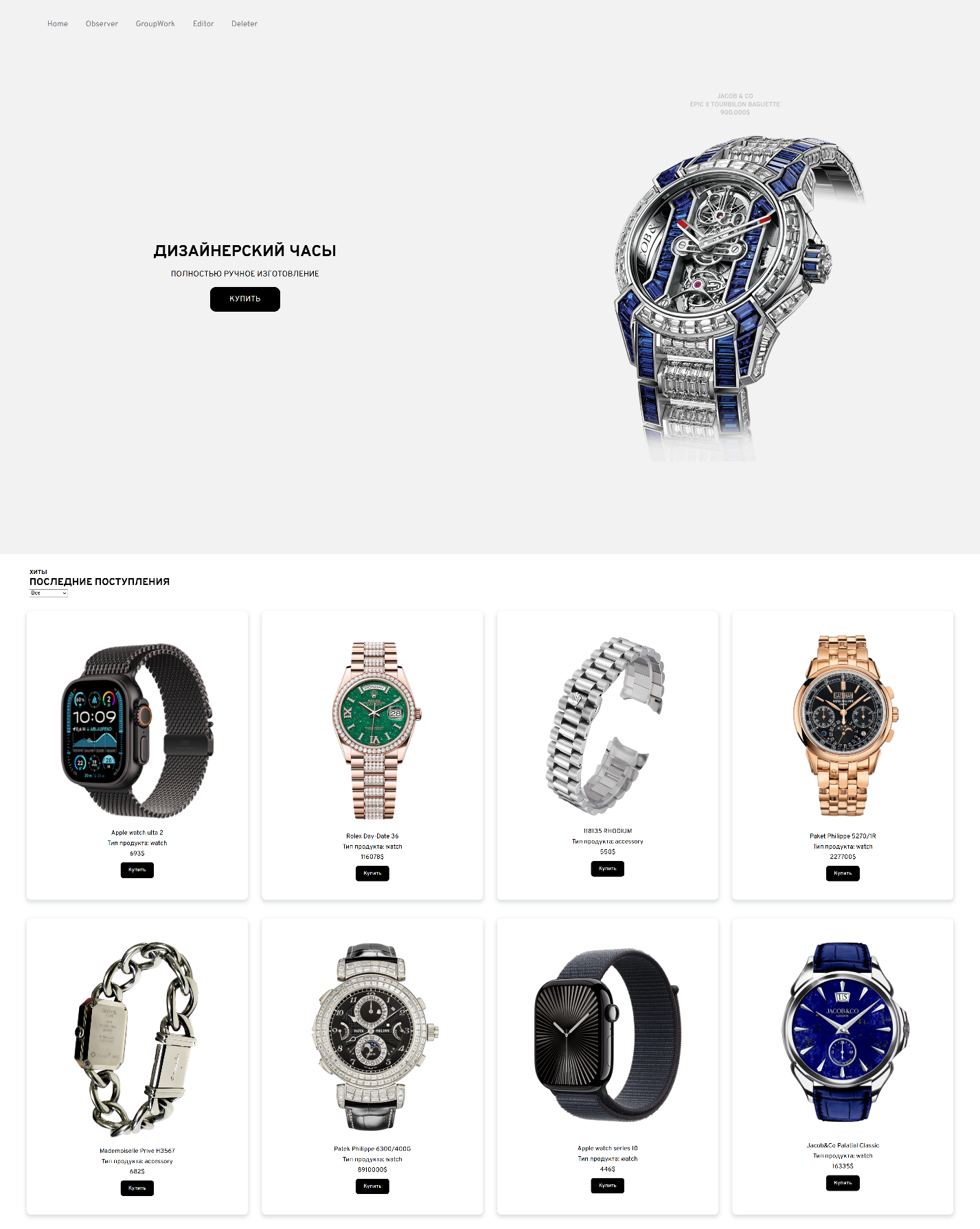


Рисунок 1. Демонстрация работы главной страницы.

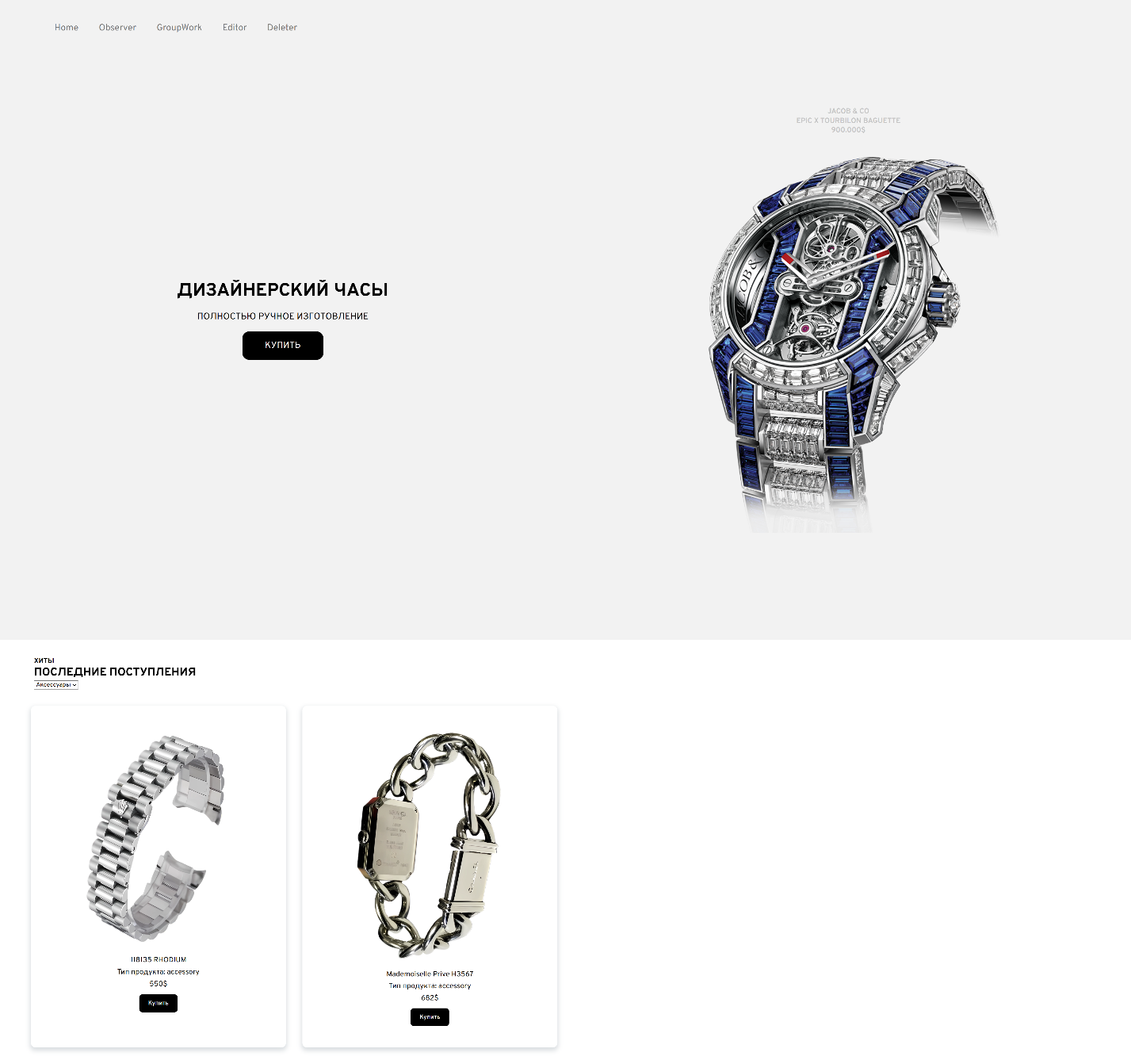


Рисунок 2. Главная страница с выбранной группой.

  
Рисунок 3. Страница детального просмотра.

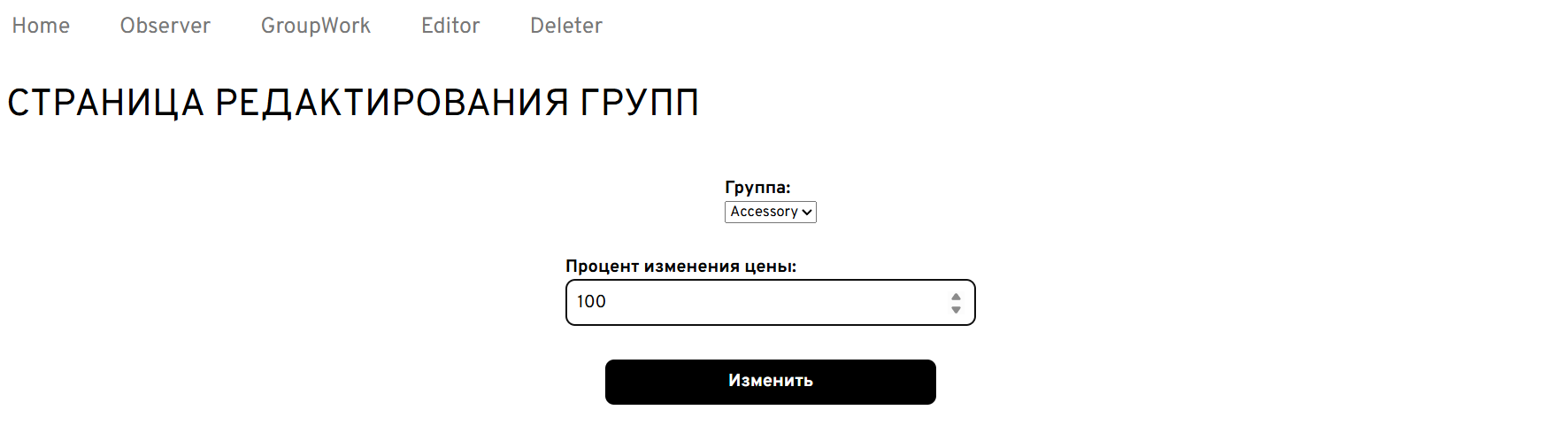


Рисунок 4. Страница редактирования групп.

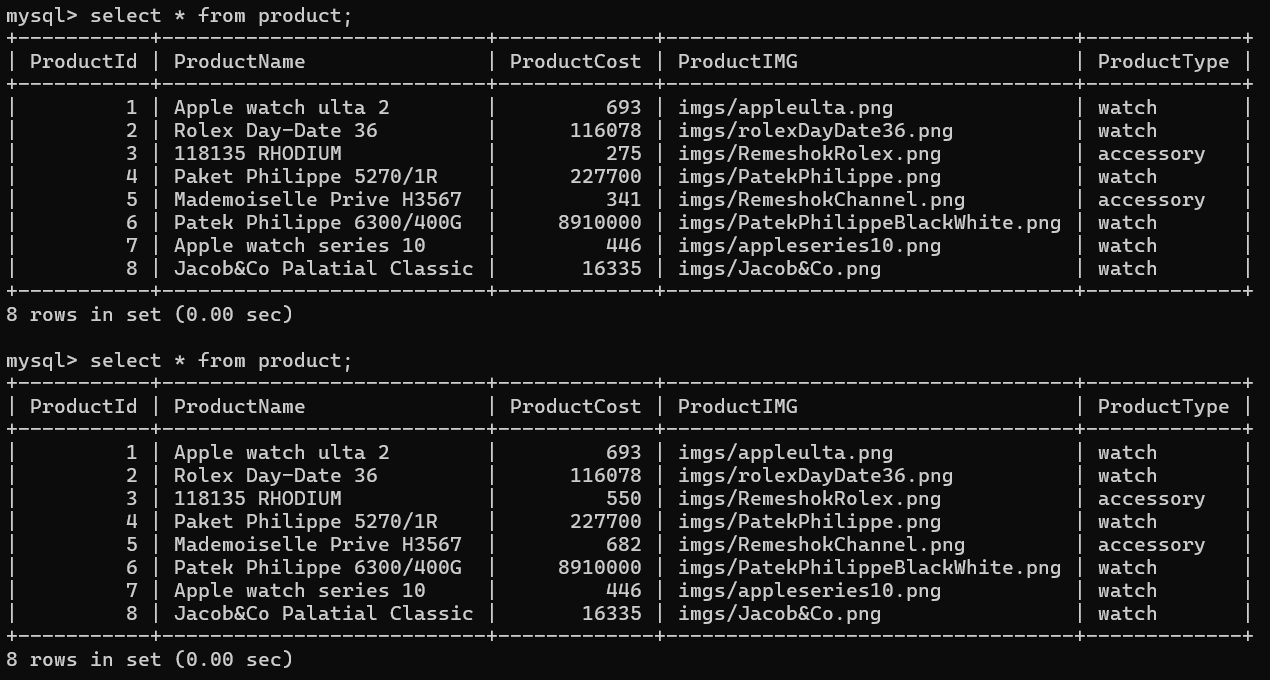


Рисунок 5. Изменение цены в базе данных.

ВЫВОД К РАБОТЕ:

Я познакомился с технологией использования языка интегрированных запросов LINQ to Entities.