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 №2

по дисциплине «Технологии программирования»

Тема: «Работа с данными в ASP.NET»

Выполнил:

студент гр. ПРИ-123

А.Ц. Нямаа

Принял:

Ст. пр. кафедры ИСПИ

В.В. Данилов

Владимир, 2025 г.

ЦЕЛЬ РАБОТЫ:

Познакомиться с рядом средств ASP.NET для представления и отображения данных.

ЗАДАНИЕ (вариант 12).

Составить свой пример, используя вместо компонента Repeater компоненты DataList или GridView, а также реализовать проверку данных на своем сайте.

ВЫПОЛНЕНИЕ РАБОТЫ:

Для множественного отображения данных я использовал стандартный метод (цикл foreach), так как в Razor Pages нет компонентов Repeater, DataList и GridView.

Листинг Index.cshtml:

@page

@using calculator.Pages

@model calculator.Pages.IndexModel

@{

}

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<link rel="preconnect" href="https://fonts.googleapis.com">

<link rel="preconnect" href="https://fonts.gstatic.com" crossorigin>

<link href="https://fonts.googleapis.com/css2?family=Overpass:ital,wght@0,100..900;1,100..900&display=swap" rel="stylesheet">

<title>Clock Shop</title>

<link rel="stylesheet" href="~/css/StyleSheet.css">

</head>

<body>

<header>

<div class="leftText">

<h1> ДИЗАЙНЕРСКИЙ ЧАСЫ </h1>

<p> ПОЛНОСТЬЮ РУЧНОЕ ИЗГОТОВЛЕНИЕ </p>

<button class="buyHeader buttonToChange openModel" data-price="900000"> КУПИТЬ </button>

<dialog class="dialogMain">

<form class="orderForm" method="post">

<h2>Оформление заказа</h2>

@if (Model.successValid != "")

{

<p class="validSuccess">@Model.successValid</p>

}

else if (Model.errorValid != "")

{

<p class="validError">@Model.errorValid</p>

}

<p>

<label asp-for="order.NameOrder">Имя:</label>

<input type="text" id="name" asp-for="order.NameOrder" placeholder="Имя">

<span asp-validation-for="order.NameOrder" class="text-danger"></span>

</p>

<p>

<label asp-for="order.SecondNameOrder">Фамилия:</label>

<input type="text" id="SecondNameOrder" asp-for="order.SecondNameOrder" placeholder="Фамилия">

<span asp-validation-for="order.SecondNameOrder" class="text-danger"></span>

</p>

<p>

<label asp-for="order.EmailOrder">Почтовый ящик:</label>

<input type="email" id="email" asp-for="order.EmailOrder" placeholder="Почтовый ящик">

<span asp-validation-for="order.EmailOrder" class="text-danger"></span>

</p>

<p>

<label asp-for="order.AddressOrder">Адрес доставки:</label>

<input type="text" id="address" asp-for="order.AddressOrder" placeholder="Ул. Пушкино, д. 2, кв. 1">

<span asp-validation-for="order.AddressOrder" class="text-danger"></span>

</p>

<p>

<label asp-for="order.QuantityOrder">Количество:</label>

<input type="number" id="quantity" asp-for="order.QuantityOrder" placeholder="1">

<span asp-validation-for="order.QuantityOrder" class="text-danger"></span>

</p>

<p>

<label>Промокод:</label>

<input type="text" id="promo" placeholder="\*Промокод\*">

</p>

<p class="priceHeader"></p>

<button> Заказать! </button>

</form>

<button id="closeModel" class="closeModel">&#x2715 </button>

</dialog>

</div>

<div class="rightText">

<p> JACOB & CO <br>EPIC X TOURBILON BAGUETTE <br>900.000$</p>

<img src="imgs/mainPhotoWatches.png">

</div>

</header>

<main>

<div class="textInMain">

<p>ХИТЫ</p>

<p>ПОСЛЕДНИЕ ПОСТУПЛЕНИЯ</p>

</div>

<div class="productsList">

@foreach(var product in Model.products)

{

<div class="productUnit">

<img src="@product.imagePathProduct" />

<p>@product.NameProduct</p>

<p>@($"{product.CostProduct}$")</p>

<button class="buttonUnit buttonToChange openModel" data-price="@product.CostProduct"> Купить </button>

</div>

}

</div>

<div class="addModule orderContainer ">

<div class="textInMain addText textOrder">

<p>ЗАКАЗЫ</p>

</div>

<div class="orderItems">

<div class="orderItem">

<p class="singleItem">Id заказа</p>

<p class="singleItem">Имя покупателя</p>

<p class="singleItem">Фамилия покупателя</p>

<p class="singleItem">Почта покупателя</p>

<p class="singleItem"> Адресс доставки</p>

<p class="singleItem">Количество товара</p>

</div>

</div>

<div class="orderItems">

@foreach (var order in IndexModel.orders)

{

<div class="orderItem">

<p class="singleItem">@order.orderListId</p>

<p class="singleItem">@order.nameCustomer</p>

<p class="singleItem">@order.secondNameCustomer</p>

<p class="singleItem">@order.mailCustomer</p>

<p class="singleItem"> @order.addressCustomer</p>

<p class="singleItem">@order.quantityCustomer</p>

</div>

}

</div>

</div>

</main>

<script src="~/scripts/JavaScript.js"></script>

</body>

</html>

Листинг Index.cshtml.cs:

using Microsoft.AspNetCore.Mvc;

using Microsoft.AspNetCore.Mvc.RazorPages;

using task2.Models;

namespace calculator.Pages

{

[IgnoreAntiforgeryToken]

public class IndexModel : PageModel

{

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

[BindProperty]

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

public void OnGet()

{

products.Add(new Product { IdProduct = 1, NameProduct = "Apple watch ulta 2", CostProduct = 700, imagePathProduct = "imgs/appleulta.png" });

products.Add(new Product { IdProduct = 2, NameProduct = "Rolex Day-Date 36", CostProduct = 117250, imagePathProduct = "imgs/rolexDayDate36.png" });

products.Add(new Product { IdProduct = 3, NameProduct = "King Seiko Automatic", CostProduct = 2000, imagePathProduct = "imgs/KingSeiko.png" });

products.Add(new Product { IdProduct = 4, NameProduct = "Paket Philippe 5270/1R", CostProduct = 230000, imagePathProduct = "imgs/PatekPhilippe.png" });

products.Add(new Product { IdProduct = 5, NameProduct = "Rolex DateJust 31", CostProduct = 51900, imagePathProduct = "imgs/rolexDateJust31.png" });

products.Add(new Product { IdProduct = 6, NameProduct = "Patek Philippe 6300/400G", CostProduct = 9000000, imagePathProduct = "imgs/PatekPhilippeBlackWhite.png" });

products.Add(new Product { IdProduct = 7, NameProduct = "Apple watch series 10", CostProduct = 450, imagePathProduct = "imgs/appleseries10.png" });

products.Add(new Product { IdProduct = 8, NameProduct = "Jacob&Co Palatial Classic", CostProduct = 16500, imagePathProduct = "imgs/Jacob&Co.png" });

}

public string successValid = "";

public string errorValid = "";

public static List<OrderList> orders = new List<OrderList>();

public void OnPost()

{

OnGet();

if (!ModelState.IsValid)

{

errorValid = "Некоторые данные были введены неправильно!";

return;

}

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

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

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

;

orders.Add(new OrderList { orderListId = orders.Count + 1, nameCustomer = order.NameOrder, secondNameCustomer = order.SecondNameOrder,

mailCustomer = order.EmailOrder, addressCustomer = order.AddressOrder, quantityCustomer = order.QuantityOrder});

ModelState.Clear();

}

}

}

Листинг Order.cs:

using System.ComponentModel.DataAnnotations;

namespace task2.Models

{

public class Order

{

[Required(ErrorMessage = "Имя обязательно!")]

[StringLength(50, MinimumLength = 2, ErrorMessage = "Не менее 2 и не более 50 символов")]

public string NameOrder { get; set; }

[Required(ErrorMessage = "Фамилия обязательна!")]

[StringLength(50, MinimumLength = 2, ErrorMessage = "Не менее 2 и не более 50 символов")]

public string SecondNameOrder { get; set; }

[Required(ErrorMessage = "Почтовый ящик обязателен!")]

[EmailAddress(ErrorMessage = "Некорректый адрес")]

public string EmailOrder { get; set; }

[Required(ErrorMessage = "Адрес обязателен!")]

public string AddressOrder { get; set; }

[Required(ErrorMessage = "Количество обязательно!")]

[Range(1, 10, ErrorMessage = "Недопустимое количество")]

public int QuantityOrder { get; set; }

}

public class OrderList

{

public int orderListId;

public string nameCustomer;

public string secondNameCustomer;

public string mailCustomer;

public string addressCustomer;

public int quantityCustomer;

}

}

Листинг Product.cs:

namespace task2.Models

{

public class Product

{

public int IdProduct;

public string NameProduct;

public int CostProduct;

public string imagePathProduct;

}

}

Листинг JavaScript.js:

// Смена цвета кнопки при наведении

const button = document.body.querySelectorAll(".buttonToChange");

for (let i = 0; i < button.length; i++) {

button[i].addEventListener("mouseover", function () {

button[i].classList.add("changebackground");

});

button[i].addEventListener("mouseout", function () {

button[i].classList.remove("changebackground");

});

}

// Модальное окно

const dialog = document.querySelector(".dialogMain");

const html = document.querySelector("html");

const openModels = document.querySelectorAll(".openModel");

const priceHeader = document.querySelector(".priceHeader");

let price;

for (let i = 0; i < openModels.length; i++) {

openModels[i].addEventListener("click", function () {

dialog.showModal();

html.classList.add("changehtml");

if (openModels[i].getAttribute("data-price") > 0) {

price = parseInt(openModels[i].getAttribute("data-price"));

}

priceHeader.textContent = `Цена: ${price.toLocaleString()}$`;

});

}

document.querySelector("#closeModel").onclick = function () {

dialog.close();

html.classList.remove("changehtml");

};

//Изменения цены

const formElement = document.querySelector(".orderForm");

const quantity = document.querySelector("#quantity");

const promo = document.querySelector("#promo");

const promoList = {

"10procentSaleWOW": 0.9,

"15yearShop": 0.85,

};

function updatePrice() {

let quantityVal = 1;

if (quantity.value > 1) {

quantityVal = quantity.value;

}

let discount = promo.value in promoList ? promoList[promo.value] : 1;

priceHeader.textContent = `Цена: ${createPrice(price, quantityVal, discount).toLocaleString()}$ (${quantityVal} шт, скидка ${Math.round((1 - discount) \* 100)}%)`;

}

function createPrice(priceFunc, quantityFunc, discountFunc) {

let finalPrice = priceFunc \* quantityFunc \* discountFunc;

return finalPrice;

}

quantity.addEventListener("input", updatePrice);

promo.addEventListener("input", updatePrice);

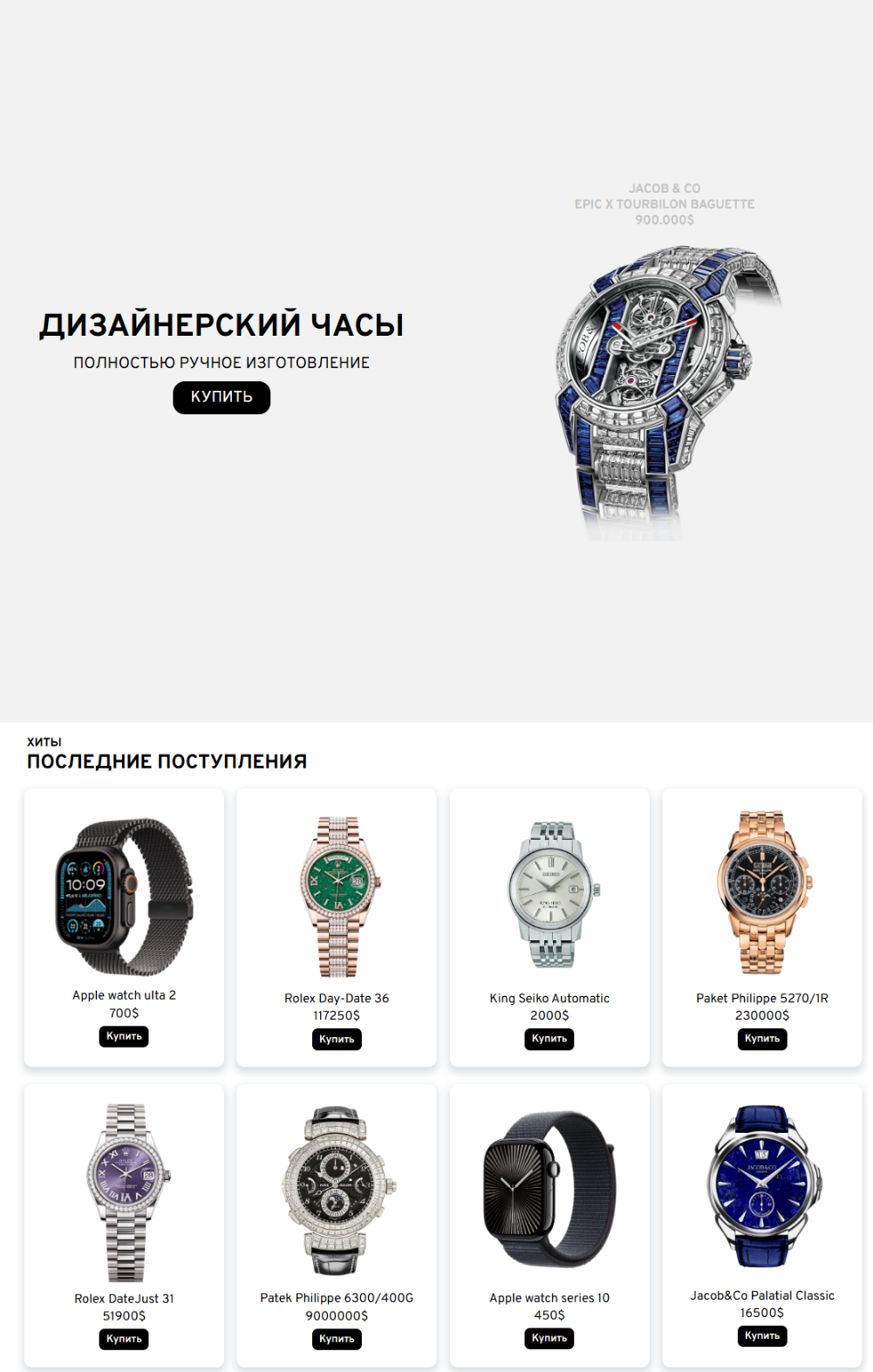


Рисунок 1. Разработанное веб-приложение.

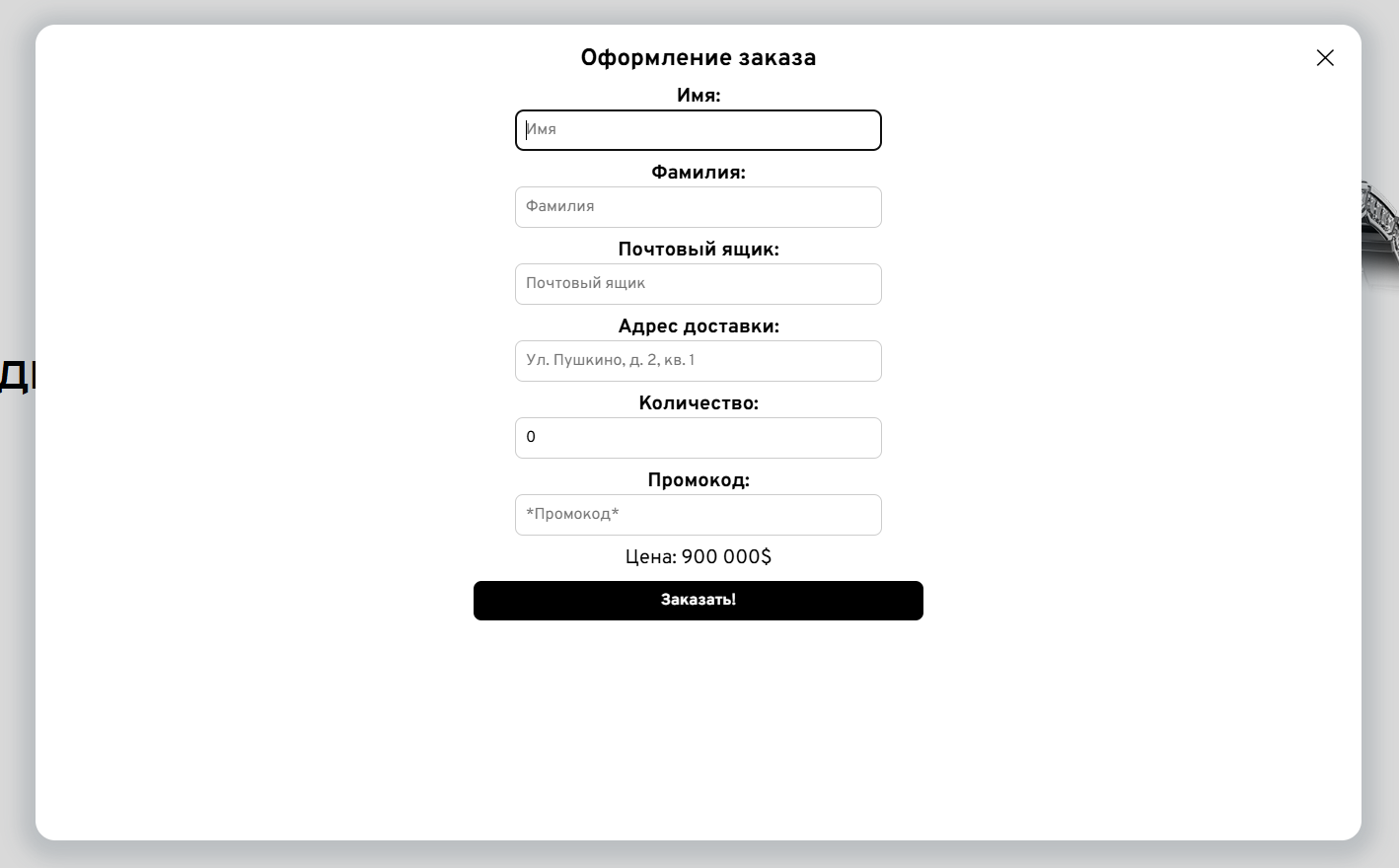


Рисунок 2. Форма оформления заказа.



Рисунок 3. Проверка на ввод данных не пройдена.

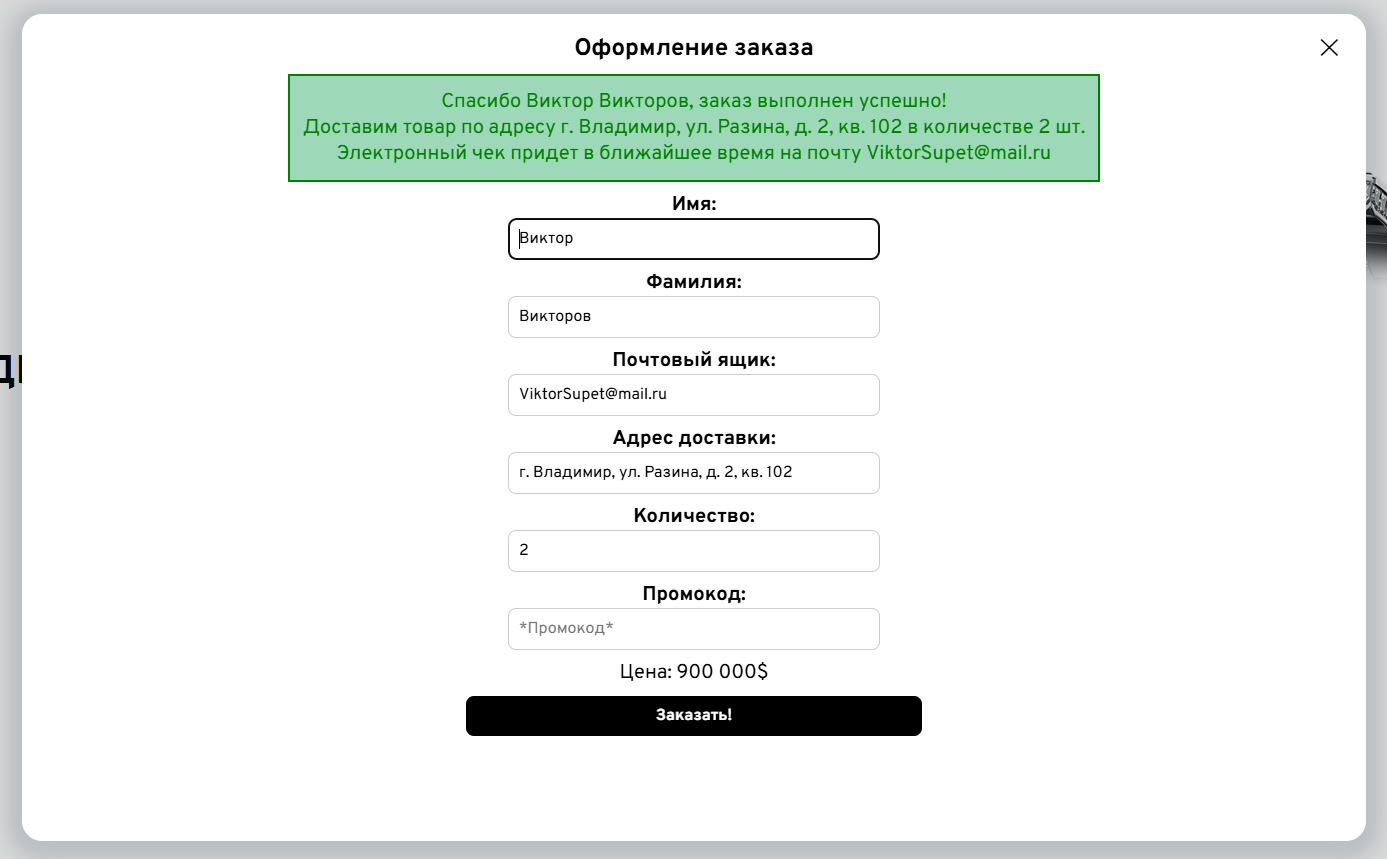


Рисунок 4. Проверка на ввод данных пройдена.

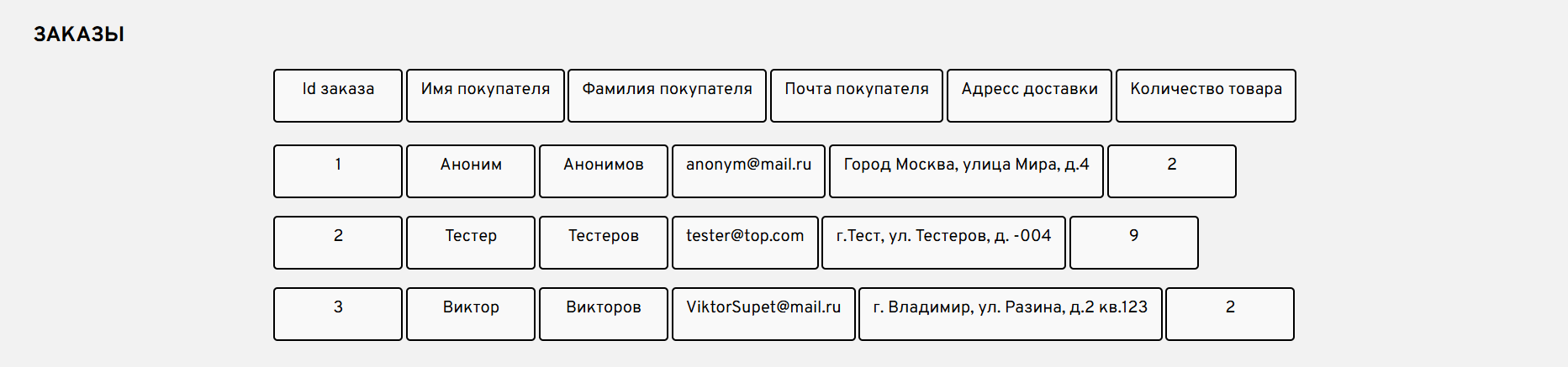


Рисунок 5. Вывод записанных данных.

ВЫВОД К РАБОТЕ:

Я составил свой пример, используя вместо компонента Repeater компоненты DataList или GridView, а также реализовал проверку данных на своем сайте.