**Лабораторная работа №3-4**

**Тема:** Взаимодействие с БД и основы JPA и EJB

**Задание**

Создать БД с использованием сервера MySQL или JavaDB (Derby). База данных должна несколько таблиц (не менее 2). Список таблиц и их полей определяет преподаватель.

Разработать WEB-приложение, которое позволяет выполнять с указанными таблицами операции CRUD (Create, Retrieve, Update, Delete).

Предусмотреть возможность выполнения запросов к созданной БД с отображением результатов на WEB-странице. Для ввода запросов использовать страницу JSP.

**Ход работы**

Далее будут представлен код исходных пакетов программы. Код представлен листингами 1-3.

Структура базы данных

База данных интернет-магазина "Люстры" содержит три таблицы: "Люстры", "Очередь заказов" и "Заказы".

Таблица "Люстры" предназначена для хранения информации о люстры, которые предполагается продать.

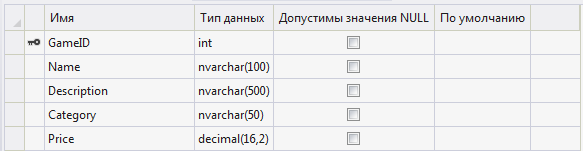


Рисунок 8 – Таблица "Товар"

Таблица "Очередь заказов" предназначена для хранения сведений о товарах, входящих в заказ.

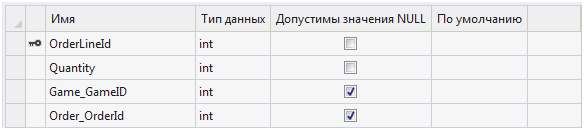


Рисунок 9 – Таблица "Очередь заказов"

Таблица "Заказы" предназначена для хранения информации о доставке заказа.

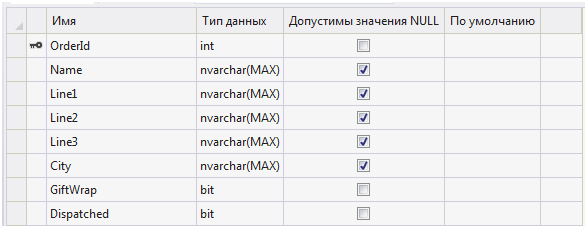


Рисунок 10 – Таблица "Заказы"

**Класс «Orders.cs”**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

using System.ComponentModel.DataAnnotations;

namespace GameStore.Models{

public class Order {

public int OrderId { get; set; }

[Required(ErrorMessage="Пожалуйста введите свое имя")]

public string Name { get; set; }

[Required(ErrorMessage="Вы должны указать хотя бы один адрес доставки")]

public string Line1 { get; set; }

public string Line2 { get; set; }

public string Line3 { get; set; }

[Required(ErrorMessage="Пожалуйста укажите город, куда нужно доставить заказ")]

public string City { get; set; }

public bool GiftWrap { get; set; }

public bool Dispatched { get; set; }

public virtual List<OrderLine> OrderLines { get; set; }

}

public class OrderLine {

public int OrderLineId { get; set; }

public Order Order { get; set; }

public Game Game { get; set; }

public int Quantity { get; set; }

}

}

**Класс «Cart.cs»**

using System.Collections.Generic;

using System.Linq;

namespace CandleStore.Models{

public class Cart{

private List<CartLine> lineCollection = new List<CartLine>();

public void AddItem(Game game, int quantity){

CartLine line = lineCollection

.Where(p => p.Game.GameId == game.GameId)

.FirstOrDefault();

if (line == null){

lineCollection.Add(new CartLine{

Game = game,

Quantity = quantity

});

}

else{

line.Quantity += quantity;

}

}

public void RemoveLine(Game game){

lineCollection.RemoveAll(l => l.Game.GameId == game.GameId);

}

public decimal ComputeTotalValue(){

return lineCollection.Sum(e => e.Game.Price \* e.Quantity);

}

public void Clear(){

lineCollection.Clear();

}

public IEnumerable<CartLine> Lines {

get { return lineCollection; }

}

}

public class CartLine{

public Game Game { get; set; }

public int Quantity { get; set; }

}

}

**Класс «Сhandel.cs»**

using System;

namespace GameStore.Models{

public class Game{

public int GameId { get; set; }

public string Name { get; set; }

public string Description { get; set; }

public string Category { get; set; }

public decimal Price { get; set; }

}

}

**Класс «VInput.cs»**

using System;

using System.ComponentModel.DataAnnotations;

using System.Reflection;

using System.Web.UI;

using System.Web.UI.WebControls;

namespace GameStore.Controls{

public class VInput : WebControl{

private string nspace = "GameStore.Models";

private string model = "Order";

public string Namespace{

get { return nspace; }

set { nspace = value; }

}

public string Model{

get { return model; }

set { model = value; }

}

public string Property { get; set; }

protected override void RenderContents(HtmlTextWriter output){

output.AddAttribute(HtmlTextWriterAttribute.Id, Property.ToLower());

output.AddAttribute(HtmlTextWriterAttribute.Name, Property.ToLower());

Type modelType = Type.GetType(string.Format("{0}.{1}", Namespace, Model));

PropertyInfo propInfo = modelType.GetProperty(Property);

var attr = propInfo.GetCustomAttribute<RequiredAttribute>(false);

if (attr != null)

{

output.AddAttribute("data-val", "true");

output.AddAttribute("data-val-required", attr.ErrorMessage);

}

output.RenderBeginTag("input");

output.RenderEndTag();

}

public override void RenderBeginTag(HtmlTextWriter writer){

}

public override void RenderEndTag(HtmlTextWriter writer){

}

}

}

**Результат**

Результаты работы программы представлены рисунками 1-5.

****

Рисунок 1 – «Бра»



Рисунок 2 – «Светильники»



Рисунок 3 – «Люстры»

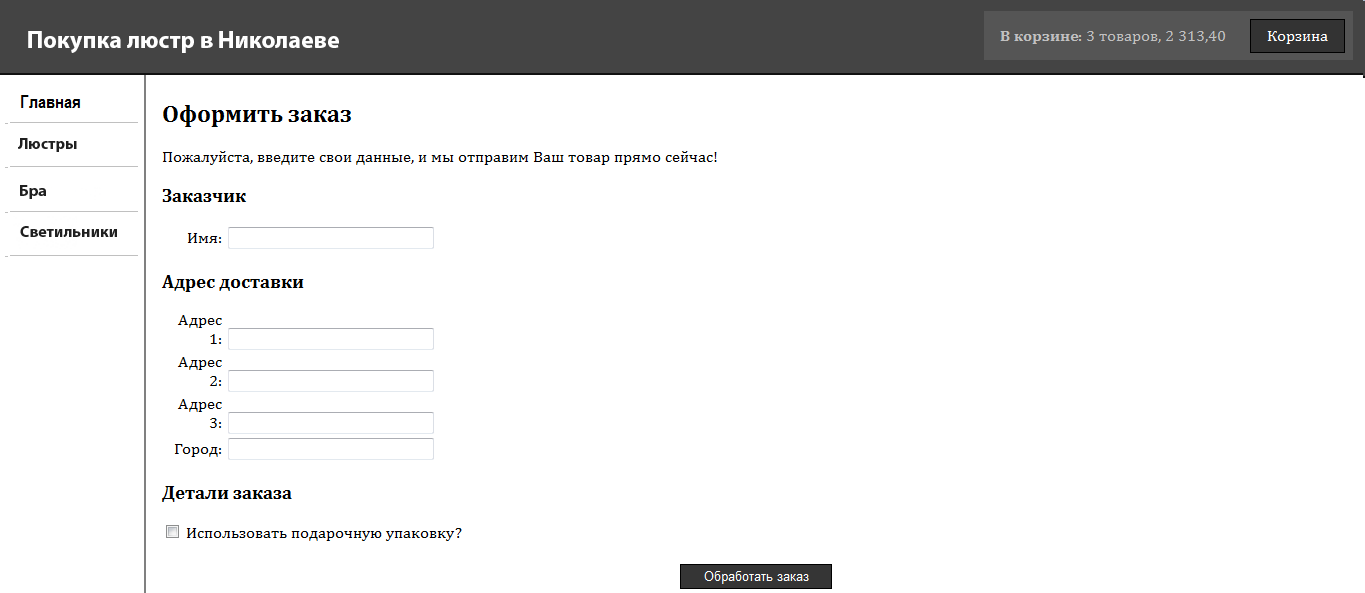


Рисунок 4 – «Заказ»

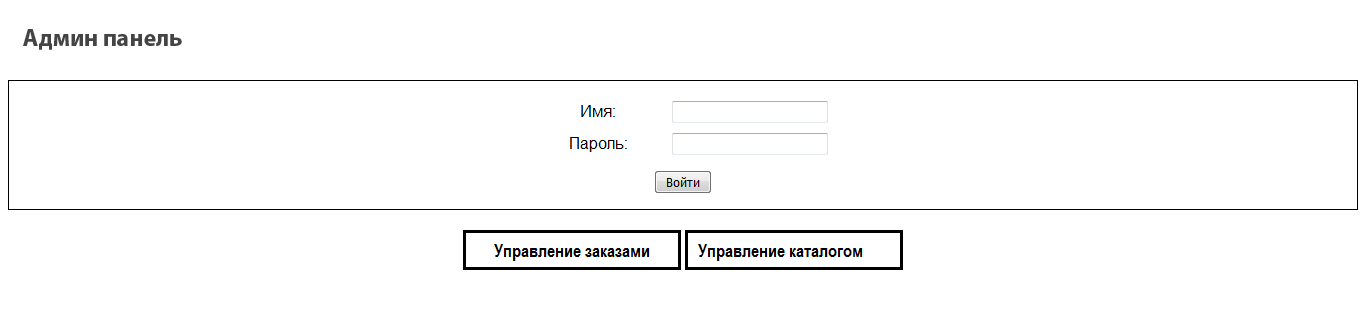


Рисунок 5 – «Админ панель»

**Вывод**: При выполнении данной лабораторной работы я создал базу данных и подключил ее к сайту.