基于MVC架构的电子商城后台管理设计

1. 引言

随着Internet的高速发展，网络与人们的生活息息相关，逐渐改变着人们工作和生活的方式，网上购物也成为了一种流行的生活方式。电子商务[1]正在迅猛发展，不仅改变着传统的社会生产方式，而且对经济结构的调整产生了极为深刻的影响，成为经济新的增长点。

电子商务[1]是指在互联网(Internet)、企业内部网(Intranet)和增值网(VAN，Value Added Network)上以电子交易方式进行交易活动和相关服务活动，是传统商业活动各环节的电子化、网络化。电子商务包括电子货币交换、供应链管理、电子交易市场、网络营销、在线事务处理、电子数据交换(EDI)、存货管理和自动数据收集系统。电子商务利用到的信息技术包括：互联网、外联网、电子邮件、数据库、电子目录和移动电话等，买卖双方不谋面地进行各种商贸活动，实现消费者的网上购物、商户之间的网上交易和在线电子支付以及各种商务活动、交易活动、金融活动和相关的综合服务活动的一种新型的商业运营模式。

随着中国电子商务的不断高速发展，其影响力逐步增大，无论是电子商务的所面临的问题还是电子商务的优势特点，都开始受政府的广泛关注，健康发展电子商务，拉动中国经济发展已经被提到政府的工作日程之中。由于电子商务可以压缩企业供应链长度，提高生产、库存、运输与销售环节的效率，从中央部委到地方政府，一系列电子商务发展规划紧锣密鼓的开展起来。

对于网上购物这样一个发展迅速的消费模式，又存在一些制约其发展的问题，从购物网站中不难发现这些问题。对于一个好的购物网站往往有着这样几点要求：

第一，结构清晰。要求购物平台结构清晰简洁，用户易于操作，不会因为页面设计的过于繁琐导致用户操作困难。

第二，方便查找，用户需要通过某种商品名称查找商品。

第三，便于管理，设置管理员对商品信息进行管理，且管理员能够很容易的对商品进行管理。

第四，消费安全，要求网站具有比较高的安全防护措施，能够比较好的保护用户的隐私与财产。

这些问题往往是一个购物网站发展的关键因素。

为解决以上几点问题首先需要为系统设立三类用户，一种普通用户，一种入驻商家，另外还有管理员用户。其次为满足结构清晰的问题，需要对页面布局花更多时间设计，使得页面结构体系分块明显。最后为方便用户查找商品在系统中设置搜索功能。只有解决了以上几点，才能够吸引更多的顾客使用便民超市电子商务系统中。

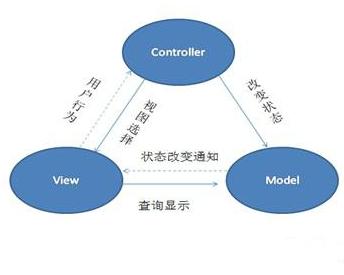
1. B/S结构与设MVC模式
2. B/S结构

B/S结构（Browser/Server，浏览器/服务器模式），是WEB兴起后的一种网络结构模式，WEB浏览器是客户端最主要的应用软件。客户机上只要安装一个浏览器（Browser），如Netscape Navigator或Internet Explorer，服务器安装Oracle、Sybase、Informix或 SQL Server等数据库。浏览器通过Web Server 同数据库进行数据交互。B/S最大的优点就是可以在任何地方进行操作而不用安装任何专门的软件，只要有一台能上网的电脑就能使用，客户端零安装、零维护；系统的扩展非常容易。

B/S结构实现了跨平台、跨地域，它打破了地域限制，真正使信息得以高度共享与快速传播，而这正是电子商务得以迅猛发展的基础。

1. MVC模式

MVC（Model View Controller）是一种架构设计模式，该模式主要应用于图形化用户界面(GUI)应用程序。MVC由Model（模型）、View（视图）及Controller（控制器）三部分组成。MVC是一种软件设计典范，用于组织代码用一种业务逻辑和数据显示分离的方法，这个方法的假设前提是如果业务逻辑被聚集到一个部件里面，而且界面和用户围绕数据的交互能被改进和个性化定制而不需要重新编写业务逻辑MVC被独特的发展起来用于映射传统的输入、处理和输出功能在一个逻辑的图形化用户界面的结构中。



**视图**

视图是用户看到并与之交互的界面。对老式的Web应用程序来说，视图就是由HTML元素组成的界面，在新式的Web应用程序中，HTML依旧在视图中扮演着重要的角色，但一些新的技术已层出不穷，它们包括Adobe Flash和像XHTML，XML/XSL，WML等一些标识语言和Web services.

MVC好处是它能为应用程序处理很多不同的视图。在视图中其实没有真正的处理发生，不管这些数据是联机存储的还是一个雇员列表，作为视图来讲，它只是作为一种输出数据并允许用户操纵的方式。

**模型**

模型表示企业数据和业务规则。在MVC的三个部件中，模型拥有最多的处理任务。例如它可能用像EJBs和ColdFusion Components这样的构件对象来处理数据库，被模型返回的数据是中立的，就是说模型与数据格式无关，这样一个模型能为多个视图提供数据，由于应用于模型的代码只需写一次就可以被多个视图重用，所以减少了代码的重复性。

**控制器**

控制器接受用户的输入并调用模型和视图去完成用户的需求，所以当单击Web页面中的超链接和发送HTML表单时，控制器本身不输出任何东西和做任何处理。它只是接收请求并决定调用哪个模型构件去处理请求，然后再确定用哪个视图来显示返回的数据。

MVC模式的优点。

1. 耦合性低

视图层和业务层分离，这样就允许更改视图层代码而不用重新编译模型和控制器代码，同样，一个应用的业务流程或者业务规则的改变只需要改动MVC的模型层即可。因为模型与控制器和视图相分离，所以很容易改变应用程序的数据层和业务规则。

模型是自包含的，并且与控制器和视图相分离，所以很容易改变应用程序的数据层和业务规则。如果把数据库从MySQL移植到Oracle，或者改变基于RDBMS数据源到LDAP，只需改变模型即可。一旦正确的实现了模型，不管数据来自数据库或是LDAP服务器，视图将会正确的显示它们。由于运用MVC的应用程序的三个部件是相互独立，改变其中一个不会影响其它两个，所以依据这种设计思想能构造良好的松耦合的构件。

1. 重用性高

MVC框架随着技术的不断进步，需要用越来越多的方式来访问应用程序。MVC模式允许使用各种不同样式的视图来访问同一个服务器端的代码，因为多个视图能共享一个模型，它包括任何WEB（HTTP）浏览器或者无线浏览器（wap），比如，用户可以通过电脑也可通过手机来订购某样产品，虽然订购的方式不一样，但处理订购产品的方式是一样的。由于模型返回的数据没有进行格式化，所以同样的构件能被不同的界面使用。例如，很多数据可能用HTML来表示，但是也有可能用WAP来表示，而这些表示所需要的命令是改变视图层的实现方式，而控制层和模型层无需做任何改变。由于已经将数据和业务规则从表示层分开，所以可以最大化的重用代码了。模型也有状态管理和数据持久性处理的功能，例如，基于会话的购物车和电子商务过程也能被Flash网站或者无线联网的应用程序所重用。[3]

1. 生命周期成本低

MVC使开发和维护用户接口的技术含量降低。

1. 部署快

使用MVC模式使开发时间得到相当大的缩减，它使程序员集中精力于业务逻辑，界面程序员集中精力于表现形式上。

1. 可维护性高

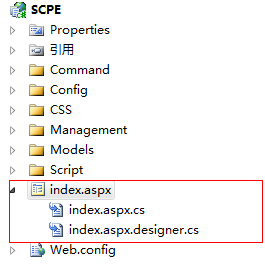
分离视图层和业务逻辑层也使得WEB应用更易于维护和修改。

1. 有利软件工程化管理

由于不同的层各司其职，每一层不同的应用具有某些相同的特征，有利于通过工程化、工具化管理程序代码。控制器也提供了一个好处，就是可以使用控制器来联接不同的模型和视图去完成用户的需求，这样控制器可以为构造应用程序提供强有力的手段。给定一些可重用的模型和视图，控制器可以根据用户的需求选择模型进行处理，然后选择视图将处理结果显示给用户。

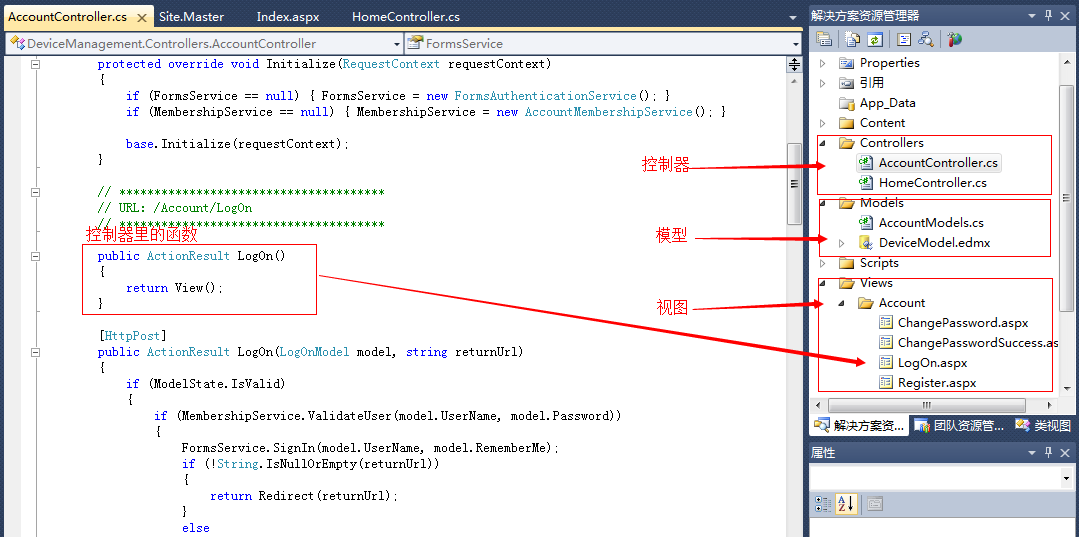
以上只是书上给的说法，对于它的真实性我们可以简单验证。

对于一般的webform程序（仅以.net为例）

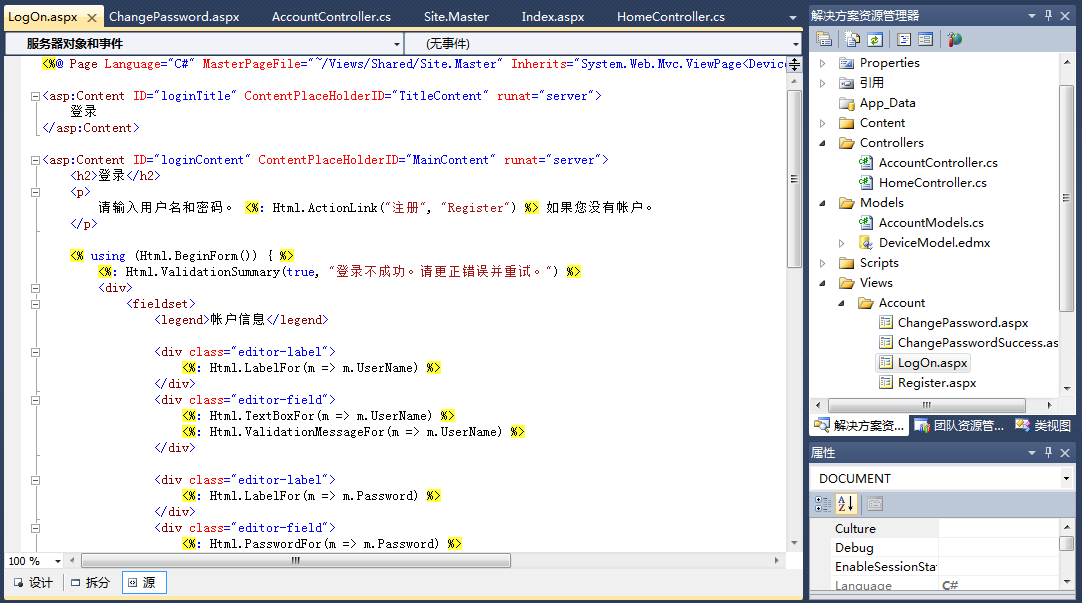


一个网页包含3个文件，一个是包含Html的.aspx文件，一个是逻辑处理的CS类文件，还有一个是页面设计的类文件，其中主要包含控件的声明。这三个文件相互关联更改其中一个，其他都必须更改，虽然vs有强大的设计工具可以大大减少我们程序员的工作，但还是不够灵活。

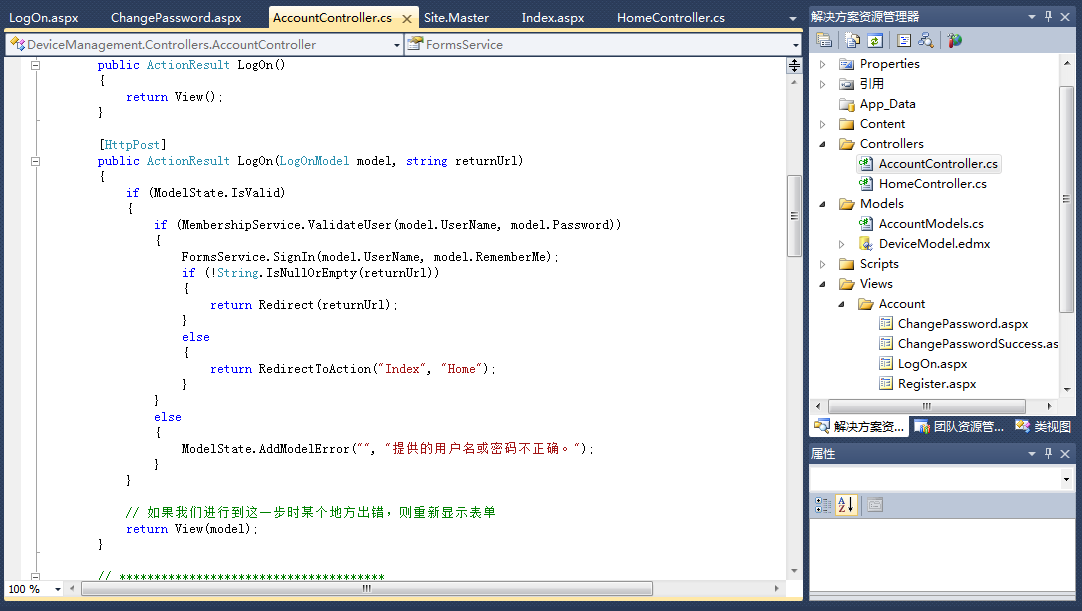
而MVC架构的web应用则灵活的多。如下图



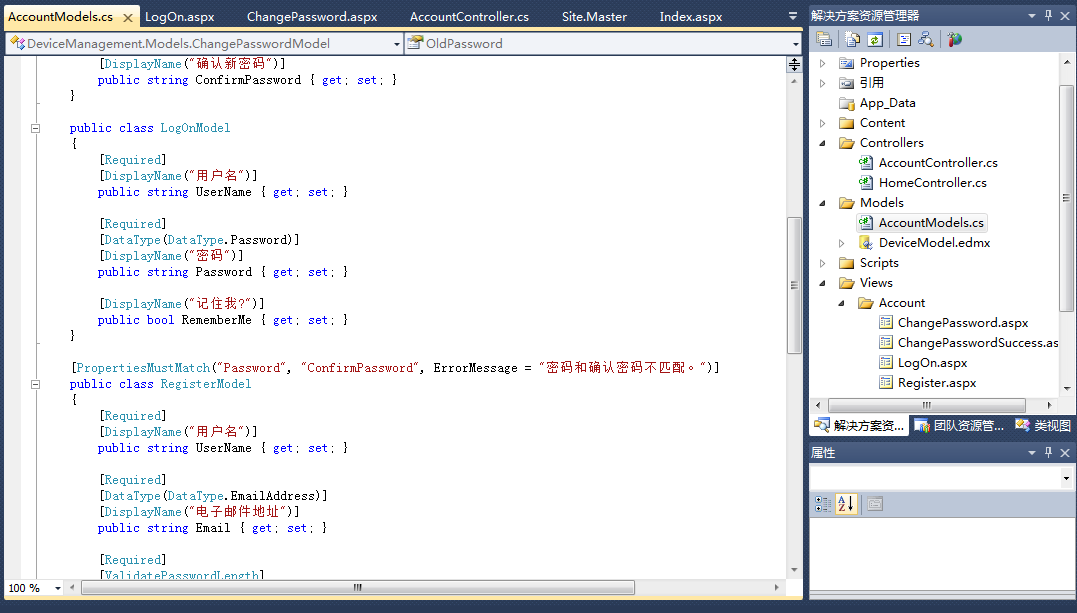
在mvc架构中，将程序分为了三部分，视图、模型、控制器。视图中包含的是网页HTML标签及脚本，控制器中包含业务规则，在控制器函数中处理过的数据通过函数名或指定传递到视图中，视图再显示。Model主要作用就是数据的载体。这种架构，将页面和代码分离，使得程序更易于分工和扩展。



此图是LoginOn的视图内容，注意，虽然此页里有C#代码和控件，却并不违反页面与代码分离的原则。这里的代码和控件只是用于显示传进此视图的数据，与控制器没有关联。



账户登录的业务



账户登录的模型模型

通过以上比较，可以很明显的发现mvc架构比webform更灵活，耦合性更低，当然更易于扩展。

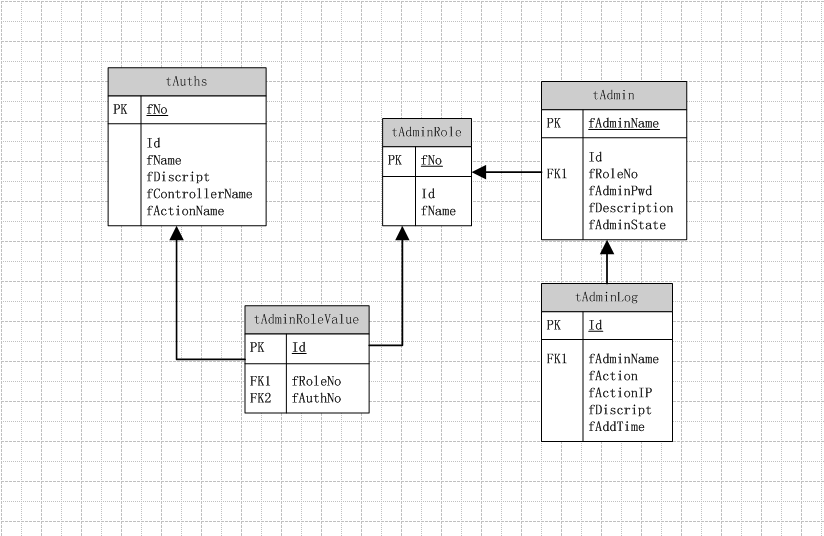
1. 功能系统设计

1、系统管理员管理和角色权限管理

系统管理员又可自定义分为：超级管理员、商品管理员、订单管理员，他们各自所拥有的权限不同，可自定义权限级别。超级管理员拥有最高权限，是商品管理员和订单管理员的上级，对商品管理员和订单管理员具有任免、评审业绩的权力，为保证超级管理员的有效管理，超级管理员的信息保密，将其设置为系统内唯一对象，其他管理员都无权管理超级管理员的信息，商品管理员主要职责为对便民超市商城官方销售的商品信息进行管理，主要职责为增删商品目录，管理商品只有此类管理员有此类权限。订单管理员负责对网上销售及售后服务。

系统管理员有不同的角色，不同的角色有不同的操作权限。

ER模型



数据字典

**表名：tAdmin（后台管理员）**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **序号** | **列名** | **数据类型** | **长度** | **小数位** | **标识** | **主键** | **外键** | **允许空** | **默认值** | **说明** |
| 1 | Id | int | 4 | 0 |  | 是 |  | 否 |  | 标识列 |
| 3 | fRoleNo | int | 4 | 0 |  |  |  | 否 |  | 角色编号 |
| 4 | fAdminName | nvarchar | 50 | 0 |  |  |  | 否 |  | 用户名（管理员） |
| 5 | fAdminPwd | nvarchar | 100 | 0 |  |  |  | 否 |  | 用户密码 |
| 6 | fDescription | nvarchar | 255 | 0 |  |  |  | 否 | N'无' | 描述 |
| 7 | fAdminState | int | 4 | 0 |  |  |  | 否 |  | 用户状态（是否可用） |
|  |  |  |  |  |  |  |  |  |  |  |

**表名：tAdminRole（管理员角色）**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **序号** | **列名** | **数据类型** | **长度** | **小数位** | **标识** | **主键** | **外键** | **允许空** | **默认值** | **说明** |
| 1 | Id | int | 4 | 0 |  | 是 |  | 否 |  | 标识列 |
| 2 | fNo | int | 4 | 0 |  |  |  | 否 |  | 角色编号 |
| 3 | fName | nvarchar | 50 | 0 |  |  |  | 否 |  | 角色名称 |

**表名：tAdminRoleValue（管理员角色权限）**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **序号** | **列名** | **数据类型** | **长度** | **小数位** | **标识** | **主键** | **外键** | **允许空** | **默认值** | **说明** |
| 1 | Id | int | 4 | 0 |  | 是 |  | 否 |  | 标识列 |
| 2 | fRoleNo | int | 4 | 0 |  |  |  | 否 |  | 角色编号 |
| 3 | fAuthNo | int | 4 | 0 |  |  |  | 否 |  | 权限编号 |

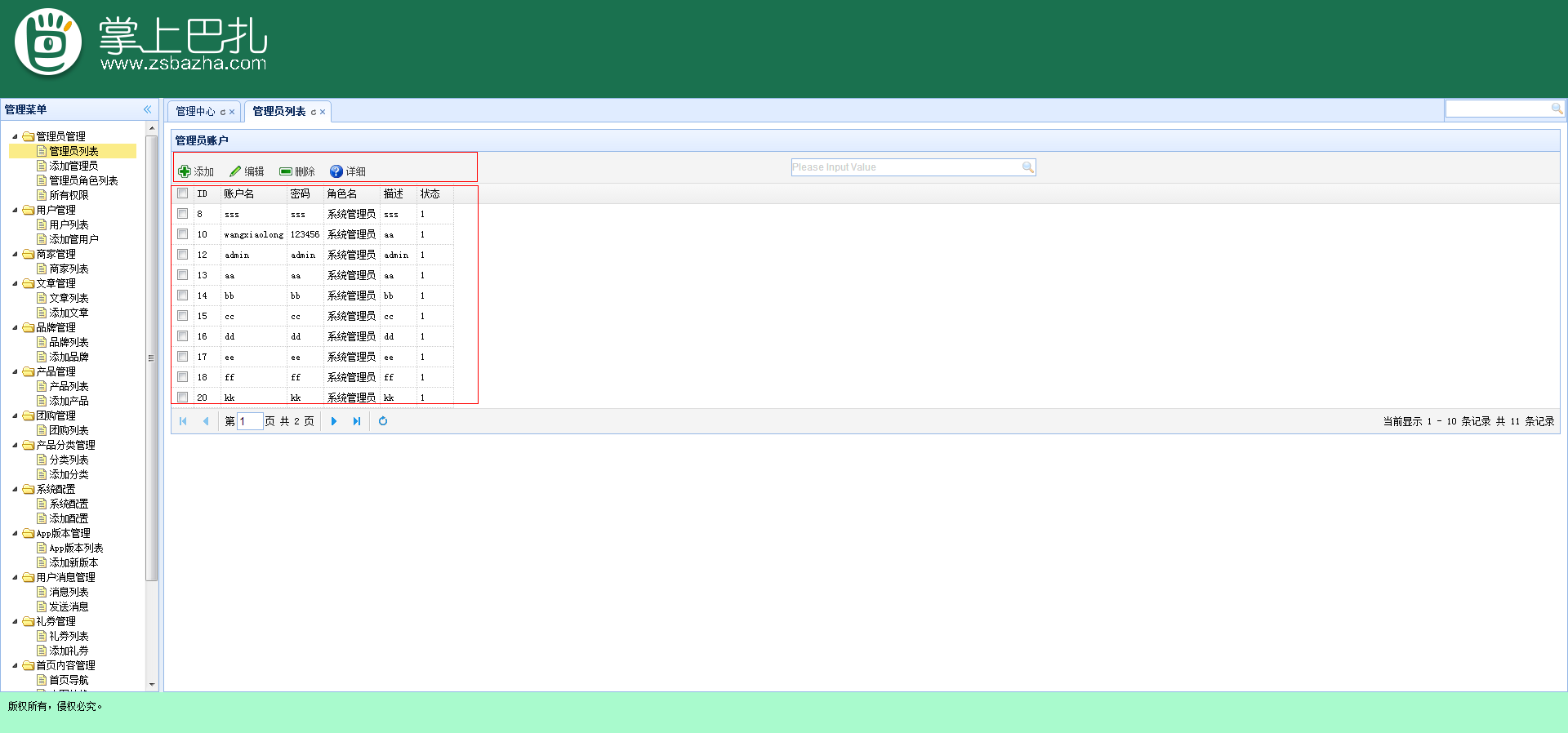
**表名：tAuths（角色权限）**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **序号** | **列名** | **数据类型** | **长度** | **小数位** | **标识** | **主键** | **外键** | **允许空** | **默认值** | **说明** |
| 1 | Id | int | 4 | 0 |  | 是 |  | 否 |  | 标识列 |
|  | fNo | Int |  |  |  |  |  |  |  | 编号 |
| 2 | fName | nvarchar | 50 | 0 |  |  |  | 否 |  | 操作名称 |
| 3 | fDiscript | nvarchar | 0 | 0 |  |  |  | 否 |  | 操作描述 |
| 5 | fControllerName | nvarchar | 50 | 0 |  |  |  | 否 |  | 控制器名称 |
| 6 | fActionName | nvarchar | 50 | 0 |  |  |  | 否 |  | 动作名称 |

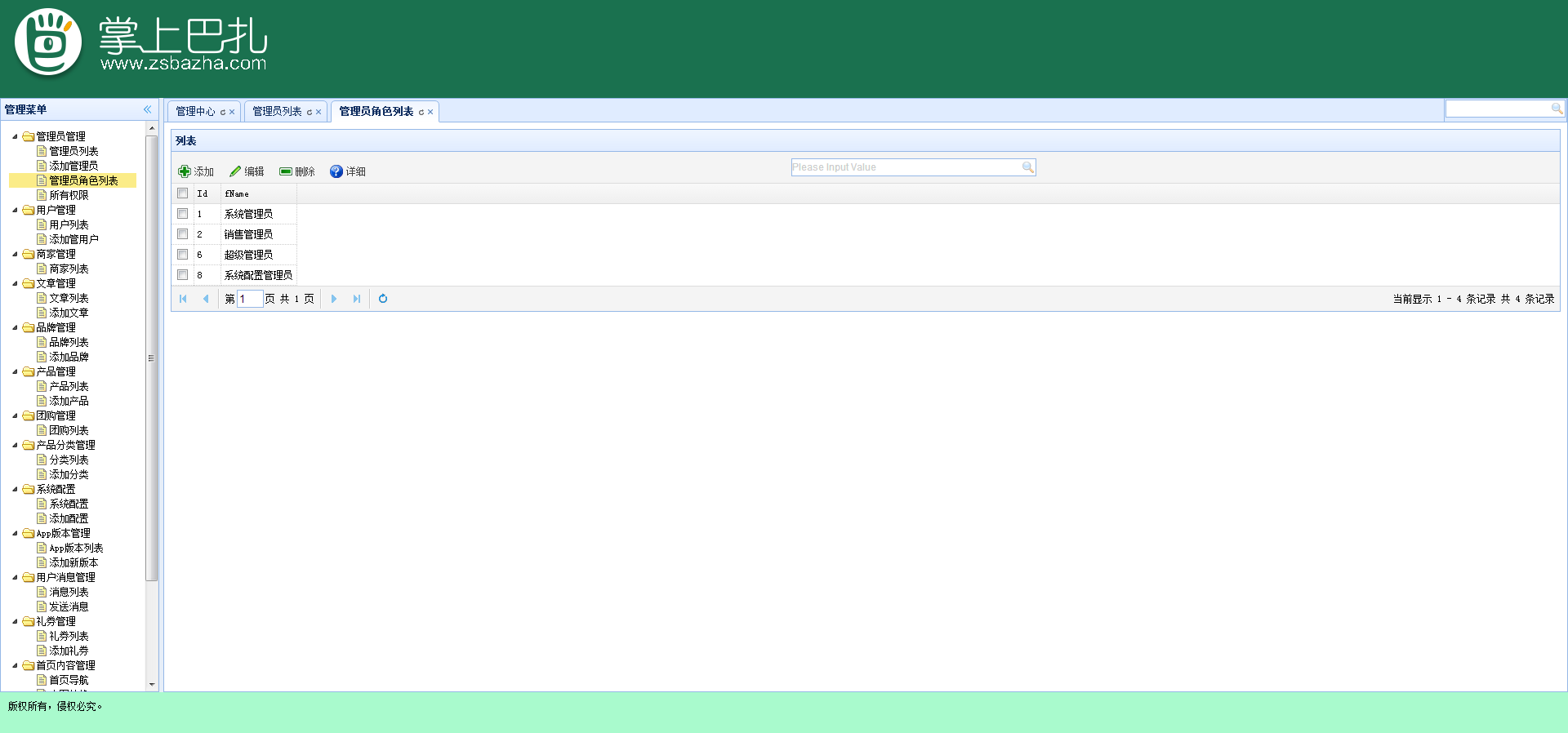
关键代码

|  |
| --- |
| using System;  using System.Collections.Generic;  using System.Linq;  using System.Web;  using System.Web.Mvc;  using SuperMarket.Model;  using System.Web.Script.Serialization;  namespace SuperMarket.Admin.Controllers  {  /\*  \*  \* 管¨¹理¤¨ª员¡À管¨¹理¤¨ª by wxl 2014-3-19  \*  \*  \*\*/  //[Common.AuthFilter]  public class AdminController : Controller  {  JavaScriptSerializer js = new JavaScriptSerializer();  SuperMarket.BLL.tAdmin bll = new SuperMarket.BLL.tAdmin();    // GET: /Admin/  public ActionResult Index()  {  Model.tAdmin m=new tAdmin();  Type t = m.GetType();  string n = t.Name;    return View(bll.GetModelList(""));  }  public ActionResult GetList(int rows,int page,string where,string order)  {  int pagesize = 10;  List<Model.tAdmin> list = bll.GetModelList("", "Id", pagesize, page, out rows);//new { listdata = list, pagedata = new { page=1,pagesize=20,recordcount=2000} }  return Json(new { rows= list, total = rows }, JsonRequestBehavior.AllowGet);  }  //  // GET: /Admin/Details/5  public ActionResult Details(int id)  {  Model.tAdmin m = bll.GetModel(id);  if (m == null) return View("Error", new Exception("未¡ä找¨°到Ì?该?账?号?"));  return View(m);  }  //  // GET: /Admin/Create  public ActionResult Create()  {  return View();  }  //  // POST: /Admin/Create  [HttpPost]  public ActionResult Create(FormCollection collection)  {  try  {  // TODO: Add insert logic here  Model.tAdmin model = new tAdmin();  model.fAdminName = collection["fAdminName"];  model.fAdminPwd = collection["fAdminPwd"];  model.fRoleName = collection["fRoleName"];  model.fDescription = collection["fDescription"];  model.fCANumber = "";  model.fAdminState =int.Parse(collection["fAdminState"]);  bll.Add(model);  return RedirectToAction("Index");  }  catch(Exception err)  {  return View("Error",err);  }  }    //  // GET: /Admin/Edit/5    public ActionResult Edit(int id)  {  Model.tAdmin m = bll.GetModel(id);  if (m == null) return View("Error", new Exception("未¡ä找¨°到Ì?该?账?号?"));  if (m.fSystem == 1) return View("error", new Exception("该?账?号?为a系¦Ì统ª3账?号?，ê?不?可¨¦修T改?"));  return View(m);  }  //  // POST: /Admin/Edit/5  [HttpPost]  public ActionResult Edit(int id, FormCollection collection)  {  try  {  // TODO: Add update logic here  Model.tAdmin model = bll.GetModel(id);  if (model == null) return View("Error", new Exception("未¡ä找¨°到Ì?该?账?号?"));    model.fAdminPwd = collection["fAdminPwd"];  model.fRoleName = collection["fRoleName"];  model.fDescription = collection["fDescription"];  model.fAdminState = int.Parse(collection["fAdminState"]);  model.fCANumber = "";  bll.Update(model);  return RedirectToAction("Index");  }  catch(Exception err)  {  return View("error", err);  }  }  //  // GET: /Admin/Delete/5    public ActionResult Delete(int id)  {  Model.tAdmin m = bll.GetModel(id);  if (m == null) return View("Error", new Exception("未¡ä找¨°到Ì?该?账?号?"));  return View(m);  }  //  // POST: /Admin/Delete/5  [HttpPost]  public ActionResult Delete(int id, FormCollection collection)  {  try  {  Model.tAdmin m = bll.GetModel(id);  if (m == null) return View("error", new Exception("未¡ä找¨°到Ì?该?账?号?"));  if (m.fSystem == 1) return View("error", new Exception("该?账?号?为a系¦Ì统ª3账?号?，ê?不?可¨¦删¦?除y"));    bll.Delete(id);  return RedirectToAction("Index");  }  catch (Exception err)  {  return View("error", err);  }  }  }  } |

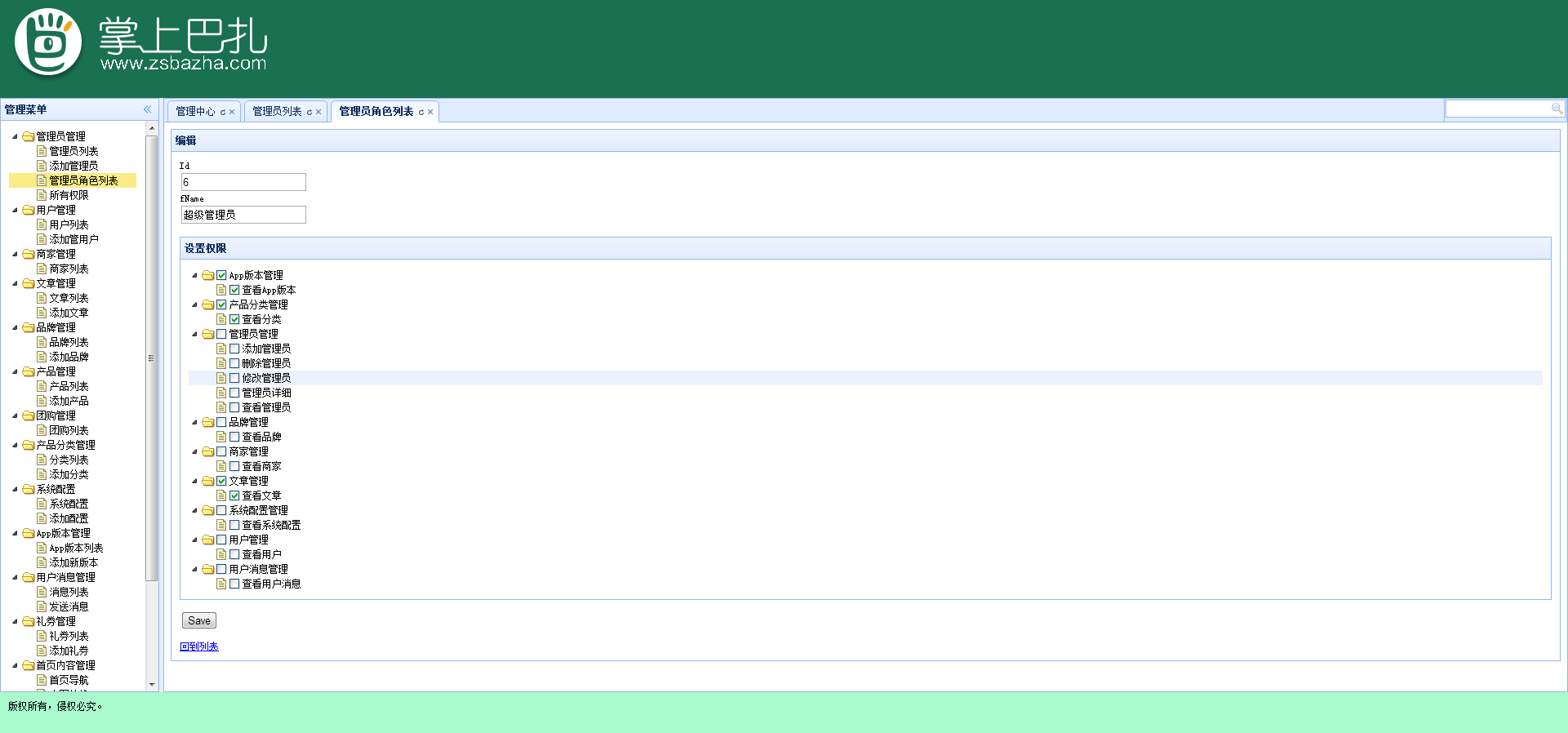
程序截图



管理员管理截图



管理员角色管理截图

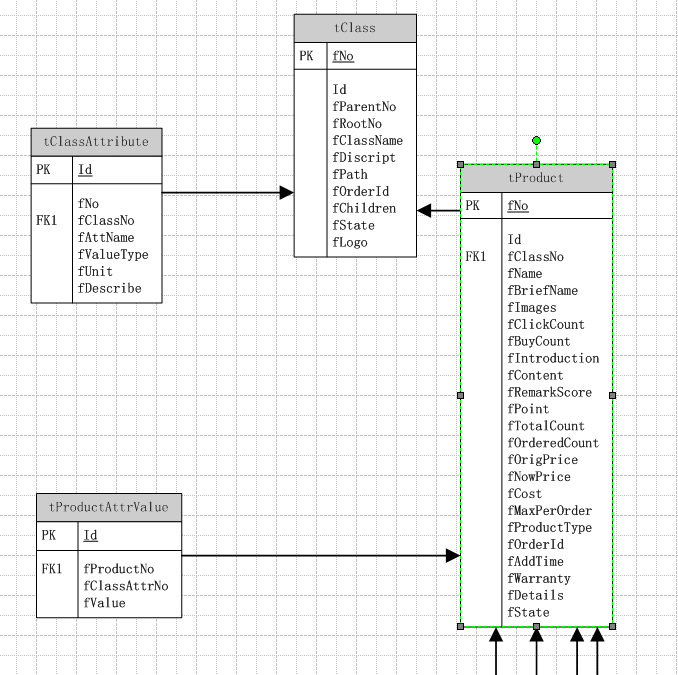


角色权限设置截图

1. 商品及商品类别管理

商品类别基础信息管理主要是对商品名称、商品生产日期、商品出厂日期、商品生产地、商品责任人、商品货号、商品责任单位、商品型号、商品库存、商品单价、商品图片、商品条形码、备注等要素信息的维护。

ER模型



数据字典

**表名：tClass（商品分类）**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **序号** | **列名** | **数据类型** | **长度** | **小数位** | **标识** | **主键** | **外键** | **允许空** | **默认值** | **说明** |
|  | Id | int |  |  |  |  |  |  |  | 标识列 |
| 1 | fNo | int | 4 | 0 |  | 是 |  | 否 |  | 编号 |
| 2 | fParentNo | int | 4 | 0 |  |  |  | 否 |  | 父类编号 |
| 3 | fRootNo | int | 4 | 0 |  |  |  | 否 |  | 根类编号 |
| 4 | fClassName | nvarchar | 50 | 0 |  |  |  | 否 |  | 分类名称 |
| 5 | fDiscript | nvarchar | 50 | 0 |  |  |  | 否 |  | 描述 |
| 6 | fPath | nvarchar | 0 | 0 |  |  |  | 否 |  | 路径（1,5,6）方便查询 |
| 8 | fOrderId | int | 4 | 0 |  |  |  | 否 |  | 排序号 |
| 9 | fChildren | int | 4 | 0 |  |  |  | 否 |  | 子类数 |
| 11 | fState | int | 4 | 0 |  |  |  | 否 |  | 状态 |
| 12 | fLogo | nvarchar | 255 | 0 |  |  |  | 否 |  | 图标 |
|  | fBusinessNo | int | 4 | 0 |  |  |  | 否 |  | 商家编号 |

**表名：tClassAttribute（分类属性）**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **序号** | **列名** | **数据类型** | **长度** | **小数位** | **标识** | **主键** | **外键** | **允许空** | **默认值** | **说明** |
| 1 | Id | int | 4 | 0 |  | 是 |  | 否 |  | 标识列 |
| 2 | fNo | int | 4 | 0 |  |  |  | 否 |  | 编号 |
| 3 | fClassNo | int | 4 | 0 |  |  |  | 否 |  | 所属分类编号 |
| 4 | fAttName | nvarchar | 50 | 0 |  |  |  | 否 |  | 属性名称，如“颜色”、“尺寸”。 |
| 5 | fValueType | nvarchar | 50 | 0 |  |  |  | 否 | N'EString' | 值类型，字符串类型，枚举。 |
| 6 | fUnit | nvarchar | 50 | 0 |  |  |  | 否 |  | 单位。 |
| 7 | fDescribe | nvarchar | 50 | 0 |  |  |  | 否 |  | 描述。 |

**表名：tProduct（产品）**

商家担保，承诺

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **序号** | **列名** | **数据类型** | **长度** | **小数位** | **标识** | **主键** | **外键** | **允许空** | **默认值** | **说明** |
| 1 | Id | int | 4 | 0 | 是 | 是 |  | 否 |  | 标识列 |
| 2 | fNo | int | 4 | 0 |  |  |  | 否 |  | 编号 |
| 3 | fClassNo | int | 4 | 0 |  |  |  | 否 | -1 | 分类编号 |
| 4 | fName | nvarchar | 200 | 0 |  |  |  | 否 | N'未命名' | 商品名称 |
| 5 | fBriefName | nvarchar | 100 | 0 |  |  |  | 否 | N'未命名' | 简单名称 |
| 6 | fImages | nvarchar | 0 | 0 |  |  |  | 否 | 0 | 商品图片 |
| 7 | fClickCount | int | 4 | 0 |  |  |  | 否 | 0 | 点击数 |
| 8 | fBuyCount | int | 4 | 0 |  |  |  | 否 | 0 | 购买数 |
| 9 | fIntroduction | nvarchar | 0 | 0 |  |  |  | 否 | ' ' | 商品介绍 |
| 10 | fContent | nvarchar | 0 | 0 |  |  |  | 否 | ' ' | 展示内容 |
| 11 | fRemarkScore | int | 4 | 0 |  |  |  | 否 | 0 | 评分 |
| 12 | fPoint | decimal | 9 | 0 |  |  |  | 否 | 0 | 积分 |
| 13 | fTotalCount | int | 4 | 0 |  |  |  | 否 | 0 | 库存总数 |
| 14 | fOrderedCount | int | 4 | 0 |  |  |  | 否 | 0 | 已订购数 |
| 15 | fOrigPrice | decimal | 9 | 2 |  |  |  | 否 | 99999999 | 原价 |
| 16 | fNowPrice | decimal | 9 | 2 |  |  |  | 否 | 99999999 | 现价 |
| 17 | fCost | decimal | 9 | 2 |  |  |  | 否 | 99999999 | 成本 |
| 18 | fMaxPerOrder | int | 4 | 0 |  |  |  | 否 | 0 | 最大订购数 |
| 19 | fProductType | int | 4 | 0 |  |  |  | 否 | 0 | ~~0:其他商品，1:水果，2:认养~~ |
| 20 | fOrderId | int | 4 | 0 |  |  |  | 否 | -1 | 排序号 |
| 21 | fAddTime | datetime | 8 | 3 |  |  |  | 否 | '10-01-1 0:0:0' | 上架时间 |
| 22 | fWarranty | nvarchar | 0 | 0 |  |  |  | 否 |  | 商家担保，承诺 |
| 23 | fDetails | nvarchar | 0 | 0 |  |  |  | 否 |  | 商品详情 |
| 24 | fState | int | 4 | 0 |  |  |  | 否 |  | 状态（是否上架、可用） |
| 25 | fBusinessNo | int | 4 | 0 |  |  |  | 否 |  | 商家编号 |

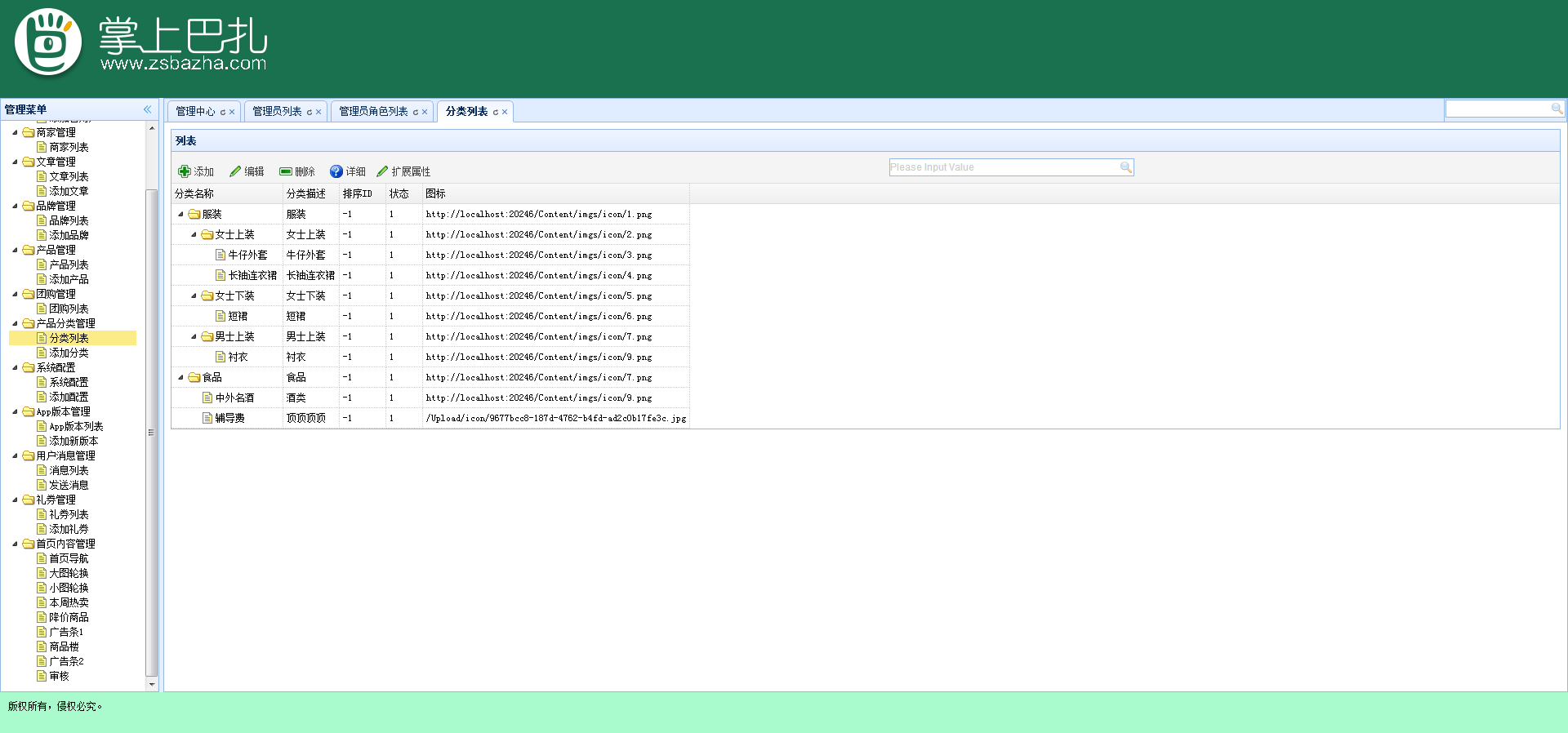
**表名：tProductAttrValue（产品属性值）**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **序号** | **列名** | **数据类型** | **长度** | **小数位** | **标识** | **主键** | **外键** | **允许空** | **默认值** | **说明** |
| 1 | Id | int | 4 | 0 |  | 是 |  | 否 |  | 标识列 |
| 2 | fProductNo | int | 4 | 0 |  |  |  | 否 |  | 商品编号 |
| 3 | fClassAttrNo | int | 4 | 0 |  |  |  | 否 |  | 分类编号 |
| 4 | fValue | nvarchar | 100 | 0 |  |  |  | 否 |  | 值 |

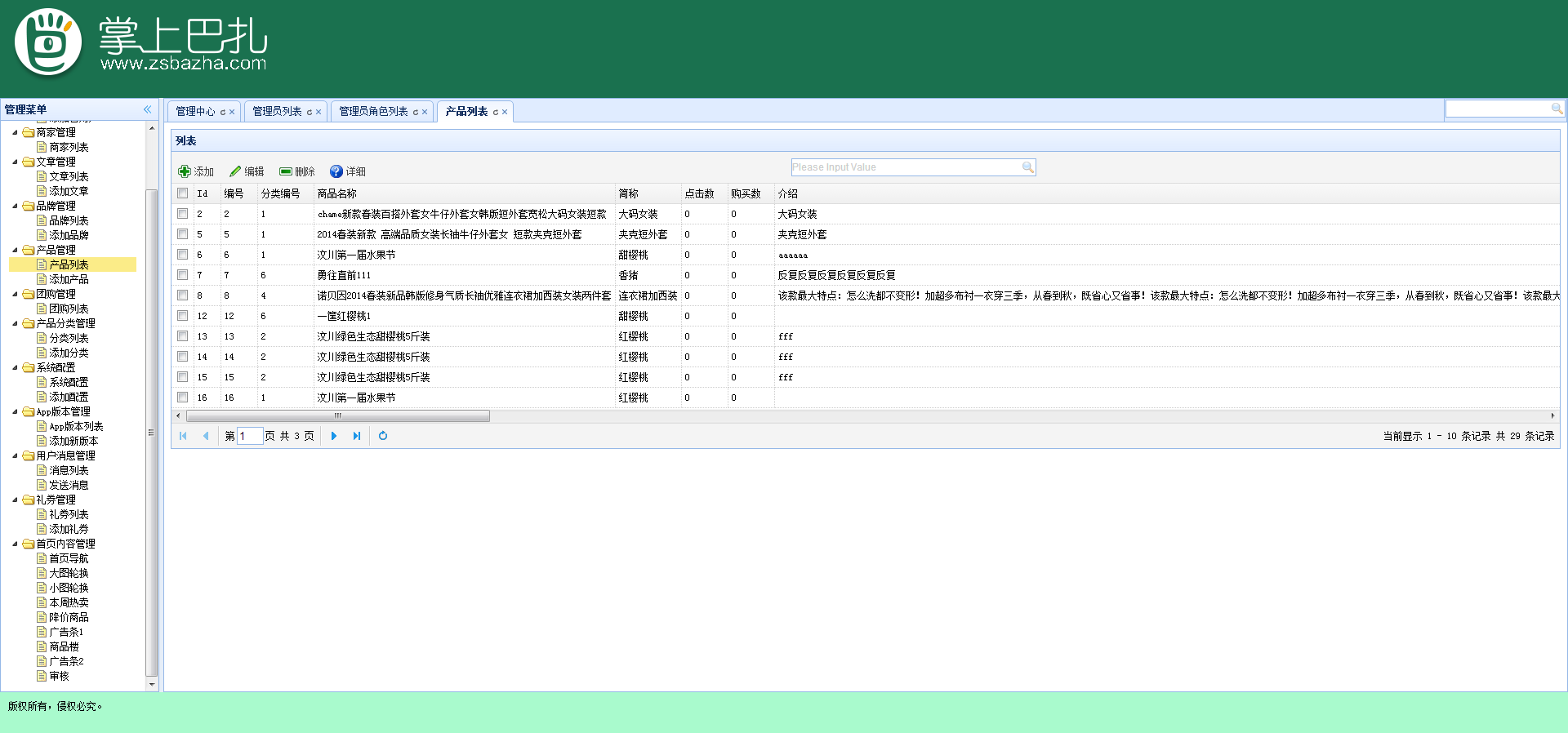
关键代码

|  |
| --- |
| using System;  using System.Collections.Generic;  using System.Linq;  using System.Web;  using System.Web.Mvc;  using SuperMarket.Model;  using System.Web.Script.Serialization;  namespace SuperMarket.Controllers  {  internal class ClassNode  {  public int id { get; set; }  public string text { get; set; }  public string fNo { get; set; }  public string fName { get; set; }  public string fDescribe { get; set; }  public int fRankId { get; set; }  public int fState { get; set; }  public string fLogo { get; set; }  public List<ClassNode> children { get; set; }  }  public class tProductClassController : Controller  {  JavaScriptSerializer js = new JavaScriptSerializer();  SuperMarket.BLL.tProductClass bll = new SuperMarket.BLL.tProductClass();  public ActionResult Index()  {  return View();  }  public ActionResult GetList()  {  List<Model.tProductClass> list = bll.GetModelList("");  List<ClassNode> tree = CreateTree(list, "-1");  string json = TreeToJson(tree);  return Content(json);  }  private List<ClassNode> CreateTree(List<Model.tProductClass> l,string parentNo)  {  List<ClassNode> list = new List<ClassNode>();  foreach( Model.tProductClass tc in l ){  if (tc.fParentNo == parentNo)  {  ClassNode cn = new ClassNode();  cn.id = tc.Id;  cn.text = tc.fName;  cn.fNo = tc.fNo;  cn.fName = tc.fName;  cn.children = CreateTree(l, tc.fNo);  cn.fDescribe = tc.fDescribe;  cn.fLogo = tc.fLogo;  cn.fRankId = tc.fRankId;  cn.fState = tc.fState;  list.Add(cn);  }  }  return list;  }  private string TreeToJson(List<ClassNode> list)  {  if (list == null) return "null";  string json = "[";  for (int i = 0; i < list.Count; i++)  {  var m = list[i];  string s = "{" + string.Format("\"id\":\"{0}\",\"fNo\":\"{1}\",\"text\":\"{8}\",\"fName\":\"{2}\",\"fDescribe\":\"{3}\",\"fRankId\":{4},\"fState\":{5},\"fLogo\":\"{6}\",\"children\":{7}", m.id,m.fNo,m.fName, m.fDescribe, m.fRankId, m.fState, m.fLogo, TreeToJson(m.children),m.text) + "}";  json += s;  if (i < list.Count - 1)  {  json += ",";  }  }  json += "]";  return json;  }  public ActionResult Details(int id)  {  Model.tProductClass model = bll.GetModel(id);  if (model == null) return View("Error", new Exception("未¡ä找¨°到Ì?该?账?号?"));  return View(model);  }  public ActionResult Create()  {  return View();  }  [HttpPost]  public ActionResult Create(FormCollection collection)  {  try  {  Model.tProductClass parent = bll.GetModelByNo(collection["fParentNo"]);  if (parent == null) return View("Error", new Exception("未¡ä找¨°到Ì?该?分¤?类¤¨¤"));  // TODO: Add insert logic here  Model.tProductClass model = new Model.tProductClass();  model.Id = 1;  model.fNo = "";  model.fParentNo = parent.fNo;  model.fRootNo = parent.fRootNo;  model.fName = collection["fName"];  model.fDescribe = collection["fDescribe"];  model.fPath = parent.fPath+","+parent.fNo;  model.fRankId = -1;  model.fChildrenCount = 0;  model.fState = 1;  HttpPostedFileBase f = Request.Files["fLogo"];  if (f == null) throw new Exception("未¡ä检¨¬测a到Ì?上¦?传ä?的Ì?文?件t，ê?请?检¨¬查¨¦后¨®再¨´试º?。¡ê");  string name = Guid.NewGuid().ToString();  string ex = f.FileName.Substring(f.FileName.LastIndexOf("."));  string path = "/Upload/icon/" + name + ex;  f.SaveAs(Server.MapPath(path));  model.fLogo = path;    int k= bll.Add(model);  model.Id = k;  model.fNo = k.ToString();  bll.Update(model);  return RedirectToAction("Index");  }  catch (Exception err)  {  return View("Error", err);  }  }  public ActionResult Edit(int id)  {  Model.tProductClass model = bll.GetModel(id);  if (model == null) return View("Error", new Exception("未¡ä找¨°到Ì?该?账?号?"));  return View(model);  }  [HttpPost]  public ActionResult Edit(int id, FormCollection collection)  {  try  {  // TODO: Add update logic here  Model.tProductClass model = bll.GetModel(id);  if (model == null) return View("Error", new Exception("未¡ä找¨°到Ì?该?账?号?"));  Model.tProductClass parent = bll.GetModelByNo(collection["fParentNo"]);  if (parent == null) return View("Error", new Exception("未¡ä找¨°到Ì?该?分¤?类¤¨¤"));  model.fNo = collection["fNo"];  model.fParentNo = parent.fNo;  model.fRootNo = parent.fRootNo;  model.fName = collection["fName"];  model.fDescribe = collection["fDescribe"];  model.fPath = parent.fPath + "," + parent.fNo;  model.fRankId = -1;  model.fState = 1;  HttpPostedFileBase f = Request.Files["fLogo"];  if (f != null)  {  string name = Guid.NewGuid().ToString();  string ex = f.FileName.Substring(f.FileName.LastIndexOf("."));  string path = "/Upload/icon/" + name + ex;  f.SaveAs(Server.MapPath(path));  model.fLogo = path;  }    bll.Update(model);  return RedirectToAction("Index");  }  catch (Exception err)  {  return View("error", err);  }  }  public ActionResult Delete(int id)  {  Model.tProductClass model = bll.GetModel(id);  if (model == null) return View("Error", new Exception("未¡ä找¨°到Ì?该?账?号?"));  return View(model);  }  [HttpPost]  public ActionResult Delete(int id, FormCollection collection)  {  try  {  Model.tProductClass model = bll.GetModel(id);  if (model == null) return View("error", new Exception("未¡ä找¨°到Ì?该?账?号?"));  bll.Delete(id);  return RedirectToAction("Index");  }  catch (Exception err)  {  return View("error", err);  }  }  }  } |

程序截图



产品分类截图



1. 礼券管理

礼券内容包括礼券名称、总数、截止日期、面值、所需消费金额等。礼券可以发放给在便民超市中消费过的用户。

ER模型

数据字典

关键代码

程序截图

1. 团购管理

团购活动针对便民超市内注册用户进行，商品管理员制定团购规则，系统超级管理员审核通过后，团购活动上线。

ER模型

数据字典

关键代码

程序截图

1. 文章管理

文章管理主要是商城新闻，丰富商城内容。

ER模型

数据字典

关键代码

程序截图

1. 首页显示内容管理

对于任何一个网站来说，首页是他的门户。放在首页上的内容总是被浏览者先看见，进而吸引顾客。系统严格控制首页每一块的内容，保证有价值有意义的内容显示在首页更能吸引顾客的眼睛。

ER模型

数据字典

关键代码

程序截图

8、用户管理

ER模型

数据字典

关键代码

程序截图

9、品牌管理

ER模型

数据字典

关键代码

程序截图

10、App版本管理

ER模型

数据字典

关键代码

程序截图

11、用户消息管理

ER模型

数据字典

关键代码

程序截图

1. 总结
2. 结语