广东科学技术职业学院

计算机工程技术学院(软件学院)

**毕 业 设 计**

|  |  |
| --- | --- |
| **题目**： 风御轩 网上书店 | |
| 专业： 软件技术 | 班级： 13ACCP2班 |
| 学生姓名： 许少丰 | 学号： 0104130235 |
| 指导教师姓名： 扶卿妮 | 职称： 讲师 |

2016年 月

**广东科学技术职业学院计算机工程技术学院**

**（ 软件技术 ）专业毕业设计任务书**

填表日期： 2016 年 4 月 15 日 填表人：许少丰

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 项目名 | 风御轩 网上书店 | | | | | | |
| 指导  教师 | 扶卿妮 | 职称 | 讲师 | 人数 | 1 | 班别 | 13ACCP2 |
| 项目  简介 | 风御轩网上书店，由后台控制管理出发，通过对书籍信息、顾客信息等数据的操控，实现管理员对前台客户端网上书店系统的优化管理 | | | | | | |
| 设  计  任  务  、  目  标  与  计  划 | 阐述毕业设计具体要完成的任务及达到的目标：  一、后台：实现前台模块内容更新显示、查看评价、管理用户和管理员、书籍管理、查看处理订单等功能  二、前台：实现客户登录注册、分类检索、书籍推荐、书名/作者查询、图书购买及查看购买状态、评价图书等环节  阐述毕业设计完成的具体计划：   1. 完成需求设计:明确后台管理的处理数据和操作逻辑，参考当当网、京东和亚马逊网上书店，优化客户端显示 2. 设计实现数据库:使用sqlserver2012数据库创建数据库脚本文件 3. 搭建框架:本项目应用ASP .NET MVC框架和前端框架AmazeUI,分层次（DAL、BLL、Common、Model、Web）以图高效代码开发和后台界面优化 4. 后台代码和后台管理页面：使用Entity Framework解决方案生成后台Model模型车；使用linq语句查询处理数据库，完成DAL访问层搭建；通过接口的定义域实现，降低代码耦合度 5. 前端客户使用界面：使用html、css与jquery搭建客户端显示效果，使用Ajax与json构建前后台交互 6. 测试、修复BUG：使用Google Chrome浏览器和Mozilla Firefox浏览器测试显示效果与功能实现 7. 优化功能，发布项目 | | | | | | |

|  |
| --- |
| 指导教师评语：  指导教师评分：  指导教师签名：  年 月 日 |
| 答辩专家组对毕业设计答辩评议及成绩评定：  答辩组长： （签章）  年 月 日 |
|  |
| 学院毕业审核意见：  院长： （签章）  年 月 日 |
|  |

**目录**

[一、引言 9](#_Toc448786345)

[二、项目需求分析 10](#_Toc448786346)

[1.后台管理 10](#_Toc448786347)

[第一，管理员身份验证 10](#_Toc448786348)

[第二，资源管理和查看客户评价 10](#_Toc448786349)

[第三，人员管理 10](#_Toc448786350)

[第四，页面模块 10](#_Toc448786351)

[第五，图书类型 10](#_Toc448786352)

[第六，书籍管理 10](#_Toc448786353)

[第七，订单管理 11](#_Toc448786354)

[第八，公告 11](#_Toc448786355)

[2.前台功能 11](#_Toc448786356)

[第一，客户登录注册 11](#_Toc448786357)

[第二，书籍检索 11](#_Toc448786358)

[第三，新书上架 11](#_Toc448786359)

[第四，热门书籍 11](#_Toc448786360)

[第五，站长推荐： 12](#_Toc448786361)

[第六，购物车功能 12](#_Toc448786362)

[第七，交易处理 12](#_Toc448786363)

[三、项目设计 13](#_Toc448786364)

[1.前台功能 13](#_Toc448786365)

[2.后台功能 13](#_Toc448786366)

[3.前后台交互 13](#_Toc448786367)

[4.数据库 14](#_Toc448786368)

[四、项目实现 15](#_Toc448786369)

[1.数据库 15](#_Toc448786370)

[第一， 人员信息表 15](#_Toc448786371)

[第二， 杂项表 15](#_Toc448786372)

[第三， 图书类型表 15](#_Toc448786373)

[第四， 书籍信息表 16](#_Toc448786374)

[第五， 特别推荐表 16](#_Toc448786375)

[第六， 客户订单表 17](#_Toc448786376)

[第七， 订单详细表 17](#_Toc448786377)

[2.后台代码：以图书信息处理为例 18](#_Toc448786378)

[第一， Models层 18](#_Toc448786379)

[第二， Common层 19](#_Toc448786380)

[第三， DAL层 23](#_Toc448786381)

[第四， BLL层 31](#_Toc448786382)

[3.后台管理 36](#_Toc448786383)

[第一，效果图 36](#_Toc448786384)

[第二，控制器代码 41](#_Toc448786385)

[第三，母版页代码 45](#_Toc448786386)

[第四，Rezor视图代码 48](#_Toc448786387)

[第五，JS代码 59](#_Toc448786388)

[4.前台页面 66](#_Toc448786389)

[五、总 结 70](#_Toc448786390)

[六、致谢 70](#_Toc448786391)

**风御轩 网上书店**

摘 要：

随着信息技术的不断发展，越来越多的人喜欢上了网络购物，网上购物不但经济实惠，而且在种类上，也相对来讲更受广大消费者的喜欢。所以，无数个网络公司把握住了这个机遇，就这样，无限个B2B电子商务平台就这样出现了。

在电子商务网站中，网上书店是目前应用最广、最成功的典范之一。通过这类的网站，用户不但可以充分利用Internet的发展给人类带来的便捷，突破时间和空间上的限制，实现网上购书。还有很多的资源供消费者选择，所以，这是电子商务网站发展的很好的一个原因。

可随着需求的不断增加，为了更好的实现后台管理，以求优化服务质量，获得更多用户认可与好评，电子商务之后也亟需有一个强有力的后台控制管理工具，故而本次设计立足于电子商务网上书店，进而深入后台管理，论述了网上购书后台管理系统和前台系统的功能设计与实现过程，包括前后台的系统需求分析和系统的功能设计、数据库设计以及主要razors视图页面的设计。本文根据系统功能需求，包含了用户注册登录、购物车、图书列表和订单生成等前台功能模块，以及书籍管理、订单详细和处理、前台模块内容显示等后台管理系统功能模块。

网上购书系统是目前应用最广的一个电子商务平台，目前有很多的大型网上购书系统平台如当当网，卓越亚马逊图书等网上购书网站，所以，网上购书系统越来越受人们的喜爱。

**关键词：**网上书店 amazeUI .NET 电子商务

Abstract：

With the development of it, more and more people like the network shopping, online shopping not only affordable, and in species, also relatively by vast consumers like more. So, many network company grasped the opportunity, so, infinite a B2B e-commerce platform so appeared.

In e-commerce sites, online bookstore is the most widely used at present, the most successful one of the model. Through this kind the website, the user can not only make full use of the development of the Internet brings human and convenient, the breakthrough time and space limitations, realize the online bookstores. Still have a lot of resources for the consumer choice, so, this is the development of e-commerce sites is a good reason.

With increasing demand, in order to achieve better management background, in order to optimize the quality of service, more user recognition and praise, electronic commerce also needs to have a strong background management tool. Therefore, this design based on e-commerce online book store, and then in-depth background management, discusses the background management system for online bookstores and the foreground of the system function design and implementation process, including the design of the foreground and background of the system requirements analysis and system functional design, database design and the main razors view page. The according to the system functional requirements, contains the user login, shopping cart, book list and order generation in front of the function module, as well as the management of books, order details and handle, a front module content display background management system function module.Online bookstores system is currently the most widely used an electronic commercial platform,there are a lot of large online bookstores system platform as dangdang network,outstanding books, amazon online bookstores website, so, online bookstores system more and more get of people love.

**Keywords：**Online shopping amazeUI .NET Electronic business

# 一、引言

随着互联网的发展，已经有越来越多的企业开始创建自己的网站。基于Internet的信息服务，商务服务已经成为了现代企业的一项不可缺少的内容。很多企业已经不满足于只建立一个简单的而且仅仅能够发布信息的静态网站。就目前而言，现代企业需要的是一个功能强大，能提供完善电子信息服务的动态商务网站。

ASP.NET是.NET FrameWork的一部分，是一项微软公司的技术，是一种使嵌入网页中的脚本可由因特网服务器执行的服务器端脚本技术，它可以在通过HTTP请求文档时再在Web服务器上动态创建它们。 指 Active Server Pages（动态服务器页面），运行于 IIS（Internet Information Server 服务，是Windows开发的Web服务器）之中的程序。

ASP.NET MVC 是微软官方提供的以MVC模式为基础的ASP.NET Web应用程序(Web Application)框架,它由Castle的MonoRail而来，MVC 是三种 ASP.NET 编程模式中的一种，是一种使用 MVC（Model View Controller 模型-视图-控制器）设计创建 Web 应用程序的模式。MVC 分层有助于管理复杂的应用程序，同时也简化了分组开发。不同的开发人员可同时开发视图、控制器逻辑和业务逻辑。

在毕业设计过程中，本人采用了ASP.NET技术，来构建一个能实现简单的电子商务的小型动态的商务网站——风御轩网上书店。该系统能够基本实现用户的注册，登录功能，而且能够实现商品的订购和查看等功能。该系统基本上具备了一个网上商品销售系统应该具备的功能，也体现了构建一个动态商务网站所需要的技术，可以说，目前的商务网站也是这种小型网站的扩充和重复。而且，随着后台管理的设计开发，书店管理员对客户的服务也在一定程度上有了优化、提升，而工作处理订单流程，图书分类、上架进货等，也有极大的辅助。

# 二、项目需求分析

## 1.后台管理

### 第一，管理员身份验证

只有管理员级别方可进入后台，进行相关操作。普通客户无权限进入后台管理页面

### 第二，资源管理和查看客户评价

一目了然，明确知道当前书籍数量和今天的交易量，历史成交量和截至目前位置总注册客户数量；可查看到客户对书店管理员的评价，优化服务质量

### 第三，人员管理

**管理员管理**：分配人员管理

**客户管理**：查看注册客户的非隐私注册信息

### 第四，页面模块

**店主推荐**：设置前台客户端首页的正中央区域的显示内容，促进相关书籍的销售量和知名度

**热门搜索**：设置搜索框下方的搜索词语，方便客户搜索处理

### 第五，图书类型

设置图书类型，方便对海量书籍的管理，以及前台客户对相关类型图书的了解。同时实现对前台客户端页面显示内容的处理

### 第六，书籍管理

对书籍进行上架、进货，信息验证等处理

### 第七，订单管理

实时显示当前相关订单信息，方便订单处理，优化书店服务质量

### 第八，公告

用于明确告知管理员当前工作任务，及时处理订单

## 2.前台功能

### 第一，客户登录注册

当客户希望进入会员中心或我的账户页面，查看相关个人信息时；或是希望购买商品时，验证当前是否为已登录状态。验证为否时，自动链接跳转至用户登录注册页面

当客户登录注册时，验证客户操作数据，并给予相关提示信息。注册用户名时验证用户名是否已存在，要求用户名数据具有唯一性

### 第二，书籍检索

**书籍分类检索**：根据书籍的类型，分类显示相关书籍信息，方便客户查询检索到目标书籍

**作者检索**：根据作者查询显示与其相关的书籍

### 第三，新书上架

根据书籍的后台创建处理时间，优先将最新上架的书籍推荐给客户

### 第四，热门书籍

根据书籍的销售数量，优先将销量领先的书籍推荐给客户

### 第五，站长推荐：

正中央区域的显示，促进相关书籍的销售量和知名度

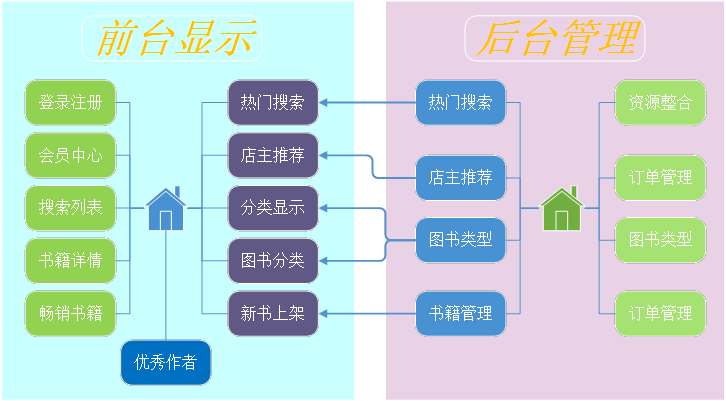
### 第六，购物车功能

记录当前客户希望购买，但未进行支付请求的书籍数据进行记录，优化客户购买操作

### 第七，交易处理

客户在完成购买支付操作后，查看当前订单是否已被管理员处理，确认发货。收到货物后完成确认收货操作，可对本次交易进行评价

1. 项目设计



1.前台功能

前台需具备电子商务平台普遍的用户登录和注册、图书的浏览和查找、购物车、订单管理等功能，另再补充店主推荐、新书上架、优秀作者、热门搜索功能优化用户体验

2.后台功能

后台需具备管理员登录、人员管理、订单管理、书籍管理、图书类型管理、店主推荐等实用性高的管理设计功能

3.前后台交互

当管理员在后台进行某些数据操作时，前台会自动更新对应的数据或显示效果:(1)后台添加或更新热门搜索信息，前台页首的搜索框下对于热搜关键字更新；（2）后台添加书籍信息，前台新书上架模块更新显示此书籍；（3）后台店主推荐更新或添加时，前台首页中央区域对应更新显示；（4）后台处理图书分类，前台图书分类列表相应更新；对应分类显示模块对类型的显示数据和效果也要有更新；（5）前台用户完成购书付款操作后，后台更新待处理订单数据，方便管理员及时处理；（6）前台客户完成交易后进行评价，及时更新到后台首页。

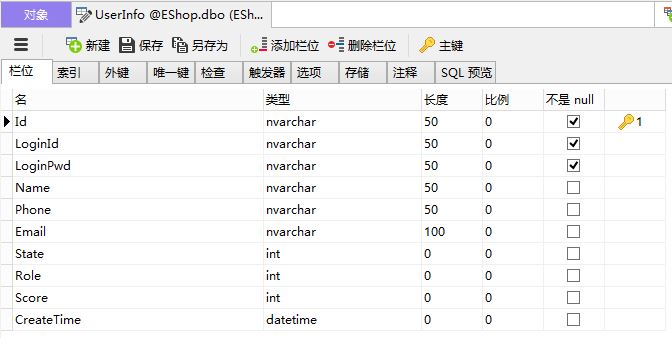
4.数据库

数据库需满足处理人员信息（管理员、注册会员）数据、图书信息数据、订单数据、类型数据、推荐书籍数据等相关需要。

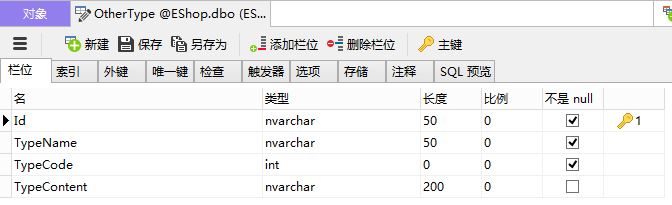
1. 项目实现

1.数据库

1. 人员信息表（记录所有人员信息）



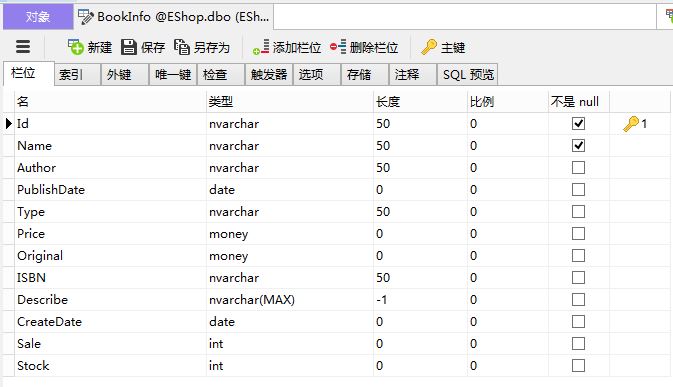
1. 杂项表（用于处理杂乱细小的分类整合）



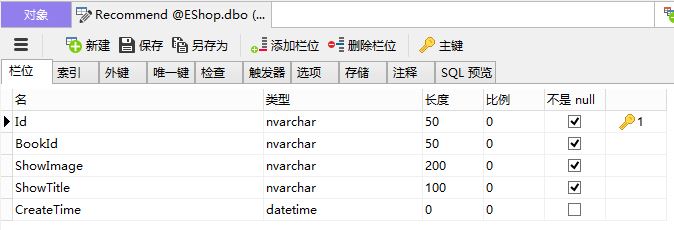
1. 图书类型表



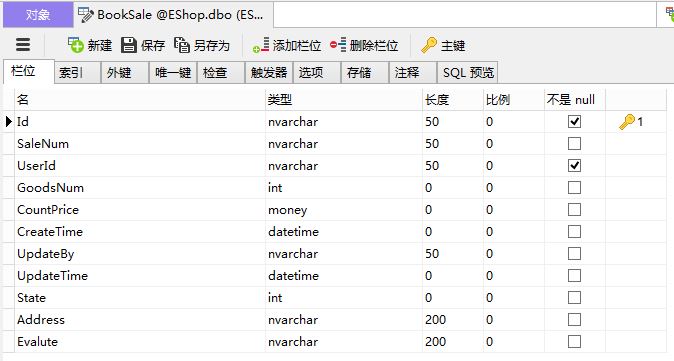
1. 书籍信息表



1. 特别推荐表



1. 客户订单表



1. 订单详细表



2.后台代码：以图书信息处理为例

* 1. Models层

//图书信息类

namespace EShop.Models

{

using System;

using System.Collections.Generic;

public partial class BookInfo

{

public string Id { get; set; }

public string Name { get; set; }

public string Author { get; set; }

public Nullable<System.DateTime> PublishDate { get; set; }

public string Type { get; set; }

public Nullable<decimal> Price { get; set; }

public Nullable<decimal> Original { get; set; }

public string ISBN { get; set; }

public string Describe { get; set; }

public Nullable<System.DateTime> CreateDate { get; set; }

public Nullable<int> Sale { get; set; }

public Nullable<int> Stock { get; set; }

}

}

* 1. Common层

//分页共用

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

namespace EShop.Common

{

public class Pager

{

private int \_pageIndex;

private int \_pageSize;

private int \_total ;

private int \_pageCount;

private int \_start;

private int \_end;

private bool \_hasPrev;

private bool \_hasNext;

private string \_remark;

/// <summary>

/// 当前页

/// </summary>

public int PageIndex

{

get { return \_pageIndex; }

set

{

if (value < 1)

{

\_pageIndex = 1;

}

else if (value > PageCount)

{

\_pageIndex = PageCount;

}

else

{

\_pageIndex = value;

}

}

}

/// <summary>

/// 每页记录数

/// </summary>

public int PageSize {

get

{

return \_pageSize;

}

set

{

if (value < 1)

{

\_pageSize = 1;

}

else

{

\_pageSize = value;

}

}

}

/// <summary>

/// 总页数

/// </summary>

public int PageCount

{

get

{

return \_pageCount;

}

}

/// <summary>

/// 总记录数

/// </summary>

public int Total

{

get

{

return \_total;

}

set

{

if (value < 0)

{

\_total = 0;

}

else

{

\_total = value;

}

}

}

/// <summary>

/// 起始页码

/// </summary>

public int Start

{

get

{

return \_start;

}

}

/// <summary>

/// 结束页码

/// </summary>

public int End

{

get

{

return \_end;

}

}

/// <summary>

/// 是否有上一页

/// </summary>

public bool HasPrev

{

get

{

return \_hasPrev;

}

}

/// <summary>

/// 是否有下一页

/// </summary>

public bool HasNext

{

get

{

return \_hasNext;

}

}

/// <summary>

/// 备注信息

/// </summary>

public string Remark

{

get { return \_remark; }

set { \_remark = value; }

}

/// <summary>

/// 唯一的构造函数

/// </summary>

/// <param name="pageIndex">当前页</param>

/// <param name="pageSize">页显示记录数</param>

/// <param name="total">总记录数</param>

public Pager(int pageIndex,int pageSize,int total) {

this.PageSize = pageSize;

this.Total = total;

\_pageCount=\_total % \_pageSize == 0 ? \_total / \_pageSize : (\_total / \_pageSize) + 1;

if (\_pageCount == 0) \_pageCount = 1;//总页数不能为0，最小也为1

PageIndex = pageIndex;

\_start = \_pageIndex - 3 > 1 ? \_pageIndex - 3 : 1;

\_end = \_pageIndex + 3 < PageCount ? \_pageIndex + 3 : PageCount;

\_hasPrev = \_pageIndex > 1;

\_hasNext = \_pageIndex < PageCount;

}

}

}

* 1. DAL层

1. 接口

using EShop.Models;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace EShop.DAL

{

public interface IBookInfoService

{

/// <summary>

/// 根据关键字分页获取相关图书信息

/// </summary>

/// <param name="search">搜索关键字(书名/作者)</param>

/// <param name="pageIndex">当前页数</param>

/// <param name="pageSize">显示数量</param>

/// <returns></returns>

List<BookInfo> PagingFindBookInfo(string search, int pageIndex, int pageSize);

/// <summary>

/// 统计对应关键字的数量

/// </summary>

/// <param name="search"></param>

/// <returns></returns>

int GetCount(string search);

/// <summary>

/// 添加图书信息

/// </summary>

/// <param name="bookInfo"></param>

/// <returns></returns>

bool InsertBookInfo(BookInfo bookInfo);

/// <summary>

/// 删除图书信息

/// </summary>

/// <param name="id"></param>

/// <returns></returns>

bool DeleteBookInfo(string id);

/// <summary>

/// 更新图书信息

/// </summary>

/// <param name="bookInfo"></param>

/// <returns></returns>

bool UpdateBookInfo(BookInfo bookInfo);

/// <summary>

/// 更新图书销售量（出售）

/// </summary>

/// <param name="id"></param>

/// <param name="num"></param>

/// <returns></returns>

bool UpdateBookInfoSale(string id, int num);

/// <summary>

/// 更新图书库存量（进货）

/// </summary>

/// <param name="id"></param>

/// <param name="num"></param>

/// <returns></returns>

bool UpdateBookInfoStock(string id, int num);

/// <summary>

/// 根据图书id获取图书信息

/// </summary>

/// <param name="id"></param>

/// <returns></returns>

BookInfo FindById(string id);

/// <summary>

/// 根据图书类型分页获取相关图书信息(后台)

/// </summary>

/// <param name="type"></param>

/// <param name="pageIndex"></param>

/// <param name="pageSize"></param>

/// <returns></returns>

List<BookInfo> FindByType(string type, int pageIndex, int pageSize);

/// <summary>

/// 统计对应图书类型的数量

/// </summary>

/// <param name="type"></param>

/// <returns></returns>

int GetTypeCount(string type);

/// <summary>

/// 根据图书类型获取指定数量的图书信息(前台界面显示)

/// </summary>

/// <param name="type"></param>

/// <param name="size"></param>

/// <returns></returns>

List<BookInfo> FindByType(string type, int size);

/// <summary>

/// 根据上架时间获取图书信息(前台界面显示)

/// </summary>

/// <param name="size"></param>

/// <returns></returns>

List<BookInfo> FindTopCreate(int size);

/// <summary>

/// 根据销售量获取图书信息(前台界面显示)

/// </summary>

/// <param name="size"></param>

/// <returns></returns>

List<BookInfo> FindTopSale(int size);

/// <summary>

/// 查询指定数量的作者(前台界面显示)

/// </summary>

/// <param name="size"></param>

/// <returns></returns>

List<string> FindAuthor(int size);

/// <summary>

/// 根据作者查询相关图书信息(前台界面显示)

/// </summary>

/// <param name="author"></param>

/// <returns></returns>

List<BookInfo> FindByAuthor(string author);

}

}

1. 实现接口

using EShop.Models;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace EShop.DAL.Service

{

public class BookInfoService : IBookInfoService

{

public List<BookInfo> PagingFindBookInfo(string search, int pageIndex, int pageSize)

{

using (EShopDB db = new EShopDB())

{

if (pageIndex == 0)

pageIndex = 1;

var obj = from b in db.BookInfo

where b.Name.Contains(search) || b.Author.Contains(search)

orderby b.CreateDate descending

select b;

List<BookInfo> list = obj.Skip((pageIndex - 1) \* pageSize).Take(pageSize).ToList();

return list;

}

}

public int GetCount(string search)

{

using (EShopDB db = new EShopDB())

{

int count = (from b in db.BookInfo

where b.Name.Contains(search) || b.Author.Contains(search)

select b).Count();

return count;

}

}

public bool InsertBookInfo(BookInfo bookInfo)

{

using (EShopDB db = new EShopDB())

{

db.BookInfo.Add(bookInfo);

bool isSuccess = db.SaveChanges() > 0;

return isSuccess;

}

}

public bool DeleteBookInfo(string id)

{

using (EShopDB db = new EShopDB())

{

BookInfo bookInfo = new BookInfo() { Id = id };

db.BookInfo.Attach(bookInfo);

db.BookInfo.Remove(bookInfo);

bool isSuccess = db.SaveChanges() > 0;

return isSuccess;

}

}

public bool UpdateBookInfo(BookInfo bookInfo)

{

using (EShopDB db = new EShopDB())

{

var oldBookInfo = db.BookInfo.Find(bookInfo.Id);

if (oldBookInfo != null)

{

oldBookInfo.Name = bookInfo.Name;

oldBookInfo.Author = bookInfo.Author;

oldBookInfo.Price = bookInfo.Price;

oldBookInfo.PublishDate = bookInfo.PublishDate;

oldBookInfo.Type = bookInfo.Type;

oldBookInfo.ISBN = bookInfo.ISBN;

bool isSuccess = db.SaveChanges() > 0;

return isSuccess;

}

else

return false;

}

}

public bool UpdateBookInfoSale(string id, int num)

{

using (EShopDB db = new EShopDB())

{

var bookInfo = db.BookInfo.Find(id);

if (bookInfo.Sale == null)

bookInfo.Sale = 0;

bookInfo.Sale = bookInfo.Sale + num;

bookInfo.Stock = bookInfo.Stock - num;

return db.SaveChanges() > 0;

}

}

public bool UpdateBookInfoStock(string id, int num)

{

using (EShopDB db = new EShopDB())

{

var bookInfo = db.BookInfo.Find(id);

if (bookInfo.Stock == null)

bookInfo.Stock = 0;

bookInfo.Stock = bookInfo.Stock + num;

return db.SaveChanges() > 0;

}

}

public BookInfo FindById(string id)

{

using (EShopDB db = new EShopDB())

{

var obj = (from b in db.BookInfo

where b.Id == id

select b).FirstOrDefault();

return obj;

}

}

public List<BookInfo> FindByType(string type, int pageIndex, int pageSize)

{

using (EShopDB db = new EShopDB())

{

if (pageIndex == 0)

pageIndex = 1;

var obj = from b in db.BookInfo

where b.Type == type

orderby b.CreateDate descending

select b;

List<BookInfo> list = obj.Skip((pageIndex - 1) \* pageSize).Take(pageSize).ToList();

return list;

}

}

public int GetTypeCount(string type)

{

using (EShopDB db = new EShopDB())

{

int count = (from b in db.BookInfo

where b.Type == type

select b).Count();

return count;

}

}

public List<BookInfo> FindByType(string type, int size)

{

using (EShopDB db = new EShopDB())

{

var obj = from b in db.BookInfo

where b.Type == type

orderby b.PublishDate descending

select b;

List<BookInfo> list = obj.Take(size).ToList();

return list;

}

}

public List<BookInfo> FindTopCreate(int size)

{

using (EShopDB db = new EShopDB())

{

var obj = from b in db.BookInfo

orderby b.CreateDate descending

select b;

List<BookInfo> list = obj.Take(size).ToList();

return list;

}

}

public List<BookInfo> FindTopSale(int size)

{

using (EShopDB db = new EShopDB())

{

var obj = from b in db.BookInfo

orderby b.Sale descending

select b;

List<BookInfo> list = obj.Take(size).ToList();

return list;

}

}

public List<string> FindAuthor(int size)

{

using (EShopDB db = new EShopDB())

{

var obj = (from b in db.BookInfo

orderby b.CreateDate descending

select b.Author).Distinct();

int count = obj.Count();

if (count > size)

return obj.Take(size).ToList();

else

return obj.ToList();

}

}

public List<BookInfo> FindByAuthor(string author)

{

using (EShopDB db = new EShopDB())

{

var obj = from b in db.BookInfo

where b.Author == author

orderby b.CreateDate descending

select b;

List<BookInfo> list = obj.ToList();

return list;

}

}

}

}

* 1. BLL层

(1)接口

using EShop.Models;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace EShop.BLL

{

public interface IBookInfoManager

{

/// <summary>

/// 根据关键字分页获取相关图书信息

/// </summary>

/// <param name="search">搜索关键字(书名/作者)</param>

/// <param name="pageIndex">当前页数</param>

/// <param name="pageSize">显示数量</param>

/// <returns></returns>

List<BookInfo> PagingFindBookInfo(string search, int pageIndex, int pageSize);

/// <summary>

/// 统计对应关键字的数量

/// </summary>

/// <param name="search"></param>

/// <returns></returns>

int GetCount(string search);

/// <summary>

/// 添加图书信息

/// </summary>

/// <param name="bookInfo"></param>

/// <returns></returns>

bool InsertBookInfo(BookInfo bookInfo);

/// <summary>

/// 删除图书信息

/// </summary>

/// <param name="id"></param>

/// <returns></returns>

bool DeleteBookInfo(string id);

/// <summary>

/// 更新图书信息

/// </summary>

/// <param name="bookInfo"></param>

/// <returns></returns>

bool UpdateBookInfo(BookInfo bookInfo);

/// <summary>

/// 更新图书销售量（出售）

/// </summary>

/// <param name="id"></param>

/// <param name="num"></param>

/// <returns></returns>

bool UpdateBookInfoSale(string id, int num);

/// <summary>

/// 更新图书库存量（进货）

/// </summary>

/// <param name="id"></param>

/// <param name="num"></param>

/// <returns></returns>

bool UpdateBookInfoStock(string id, int num);

/// <summary>

/// 根据图书id获取图书信息

/// </summary>

/// <param name="id"></param>

/// <returns></returns>

BookInfo FindById(string id);

/// <summary>

/// 根据图书id获取图书信息(处理过类型)

/// </summary>

/// <param name="id"></param>

/// <returns></returns>

BookInfo FindByIdDelType(string id);

/// <summary>

/// 根据图书类型分页获取相关图书信息(后台)

/// </summary>

/// <param name="type"></param>

/// <param name="pageIndex"></param>

/// <param name="pageSize"></param>

/// <returns></returns>

List<BookInfo> FindByType(string type, int pageIndex, int pageSize);

/// <summary>

/// 统计对应图书类型的数量

/// </summary>

/// <param name="type"></param>

/// <returns></returns>

int GetTypeCount(string type);

/// <summary>

/// 根据图书类型获取指定数量的图书信息(前台界面显示)

/// </summary>

/// <param name="type"></param>

/// <param name="size"></param>

/// <returns></returns>

List<BookInfo> FindByType(string type, int size);

/// <summary>

/// 根据上架时间获取图书信息(前台界面显示)

/// </summary>

/// <param name="size"></param>

/// <returns></returns>

List<BookInfo> FindTopCreate(int size);

/// <summary>

/// 根据销售量获取图书信息(前台界面显示)

/// </summary>

/// <param name="size"></param>

/// <returns></returns>

List<BookInfo> FindTopSale(int size);

/// <summary>

/// 查询指定数量的作者(前台界面显示)

/// </summary>

/// <param name="size"></param>

/// <returns></returns>

List<string> FindAuthor(int size);

/// <summary>

/// 根据作者查询相关图书信息(前台界面显示)

/// </summary>

/// <param name="author"></param>

/// <returns></returns>

List<BookInfo> FindByAuthor(string author);

}

}

(2)实现接口

using EShop.DAL;

using EShop.DAL.Service;

using EShop.Models;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace EShop.BLL.Manager

{

public class BookInfoManager : IBookInfoManager

{

private IBookInfoService \_bookInfoService = new BookInfoService();

private IBookTypeServicecs \_typeService = new BookTypeServicecs();

public List<BookInfo> PagingFindBookInfo(string search, int pageIndex, int pageSize)

{

List<BookInfo> list = \_bookInfoService.PagingFindBookInfo(search, pageIndex, pageSize);

foreach (BookInfo bi in list)

{

BookType bt = \_typeService.FindById(bi.Type);

bi.Type = bt.TypeName;

}

return list;

}

public int GetCount(string search)

{

return \_bookInfoService.GetCount(search);

}

public bool InsertBookInfo(BookInfo bookInfo)

{

return \_bookInfoService.InsertBookInfo(bookInfo);

}

public bool DeleteBookInfo(string id)

{

return \_bookInfoService.DeleteBookInfo(id);

}

public bool UpdateBookInfo(BookInfo bookInfo)

{

return \_bookInfoService.UpdateBookInfo(bookInfo);

}

public bool UpdateBookInfoSale(string id, int num)

{

return \_bookInfoService.UpdateBookInfoSale(id, num);

}

public bool UpdateBookInfoStock(string id, int num)

{

return \_bookInfoService.UpdateBookInfoStock(id, num);

}

public BookInfo FindById(string id)

{

BookInfo bi = \_bookInfoService.FindById(id);

return bi;

}

public BookInfo FindByIdDelType(string id)

{

BookInfo bi = \_bookInfoService.FindById(id);

BookType bt = \_typeService.FindById(bi.Type);

bi.Type = bt.TypeName;

return bi;

}

public List<BookInfo> FindByType(string type, int pageIndex, int pageSize)

{

return \_bookInfoService.FindByType(type, pageIndex, pageSize);

}

public int GetTypeCount(string type)

{

return \_bookInfoService.GetTypeCount(type);

}

public List<BookInfo> FindByType(string type, int size)

{

return \_bookInfoService.FindByType(type, size);

}

public List<BookInfo> FindTopCreate(int size)

{

return \_bookInfoService.FindTopCreate(size);

}

public List<BookInfo> FindTopSale(int size)

{

return \_bookInfoService.FindTopSale(size);

}

public List<string> FindAuthor(int size)

{

return \_bookInfoService.FindAuthor(size);

}

public List<BookInfo> FindByAuthor(string author)

{

return \_bookInfoService.FindByAuthor(author);

} } }

3.后台管理

1. 效果图

进入后台页面，首先需要进行身份验证



登录成功后，进入后台管理首页【资源整合】页面



若尚未注册管理员，请原管理员注册新管理员账户



而后，管理员便可以进行管理操作了……

此次，已书籍管理为例，点击左侧书籍管理，查看当前已有的书籍分页列表



》》添加图书，点击添加图书信息按钮





》》修改图书信息

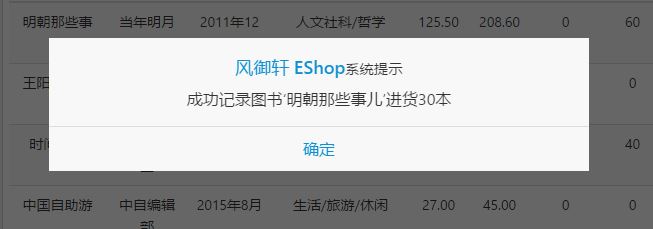




》》图书进货



每次操作完毕或出先错误，均有对应提示信息



第二，控制器代码

\*\*书籍管理对应的BookController控制器

using EShop.BLL;

using EShop.BLL.Manager;

using EShop.Common;

using EShop.Models;

using System;

using System.Collections.Generic;

using System.IO;

using System.Linq;

using System.Web;

using System.Web.Mvc;

namespace EShop.Web.Controllers

{

public class BookController : Controller

{

//

// GET: /Book/

private int biPageSize = 6;//列表页显示行数

private IBookInfoManager bookManager = new BookInfoManager();

public ActionResult Index(int? pageIndex, int? pageSize)

{

int index = pageIndex ?? 1;

int size = pageSize ?? biPageSize;

string content = String.Empty;

if (Session["searchText\_b"] != null && Session["searchText\_b"].ToString() != "")

content = (string)Session["searchText\_b"];

int count = bookManager.GetCount(content);

List<BookInfo> list = bookManager.PagingFindBookInfo(content, index, size);

Pager page = new Pager(index, size, count);

ViewData["page"] = page;

return View(list);

}

[HttpGet]

public ActionResult DelBook(string ID)

{

bool result = bookManager.DeleteBookInfo(ID);

if (!result)

TempData["mess"] = "删除图书操作失败！";

return Redirect("Index");

}

public ActionResult delUploadImg(HttpPostedFileBase file)

{

if (file != null)

{

string runPath = "~/Images/book/";

string userImg = Server.MapPath(runPath);//图书类型导航菜单显示

string fileName;

if (System.IO.File.Exists(userImg + file.FileName))

{

string extension = Path.GetExtension(file.FileName);//上传文件的拓展名

//如果存在同名文件则使用Guid生成的名字

fileName = Guid.NewGuid().ToString() + extension;

}

else

{

fileName = file.FileName;

}

Session["bookImgPath"] = runPath.Substring(1, runPath.Length - 1) + fileName;

file.SaveAs(userImg + fileName);//保存上传的图片

return Json(true);

}

else

{

return Json(false);

}

}

public ActionResult AddBook()

{

return View();

}

[HttpPost]

public ActionResult AddNewBook(string name, string author, DateTime publishDate, string type, decimal price, decimal original, string describe)

{

string imgSrc = String.Empty;

if (Session["bookImgPath"] != null && Session["bookImgPath"].ToString() != "")

imgSrc = (string)Session["bookImgPath"];

BookInfo bi = new BookInfo()

{

Id = Guid.NewGuid().ToString(),

Name = name,

Author = author,

PublishDate = publishDate,

Price = price,

Original = original,

Describe = describe,

ISBN = imgSrc,

Type = type,

Sale = 0,

Stock = 0,

CreateDate = DateTime.Now

};

Session["bookImgPath"] = null;

bool success = bookManager.InsertBookInfo(bi);

if (success)

TempData["mess"] = "成功添加新图书：" + bi.Name;

return Redirect("Index");

}

public ActionResult EditBook(string ID)

{

BookInfo bi = bookManager.FindById(ID);

return View(bi);

}

[HttpPost]

public ActionResult EditBookInfo(string biId, string name, string author, DateTime publishDate, string type, decimal price, decimal original, string describe)

{

BookInfo bi = bookManager.FindById(biId);

bi.Name = name;

bi.Author = author;

bi.PublishDate = publishDate;

bi.Type = type;

bi.Price = price;

bi.Original = original;

bi.Describe = describe;

if (Session["bookImgPath"] != null && Session["bookImgPath"].ToString() != "")

{

bi.ISBN = (string)Session["bookImgPath"];

Session["TypeImgPath"] = null;

}

bool result = bookManager.UpdateBookInfo(bi);

if (result)

TempData["mess"] = "图书‘" + bi.Name + "’修改成功";

return Redirect("Index");

}

public ActionResult AddStock(string ID)

{

BookInfo bi = bookManager.FindById(ID);

return View(bi);

}

[HttpPost]

public ActionResult AddBookStock(string biId, string stockNum)

{

int num = Int32.Parse(string.IsNullOrEmpty(stockNum) ? "0" : stockNum);

bool result = bookManager.UpdateBookInfoStock(biId, num);

if (result)

{

BookInfo bi = bookManager.FindById(biId);

TempData["mess"] = "成功记录图书‘" + bi.Name + "’进货" + num + "本";

}

return Redirect("Index");

}

[AllowAnonymous]

public ActionResult FindBySearchText(string content)

{

Session["searchText\_b"] = content;

return Redirect("Index");

}

}

}

第三，母版页代码

\*\*为了实现后台界面统一、整洁，在Views\Shared文件夹里设置母版视图

<!DOCTYPE html>

<html>

<head>

<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />

<meta name="viewport" content="width=device-width" />

<title>@ViewBag.Title - 风御轩后台管理系统</title>

<meta name="description" content="这里是风御轩后台管理系统">

<meta name="keywords" content="EShop">

<meta name="apple-mobile-web-app-title" content="Amaze UI" />

<link rel="icon" type="image/png" href="../../Images/favicon.ico">

@Styles.Render("~/bundles/amazeui/css")

@Scripts.Render("~/bundles/jquery")

</head>

<body>

<header class="am-topbar am-topbar-inverse admin-header">

<div class="am-topbar-brand">

<a href="~/"><strong>风御轩</strong> <small>后台管理系统</small></a>

</div>

<button class="am-topbar-btn am-topbar-toggle am-btn am-btn-sm am-btn-success am-show-sm-only"

data-am-collapse="{target: '#topbar-collapse'}">

<span class="am-sr-only">导航切换</span>

<span class="am-icon-bars"></span>

</button>

<div class="am-collapse am-topbar-collapse" id="topbar-collapse">

<ul class="am-nav am-nav-pills am-topbar-nav am-topbar-right admin-header-list">

<li class="am-hide-sm-only">

<a href="javascript:;" id="admin-fullscreen">

<span class="am-icon-arrows-alt"></span> <span class="admin-fullText">开启全屏</span>

</a>

</li>

</ul>

</div>

</header>

<div class="am-cf admin-main">

<!-- sidebar start -->

<div class="admin-sidebar am-offcanvas" id="admin-offcanvas">

<div class="am-offcanvas-bar admin-offcanvas-bar">

<ul class="am-list admin-sidebar-list">

<li><a href="~/Integration/Index"><span class="am-icon-qrcode"></span> 资源整合</a></li>

<li class="admin-parent">

<a class="am-cf" data-am-collapse="{target: '#collapse-nav\_1'}"><span class="am-icon-users"></span> 人员管理<span class="am-icon-angle-right am-fr am-margin-right"></span></a>

<ul class="am-list am-collapse admin-sidebar-sub am-in" id="collapse-nav\_1">

<li><a href="~/User/Show?role=admin"><span class="am-icon-user-md"></span> 管理员<span class="am-icon-star am-fr am-margin-right admin-icon-yellow"></span></a></li>

<li><a href="~/User/Show?role=customer"><span class="am-icon-user-secret"></span> 注册客户<span class="am-icon-star am-fr am-margin-right admin-icon-yellow"></span></a></li>

</ul>

</li>

<li class="admin-parent">

<a class="am-cf" data-am-collapse="{target: '#collapse-nav\_2'}"><span class="am-icon-file"></span> 页面模块<span class="am-icon-angle-right am-fr am-margin-right"></span></a>

<ul class="am-list am-collapse admin-sidebar-sub am-in" id="collapse-nav\_2">

<li><a href="~/Recommend/Index"><span class="am-icon-thumbs-up"></span> 店主推荐<span class="am-icon-star am-fr am-margin-right admin-icon-yellow"></span></a></li>

<li><a href="~/Search/Index"><span class="am-icon-fire"></span> 热门搜索<span class="am-icon-star am-fr am-margin-right admin-icon-yellow"></span></a></li>

</ul>

</li>

<li><a href="~/Type/Index"><span class="am-icon-tasks"></span> 图书类型</a></li>

<li><a href="~/Book/Index"><span class="am-icon-book"></span> 书籍管理</a></li>

<li><a href="~/Sale/Index"><span class="am-icon-suitcase"></span> 订单管理</a></li>

<li><a href="~/"><span class="am-icon-sign-out"></span> 前往客户端</a></li>

</ul>

<div class="am-panel am-panel-default admin-sidebar-panel">

<div class="am-panel-bd">

<p><span class="am-icon-bookmark"></span> 公告</p>

<p>

目前等待发货的订单数<br />

有<strong style="color:red">9</strong>单

</p>

</div>

</div>

</div>

</div>

<!-- sidebar end -->

<!-- content start -->

<div class="admin-content">

@RenderSection("featured", required: false)

<div class="am-g">

@RenderBody()

</div>

</div>

<!-- content end -->

</div>

<footer class="am-topbar am-topbar-fixed-bottom">

<p class="am-padding-left">&copy; 2016 13ACCP2 CLARENCE\_FENG 版权所有 0104130235</p>

</footer>

@Scripts.Render("~/bundles/amazeui")

@RenderSection("scripts", required: false)

</body>

</html>

第四，Rezor视图代码

》》声明引用母版\_ViewStart.cshtml

@{

Layout = "~/Views/Shared/\_Admin.cshtml";

}

》》列表页Index.cshtml

@using EShop.Models

@{

ViewBag.Title = "图书列表";

var searchStr = Session["searchText\_b"] as string;

}

@section featured {

<div class="am-cf am-padding am-margin-bottom">

<div class="am-fl am-cf"><strong class="am-text-primary am-text-lg">@ViewBag.Title</strong></div>

</div>

}

@section scripts {

<script src="~/js/admin/AlertMess.js"></script>

<script src="~/js/admin/doBook.js"></script>

}

<div class="am-u-sm-12 am-u-md-8">

<div id="TypeListHeadMenu" class="am-btn-toolbar">

<div class="am-btn-group am-btn-group-sm">

<button type="button" id="addBook" class="am-btn am-btn-secondary" onclick="AddBook();">

<span class="am-icon-plus"> 添加图书信息</span>

</button>

</div>

</div>

</div>

<div class="am-input-group">

<input type="text" class="am-form-field" id="searchText" name="searchText" placeholder="请输入图书名称" value="@searchStr" />

<span class="am-input-group-btn">

<input class="am-btn am-btn-default" type="button" value="搜索" onclick="searchBook();" />

</span>

</div>

<hr />

<input type="hidden" value="@TempData["mess"]" id="mess" />

<table id="bookInfoTab" class="am-table am-table-striped am-table-hover table-main">

<thead>

<tr id="thMenu">

<th style="display: none;">图书编号</th>

<th style="width: 12%; text-align: center;">书名</th>

<th style="width: 10%; text-align: center;">作者</th>

<th style="width: 10%; text-align: center;">出版时间</th>

<th style="width: 17%; text-align: center;">类型</th>

<th style="width: 5%; text-align: center;">售价</th>

<th style="width: 5%; text-align: center;">原价</th>

<th style="width: 8%; text-align: center;">销售量</th>

<th style="width: 8%; text-align: center;">货存量</th>

<th style="width: 25%">具体操作</th>

</tr>

</thead>

@foreach (var item in (List<BookInfo>)Model)

{

<tr>

<td style="display: none;">@item.Id</td>

<td style="text-align: center;">@item.Name</td>

<td style="text-align: center;">@item.Author</td>

@{ var dt = DateTime.Parse(@item.PublishDate.ToString());

var date = dt.GetDateTimeFormats('y')[0].ToString();

<td style="text-align: center;">@date</td>

}

<td style="text-align: center;">@item.Type</td>

@{ var d1 = decimal.Parse(@item.Price.ToString());

var d2 = decimal.Parse(@item.Original.ToString());

var p1 = Math.Round(d1, 2).ToString();

var p2 = Math.Round(d2, 2).ToString();

<td style="text-align: center;">@p1</td>

<td style="text-align: center;">@p2</td>

}

<td style="text-align: center;">@item.Sale</td>

<td style="text-align: center;">@item.Stock</td>

<td>

<div class="am-btn-toolbar">

<div class="am-btn-group am-btn-group-xs">

<button class="editBook am-btn am-btn-default am-btn-xs am-text-primary"

data-url="@Url.Content("~/Book/EditBook?ID=" + @item.Id)">

<span class="am-icon-pencil-square-o"></span>修改

</button>

<button class="addStock am-btn am-btn-default am-btn-xs am-text-primary"

data-url="@Url.Content("~/Book/AddStock?ID=" + @item.Id)">

<span class="am-icon-cart-plus"></span>进货

</button>

<button class="delBook am-btn am-btn-default am-btn-xs am-text-danger"

data-url="@Url.Content("~/Book/DelBook?ID=" + @item.Id)">

<span class="am-icon-trash-o"></span>删除

</button>

</div>

</div>

</td>

</tr>

}

</table>

@Html.Partial("\_pageNavigation")

》》新增图书信息页面AddBook.cshtml

@{

ViewBag.Title = "添加图书信息";

}

@section featured {

<div class="am-cf am-padding am-margin-bottom">

<div class="am-fl am-cf"><strong class="am-text-primary am-text-lg">@ViewBag.Title</strong></div>

</div>

}

@section scripts {

<script src="~/js/admin/AlertMess.js"></script>

<script src="~/js/admin/doBook.js"></script>

}

<div class="am-tabs" data-am-tabs>

<ul class="am-tabs-nav am-nav am-nav-tabs">

<li><a>信息录入</a></li>

</ul>

<form class="am-form am-form-horizontal am-text-primary" id="AddForm" action="@Url.Content("~/Book/AddNewBook")" method="POST">

<div class="am-margin-top am-form-group">

<label class="am-u-sm-2 am-text-right am-form-label">图书名称：</label>

<div class="am-u-sm-5 am-u-end">

<input type="text" id="name" name="name" class="am-round am-text-center" placeholder="请输入图书名称" />

<div>

&nbsp;

<div id="name\_prompt"></div>

</div>

</div>

</div>

<div class="am-form-group">

<label class="am-u-sm-2 am-form-label">作者姓名：</label>

<div class="am-u-sm-5 am-u-end">

<input type="text" id="author" name="author" class="am-round am-text-center" placeholder="请输入作者姓名" />

<div>

&nbsp;

<div id="author\_prompt"></div>

</div>

</div>

</div>

<div class="am-form-group">

<label class="am-u-sm-2 am-form-label">出版日期：</label>

<div class="am-u-sm-5 am-u-end">

<input type="date" id="publishDate" name="publishDate" class="am-round am-text-center" placeholder="请选择出版日期" />

<div>

&nbsp;

<div id="publishDate\_prompt"></div>

</div>

</div>

</div>

<div class="am-form-group">

<label class="am-u-sm-2 am-form-label">图书类型：</label>

<div class="am-u-sm-5 am-u-end">

<select id="type" name="type" class="am-round am-text-center"></select>

<div>

&nbsp;

<div id="type\_prompt"></div>

</div>

</div>

</div>

<div class="am-form-group">

<label class="am-u-sm-2 am-form-label">当前售价：</label>

<div class="am-u-sm-5 am-u-end">

<input type="text" id="price" name="price" class="am-round am-text-center" placeholder="请输入当前售价" />

<div>

&nbsp;

<div id="price\_prompt"></div>

</div>

</div>

</div>

<div class="am-form-group">

<label class="am-u-sm-2 am-form-label">原售价：</label>

<div class="am-u-sm-5 am-u-end">

<input type="text" id="original" name="original" class="am-round am-text-center" placeholder="请输入原售价" />

<div>

&nbsp;

<div id="original\_prompt"></div>

</div>

</div>

</div>

<div class="am-form-group">

<label class="am-u-sm-2 am-form-label">图书封面：</label>

<div class="am-u-sm-5 am-u-end">

<input type="button" value="选择文件..." id="browse"/>建议图片大小为：300\*300

<div>

&nbsp;<div id="upName"></div>

</div>

</div>

<div class="am-u-sm-5" id="upImg"></div>

</div>

<div class="am-form-group">

<label class="am-u-sm-2 am-form-label">描述信息：</label>

<div class="am-u-sm-5 am-u-end">

<textarea rows="5" id="describe" name="describe" placeholder="请描述此书籍的相关信息内容"></textarea>

<div>

&nbsp;

<div id="describe\_prompt"></div>

</div>

</div>

</div>

<br />

<div class="am-u-sm-10 am-u-sm-offset-2">

<button type="button" class="am-btn am-btn-primary" onclick="location.href = 'Index';">

<span class="am-icon-refresh"></span> 返 回

</button>

<button type="submit" class="am-btn am-btn-primary" style="margin-left: 10%">

<span class="am-icon-external-link"></span> 提 交

</button>

</div>

</form>

</div>

<script src="~/js/lib/plUpload/plupload.full.min.js"></script>

》》修改图书EditBook.cshtml

@using EShop.Models

@{

ViewBag.Title = "修改图书信息";

}

@section featured {

<div class="am-cf am-padding am-margin-bottom">

<div class="am-fl am-cf"><strong class="am-text-primary am-text-lg">@ViewBag.Title</strong></div>

</div>

}

@section scripts {

<script src="~/js/admin/AlertMess.js"></script>

<script src="~/js/admin/doBook.js"></script>

}

<div class="am-tabs" data-am-tabs>

<ul class="am-tabs-nav am-nav am-nav-tabs">

<li><a>修改信息</a></li>

</ul>

<form class="am-form am-form-horizontal am-text-primary" id="EditForm" action="@Url.Content("~/Book/EditBookInfo")" method="POST">

@{var oldBi = (BookInfo)Model;}

<input type="hidden" name="biId" value="@oldBi.Id" />

<div class="am-margin-top am-form-group">

<label class="am-u-sm-2 am-text-right am-form-label">图书名称：</label>

<div class="am-u-sm-5 am-u-end">

<input type="text" id="name" name="name" class="am-round am-text-center" value="@oldBi.Name" placeholder="请输入图书名称" />

<div>

&nbsp;

<div id="name\_prompt"></div>

</div>

</div>

</div>

<div class="am-form-group">

<label class="am-u-sm-2 am-form-label">作者姓名：</label>

<div class="am-u-sm-5 am-u-end">

<input type="text" id="author" name="author" class="am-round am-text-center" value="@oldBi.Author" placeholder="请输入作者姓名" />

<div>

&nbsp;

<div id="author\_prompt"></div>

</div>

</div>

</div>

<div class="am-form-group">

<label class="am-u-sm-2 am-form-label">出版日期：</label>

<div class="am-u-sm-5 am-u-end">

<input type="date" id="publishDate" name="publishDate" class="am-round am-text-center" value="@oldBi.PublishDate" placeholder="请选择出版日期" />

<div>

&nbsp;

<div id="publishDate\_prompt"></div>

</div>

</div>

</div>

<div class="am-form-group">

<label class="am-u-sm-2 am-form-label">图书类型：</label>

<div class="am-u-sm-5 am-u-end">

<input type="hidden" id="oldType" value="@oldBi.Type" />

<select id="type" name="type" class="am-round am-text-center"></select>

<div>

&nbsp;

<div id="type\_prompt"></div>

</div>

</div>

</div>

<div class="am-form-group">

<label class="am-u-sm-2 am-form-label">当前售价：</label>

<div class="am-u-sm-5 am-u-end">

<input type="text" id="price" name="price" class="am-round am-text-center" value="@oldBi.Price" placeholder="请输入当前售价" />

<div>

&nbsp;

<div id="price\_prompt"></div>

</div>

</div>

</div>

<div class="am-form-group">

<label class="am-u-sm-2 am-form-label">原售价：</label>

<div class="am-u-sm-5 am-u-end">

<input type="text" id="original" name="original" class="am-round am-text-center" value="@oldBi.Original" placeholder="请输入原售价" />

<div>

&nbsp;

<div id="original\_prompt"></div>

</div>

</div>

</div>

<div class="am-form-group">

<label class="am-u-sm-2 am-form-label">图书封面：</label>

<div class="am-u-sm-3">

@{

if (string.IsNullOrEmpty(@oldBi.ISBN))

{

<img src="~/Images/notImage.jpg" width="150" height="150" />

}

else

{

<img src="@oldBi.ISBN" width="150" height="150" />

}

}

</div>

<div class="am-u-sm-2">

<input type="button" value="重新上传图片?" id="browse" />

<br />建议图片大小为：300\*300<br />

&nbsp;<div id="upName"></div>

</div>

<div class="am-u-sm-3" id="upImg"></div>

</div>

<div class="am-form-group">

<label class="am-u-sm-2 am-form-label">描述信息：</label>

<div class="am-u-sm-5 am-u-end">

<textarea rows="5" id="describe" name="describe" placeholder="请描述此书籍的相关信息内容">@oldBi.Describe</textarea>

<div>

&nbsp;

<div id="describe\_prompt"></div>

</div>

</div>

</div>

<br />

<div class="am-u-sm-10 am-u-sm-offset-2">

<button type="button" class="am-btn am-btn-primary" onclick="location.href = 'Index';">

<span class="am-icon-refresh"></span> 返 回

</button>

<button type="submit" class="am-btn am-btn-primary" style="margin-left: 10%">

<span class="am-icon-external-link"></span> 提 交

</button>

</div>

</form>

</div>

<script src="~/js/lib/plUpload/plupload.full.min.js"></script>

》》图书进货页面AddStock.cshtml

@using EShop.Models

@{

ViewBag.Title = "仓库进货";

}

@section featured {

<div class="am-cf am-padding am-margin-bottom">

<div class="am-fl am-cf"><strong class="am-text-primary am-text-lg">@ViewBag.Title</strong></div>

</div>

}

@section scripts {

<script src="~/js/admin/AlertMess.js"></script>

<script src="~/js/admin/doBook.js"></script>

}

<div class="am-tabs" data-am-tabs>

<ul class="am-tabs-nav am-nav am-nav-tabs">

<li><a>进货信息</a></li>

</ul>

<form class="am-form am-form-horizontal am-text-primary" id="ChangeForm" action="@Url.Content("~/Book/AddBookStock")" method="POST">

@{var oldBi = (BookInfo)Model;}

<input type="hidden" name="biId" value="@oldBi.Id" />

<div class="am-margin-top am-form-group">

<label class="am-u-sm-2 am-text-right am-form-label">图书名称：</label>

<div class="am-u-sm-5 am-u-end">

<input type="text" class="am-round am-text-center" value="@oldBi.Name" readonly />

</div>

</div>

<div class="am-form-group">

<label class="am-u-sm-2 am-form-label">当前售价：</label>

@{ var d = decimal.Parse(@oldBi.Price.ToString());

var p = Math.Round(d, 2).ToString();

<label class="am-u-sm-3 am-form-label am-u-end" style="font-size:20px;color:red;">￥&nbsp;@p</label>

}

</div>

<div class="am-form-group">

<label class="am-u-sm-2 am-form-label">当前货存：</label>

<label class="am-u-sm-3 am-form-label am-u-end am-text-lg">&nbsp;@oldBi.Stock &nbsp;本</label>

@\*<div class="am-u-sm-5 am-u-end">

<strong class="am-warning am-padding-left-sm">@oldBi.Stock </strong>本

</div>\*@

</div>

<div class="am-form-group">

<label class="am-u-sm-2 am-form-label">图书封面：</label>

<div class="am-u-sm-3 am-u-end">

@{

if (string.IsNullOrEmpty(@oldBi.ISBN))

{

<img src="~/Images/notImage.jpg" width="150" height="150" />

}

else

{

<img src="@oldBi.ISBN" width="150" height="150" />

}

}

</div>

</div>

<div class="am-form-group">

<label class="am-u-sm-2 am-form-label">进货数量：</label>

<div class="am-u-sm-5 am-u-end">

<input type="hidden" id="stockNum" name="stockNum" />

<div class="item-link">

<a id="reduceNum" class="am-icon-btn am-secondary am-icon-minus">10</a>

<a id="setNum" class="am-btn am-btn-default am-btn-lg" @\*style="width:50px;height:50px;line-height:50px;font-size:20px; text-align:center;border:solid 1px blue;"\*@>0</a>

@\*<input type="text" id="stockNum" name="stockNum" style="width:50px; text-align:center;" value="0" readonly />\*@

<a id="addNum" class="am-icon-btn am-secondary am-icon-plus">10</a>

</div>

</div>

</div>

<br />

<div class="am-u-sm-10 am-u-sm-offset-2">

<button type="button" class="am-btn am-btn-primary" onclick="location.href = 'Index';">

<span class="am-icon-refresh"></span> 返 回

</button>

<button type="submit" class="am-btn am-btn-primary" style="margin-left: 10%">

<span class="am-icon-external-link"></span> 提 交

</button>

</div>

</form>

</div>

<script src="~/js/lib/plUpload/plupload.full.min.js"></script>

* 1. JS代码

由于添加修改图书信息时，设计需求包括图片显示和上传，故引用了plUpload这一JS插件，在这里只显示个人编写的js代码 (处理书籍管理的为doBook.js)

$(function () {

setType();

TypeSelect;

var addStock = 0;

$(".editBook").click(function (event) {

var href = $(this).attr("data-url");

try {

location.href = href;

} catch (err) {

alert(err);

}

});

$(".addStock").click(function (event) {

var href = $(this).attr("data-url");

try {

location.href = href;

} catch (err) {

alert(err);

}

});

$(".delBook").click(function (event) {

var href = $(this).attr("data-url");

try {

var name = $(this).parents("tr").children("td:eq(1)").text();

var alterMess = new EShop("确定要删除图书<strong class='am-text-danger'>" + name + "</strong>吗?");

var confirmFunc = function () {

location.href = href;

}

alterMess.confirm(confirmFunc);

} catch (err) {

alert(err);

}

});

$("#reduceNum").click(function () {

if (addStock >= 10) {

addStock = addStock - 10;

$("#setNum").text(addStock);

$("#stockNum").val(addStock);

}

else {

var alterMess = new EShop("<label class='am-text-danger'>添加数量不得小于0</label>");

alterMess.alert();

}

});

$("#addNum").click(function () {

addStock = addStock + 10;

$("#setNum").text(addStock);

$("#stockNum").val(addStock);

});

});

//快捷模糊查询

function searchBook() {

var content = $('#searchText').val();

location.href = "FindBySearchText?content=" + content;

}

//添加项目

function AddBook() {

window.location = "../Book/AddBook";

}

$(function () {

function validate($dom) {

var v = $dom.val();

var id = $dom.attr("id");

var flag = true;

switch (id) {

case "name":

$("#name\_prompt").html("&nbsp;");

if (v == "") {

$("#name\_prompt").removeClass().addClass("am-text-danger am-icon-close").html("图书名称不能为空！");

flag = false;

} else {

$("#name\_prompt").removeClass().addClass("am-text-success am-icon-check").html("输入正确");

}

break;

case "author":

$("#author\_prompt").html("&nbsp;");

if (v == "") {

$("#author\_prompt").removeClass().addClass("am-text-danger am-icon-close").html("作者姓名不能为空！");

flag = false;

} else {

$("#author\_prompt").removeClass().addClass("am-text-success am-icon-check").html("输入正确");

}

break;

case "publishDate":

$("#publishDate\_prompt").html("&nbsp;");

if (v == "") {

$("#publishDate\_prompt").removeClass().addClass("am-text-danger am-icon-close").html("出版日期不能为空！");

flag = false;

} else {

$("#publishDate\_prompt").removeClass().addClass("am-text-success am-icon-check").html("输入正确");

}

break;

case "type":

$("#type\_prompt").html("&nbsp;");

if (v == "") {

$("#type\_prompt").removeClass().addClass("am-text-danger am-icon-close").html("请选择图书类型！");

flag = false;

} else {

$("#type\_prompt").removeClass().addClass("am-text-success am-icon-check").html("输入正确");

}

break;

case "price":

$("#price\_prompt").html("&nbsp;");

if (v == "") {

$("#price\_prompt").removeClass().addClass("am-text-danger am-icon-close").html("当前售价不能为空！");

flag = false;

} else {

$("#price\_prompt").removeClass().addClass("am-text-success am-icon-check").html("输入正确");

}

break;

case "original":

$("#original\_prompt").html("&nbsp;");

if (v == "") {

$("#original\_prompt").removeClass().addClass("am-text-danger am-icon-close").html("原售价不能为空！");

flag = false;

} else {

$("#original\_prompt").removeClass().addClass("am-text-success am-icon-check").html("输入正确");

}

break;

default:

break;

}

return flag;

}

$("#name").blur(function () {

validate($(this));

});

$("#author").blur(function () {

validate($(this));

});

$("#publishDate").blur(function () {

validate($(this));

});

$("#type").blur(function () {

validate($(this));

});

$("#price").blur(function () {

validate($(this));

});

$("#original").blur(function () {

validate($(this));

});

$("#AddForm").submit(function () {

var flag = true;

$(this).find("input").each(function (i, ele) {

if (!validate($(ele))) {

flag = false;

}

});

if (flag === false) {

var alterMess = new EShop("<label class='am-text-danger'>输入存在错误，提交失败</label>");

alterMess.alert();

}

return flag;

});

$("#EditForm").submit(function () {

var flag = true;

$(this).find("input").each(function (i, ele) {

if (!validate($(ele))) {

flag = false;

}

});

if (flag === false) {

var alterMess = new EShop("<label class='am-text-danger'>输入存在错误，提交失败</label>");

alterMess.alert();

}

return flag;

});

});

var uploader = new plupload.Uploader({ //实例化一个plupload上传对象

browse\_button: 'browse',

url: '/Book/delUploadImg',

file\_data\_name: "file",

flash\_swf\_url: '~/js/lib/plUpload/Moxie.swf',

silverlight\_xap\_url: '~/js/lib/plUpload/Moxie.xap',

filters: {

mime\_types: [ //只允许上传图片文件

{ title: "图片文件", extensions: "jpg,gif,png" }

],

prevent\_duplicates: true,//不允许选取重复文件

multi\_selection: false//单文件

}

});

uploader.init(); //初始化

//绑定文件添加进队列事件

uploader.bind('FilesAdded', function (uploader, files) {

$.each(uploader.files, function (i, file) {

if (uploader.files.length <= 1) {

return;

}

uploader.removeFile(file);

});

var file\_name = files[0].name;

$('#upName').html('您选择的文件名: ' + file\_name);

previewImage(files[0], function (imgsrc) {

$('#upImg').empty();

$('#upImg').append('<img src="' + imgsrc + '" width="150" height="200"/>');

});

uploader.start();

$(".moxie-shim").hide();

});

function previewImage(file, callback) {

if (!file || !/image\//.test(file.type)) return;

if (file.type == 'image/gif') {

var fr = new mOxie.FileReader();

fr.onload = function () {

callback(fr.result);

fr.destroy();

fr = null;

}

fr.readAsDataURL(file.getSource());

} else {

var preloader = new mOxie.Image();

preloader.onload = function () {

preloader.downsize(150, 150);

var imgsrc = preloader.type == 'image/jpeg' ? preloader.getAsDataURL('image/jpeg', 80) : preloader.getAsDataURL();

callback && callback(imgsrc);

preloader.destroy();

preloader = null;

};

preloader.load(file.getSource());

}

}

//获取类型下拉列表

function setType() {

$.ajax({

url: '/Type/FindBookType',

type: 'POST',

dataType: 'json',

async: true,

success: function (data) {

if (data != null) {

for (var i = 0; i < data.length; i++) {

var obj = data[i];

$("#type").append("<option value=\"" + obj.Id + "\">" + obj.TypeName + "</option>");

}

}

},

error:function(xhr,textStatus,thrown){

console.log("请求异常："+xhr.status+" "+textStatus+" "+thrown);

}

});

}

function TypeSelect() {

var oldtype = $("#oldType").val();

if (oldtype != null && oldtype != "")

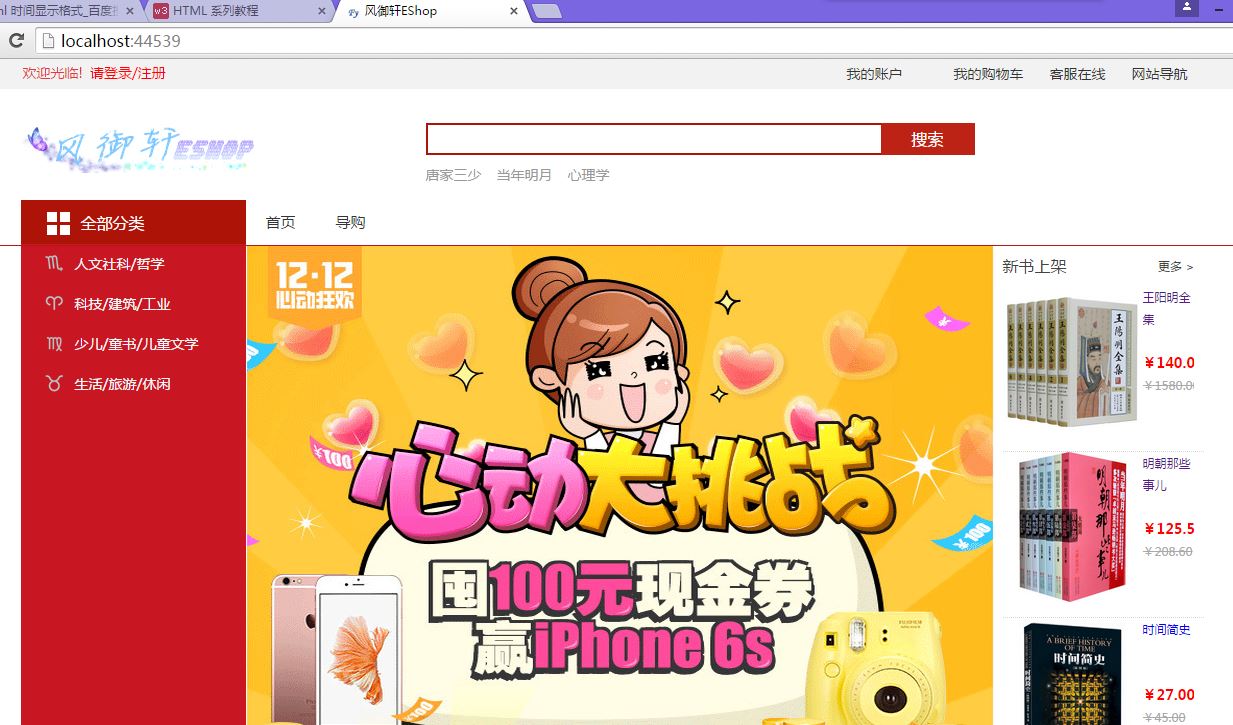
$("#type option[value='" + oldtype + "']").attr("selected", true);

}

4.前台页面

此项目设计着重点为后台管理，动态实现前台数据，故而，前台客户端只做效果显示介绍

》》前台首页















》》前台客户登录页面



》》图书详细页面



五、总 结

本设计完成了一个网上书店系统，实现的功能较为简单，后台有管理员登录、人员管理、订单管理、书籍管理、图书类型管理、店主推荐等实用性高的管理设计功能；前台有用户登录和注册、图书的浏览和查找、购物车、订单管理等几个电子商务平台较为普遍的功能和店主推荐、新书上架、优秀作者、热门搜索功能，其他扩展的功能会在以后慢慢改善。

网上书店系统在我们的日常生活中已经处于一个无法替代的地位。网上书店的出现，在选择图书种类和速度上充分的方便了我们的日常生活，提高了我们的生活效率。在此基础上，我们还要不断的改进，一方面是在系统上实施新的功能，使网上书店系统更加完美，另一方面也要完善后台管理，使网上书店后台管理人员的工作细致高效，获得更多用户的认可。

在此次设计的分析处理和开发实现过程中，我充分的感受到了大学三年所学习的知识的重要性以及连贯性，并充分的认识到了编程的不易，并学到了一些平时接触不到的知识，在工作中也巩固了对所学知识的理解程度，对于以后的生活以及工作都有很大的帮助。

六、致谢

在这里，我要感谢我的指导老师，在论文修改期间，扶卿妮老师认真负责地帮我修改程序上的一些问题以及论文上的问题。她严谨的治学精神，精益求精的工作作风和亲切亲和的言语，深深地感染我激励我。在此对她表示衷心的感谢。同时，我还要感谢帮助过我的同学，与同学之间的探讨对我平时的设计都起到了一些关键性的作用，在此，感谢我的同学。最后还要感谢所有的答辩老师，你们辛苦了，在百忙之中还有抽时间来指导我们的毕业设计，在此向你们表示衷心的感谢！

[参考文献]

[1](美国)马丁**.** 代码整洁之道[M]**.**北京：人民邮电出版社，2010．

[2]王涛**.**你必须知道的.net[M]**.**北京：电子工业出版社，2008．

[3]蒋金楠**.**ASP.NET MVC4框架揭秘[M]**.**北京：电子工业出版社，2013．

[4]jQuery教程. http://www.w3school.com.cn/jquery/

**毕业设计（论文）指导教师指导情况记录表**

**由学生填写（指导方式及内容：集中开会讨论、面授、电子邮件、QQ、电话、微信）**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 指导老师姓名 | | 扶卿妮 | 指导老师联系电话 |  |
| **开题报告指导方式及内容** | 联系方式：QQ群  问题：开题报告需要交任务书吗？  解答：否。 | | | |
| **毕业设计及（论文）撰写指导方式及内容** | 联系方式：QQ群  问题：毕业设计什么时候交？  解答：4月初。 | | | |
| **答辩前指导方式及内容** | 联系方式：QQ群  问题：毕业答辩需要什么？什么时候答辩？  解答：毕业设计PPT；4月24日早上8点。 | | | |