**设计说明书**

# 文档说明

此文档是斯多克科技内部资料，用于内部交流，需要严格保密。

# 2.系统需求分析

## 2.1 需求说明

该系统在性能功能上应达到如下需求：操作简单、界面友好、完全框架式的页面布局，使得新闻的录入工作更简便，许多选项包括新闻分类、新闻列表、新闻出处等只需要点击鼠标就可以完成；另外，跟踪出现的提示信息也让用户随时清楚自己的操作情况。对常见网站的新闻管理的各个方面：新闻录入、浏览、删除、修改、搜索等方面都大体实现，实现了网站对即时新闻的管理要求。

通过相关调查，要求该系统应具有以下功能：

1. 通过网络浏览各类新闻及相关信息。

2. 新闻能分类显示相关信息，一条新闻可以属于多个类。

3. 提供站内新闻的搜索功能。

4. 支持软件下载功能。

5. 为后台管理提供管理入口。

6. 后台编辑个类新闻并且管理新闻信息。

## 2.2 用户功能分析

对于本系统，用户共分为二：会员用户、管理员，各有不同的权限。

（一）会员用户：

此类用户具有以下权限：

1.注册会员。

2.登陆会员。

3.浏览前台网页。

4.阅读已发布新闻。

5.其它特殊权限

（二）管理员：

此类用户具有以下权限：

1. 浏览所有网页。

2. 具有管理普通用户的权限。

3. 管理新闻信息，可供用户浏览，新闻中可包含附件。

5. 管理软件，可以批量上传软件供用户下载。

6. 登陆后可修改个人密码。

7. 可以删除任何一个普通用户的所有信息。

## 2.3 用例图

浏览者和管理员的具体分工操作所涉及到的功能（如图 2.1）所示。



图2.1 用例图

## 2.4 结构图

根据需求分析得出所需要的功能，和结构得出系统结构（如图2.1）所示。



图 2.2 结构图

# 3. 总体设计方案

## 3.1开发工具的选用和介绍

### 3.1.1 JDK环境

JDK环境，这是Java系统编译运行时必须配置的环境，只有在此环境下Java代码才能够执行出结果，并且在本次设计中我们选用的Tomcat服务器的配置也必须在配置好JDK环境之后才能运行，所以本次设计我们选用了1.6的版本进行开发。

### 3.1.2 MySQL

MySQL数据库管理系统是用于关联数据库并将数据保存到所创建的表中，在本次设计中涉及到数据的存储过程，所以必须要有数据库管理系统来进行管理。目前来说MySQL一般比价适用于中小型企业的，如果数据规模非常庞大一般会选择[Oracle](http://baike.baidu.com/view/15020.htm)数据库管理系统进行开发。本次设计数据非常有限所以选择MySQL，除此之外他还有便于学习，成本低，开源等特点。

### 3.1.3 WEB服务器Tomcat

Tomcat是Apache底下的一个项目，它拥有免费、开放源码的特点因此Tomcat成为许多程序开发、调试时所选择的服务器，在本次设计过程中主要使用它来进行程序的调试。同时在本次设计中需要用到JSP和Servlet技术，Tomcat对于这些技术也是兼容的，因此它成为了我对服务器选择的首选。

在本系统中我们选用稳定版本Tomcat6.0进行开发。

### 3.1.4 集成开发工具MyEclipse

是目前最流行的Java集成开发工具。MyEclipse一经出现就以其快速、丰富的特性赢得了认可，其主要的属性有快速的性能、重构性能、快速修复错误、调整组织导入包、弹出窗口进行代码自动装配并且它能够很好地与MySQL、Tomcat进行集成。

本系统中我们选用MyEclipse8.6。

### 3.1.5 Dreamweaver CS5

Adobe Dreamweaver CS5是一款集网页制作和管理网站于一身的所见即所得网页编辑器，Dreamweaver CS5是第一套针对专业[网页设计](http://baike.baidu.com/view/8972.htm)师特别发展的视觉化网页开发工具，利用它可以轻而易举地制作出跨越平台限制和跨越[浏览器](http://baike.baidu.com/view/7718.htm)限制的充满动感的网页。

## 3.2 开发技术的介绍

### 3.2.1 JSP页面技术

JSP产生于Servlet之后，主要是为了解决Servlet中输出HTML代码效率低的问题而产生的。JSP技术比较简单，类似于ASP、PHP这些脚本语言。JSP的技术主要是由以下两个部分组成：

JSP的基本语法：两个注释、3个脚本元素、8个动作指令。

JSP的九大内置对象：request请求对象、response响应对象、pageContext页面上下文对象、session会话对象、application应用程序对象、out输出对象、config配置对象、page页面对象和exception例外对象。

### 3.2.2 Servlet技术

Servlet产生于JSP技术之前，它最大的特点就是基于线程。与传统的CGI相比，每一个CGI请求都会产生一个新的进程，而每一个Servlet请求都只会产生一个线程，这对于系统的利用率来说是一场革命。

Servlet是MVC开发的技术，所有基于J2EE的MVC框架都离不开Servlet。通常Servlet用于进行系统的初始化、服务拦截、过滤等。

### 3.2.3 HTML标签语言和JavaScript脚本语言的编写

HTML：超级文本标记语言是[标准通用标记语言](http://baike.baidu.com/view/5286041.htm)下的一个应用，也是一种规范，一种[标准](http://baike.baidu.com/view/8079.htm)，它通过标记符号来标记要显示的网页中的各个部分。

Javascript是由Netscape公司创造的一种脚本语言。为便于推广，被定为javascript，但是javascript与java是两门不相干的语言，作用也不一样。作为一门独立的编程语言，javascript可以做很多的事情，但它最主流的应用还是在Web上——创建动态网页(即网页特效)。

提要：Javascript是学习脚本语言的首选。她兼容性好，绝大多数浏览器均支持Javascript，而且她功能强大，实现简单方便，入门简单，即使是程序设计新手也可以非常快速容易地使用JavaScript进行简单的编程。

## 3.3 开发框架MVC的概述

### 3.3.1 MVC模式简介

MVC全名是Model View Controller，是模型(model)－视图(view)－控制器(controller)的缩写，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。MVC（模型Model-视图View-控制器Controller）是一种设计模式。  
 MVC模式作为系统的搭建框架，采用这种模式的好处是可以使每一层次的作用分工明确，它强制性的使[应用程序](http://baike.baidu.com/view/330120.htm)的[输入](http://baike.baidu.com/view/700406.htm)、[处理](http://baike.baidu.com/view/989420.htm)和[输出](http://baike.baidu.com/view/700402.htm)分开。使用MVC应用程序被分成三个核心部件：[模型](http://baike.baidu.com/view/96500.htm)（model）、[视图](http://baike.baidu.com/view/71981.htm)（model）、[控制器](http://baike.baidu.com/view/122229.htm)（controller）。本文采用的是JSP+Servlet+DAO三层模式来搭建整个框架。

### 3.3.2 MVC模式框架图

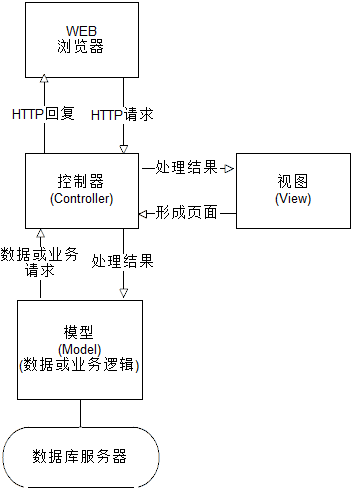


图2.1 MVC模式

# 4.数据库设计

## 4.1设计步骤

1.根据需求分析对所需存储数据进行分类，建立数据库表存储、归类所需数据，确定表中各属性的类型。

2.在数据库表建立完成后还要建立各个表之间的连接关系，确定表与表之间是主外键的关系还是多对多、一对多或是一对一等的关系。

3.分析前台与后台的功能建立用户的权限关系。

## 4.2 数据库表的设计

根据用户需求，共创建了六个表，分别为：管理员信息表admin、会员信息表users、文章信息表newsinfo、评论信息表replys、文章频道表newstypeinfo、会员留言表leavemessage。以下是新闻发布系统的数据库表结构的详细描述：

### 4.2.1 管理员信息表

管理员信息表admin包含着管理员在登录时候所需的账号、密码，还有管理员的信息，管理员的Id作为主键，在查询是可通过Id来查找管理员的信息。（如表4.1）

表4.1 管理员信息表

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 字段名 | 字段说明 | 类型 | 长度 | 允许空值 | 备注 |
| adminId | 管理员Id | int | 11 | 否 | 主键 |
| adminName | 管理员用户 | varchar | 20 | 否 |  |
| adminPass | 管理员密码 | varchar | 32 | 是 |  |
| adminInfo | 信息 | text |  | 是 |  |

### 4.2.2 会员信息表

会员信息表users包含着会员注册时候所需要的填写的所有信息，以及会员Id设为主键，管理员在查找会员信息时可以根据Id来查找。（如表4.2）

表4.2 会员信息表

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 字段名 | 字段说明 | 类型 | 长度 | 允许空值 | 备注 |
| usersId | 会员Id | int | 11 | 否 | 主键 |
| usersName | 会员用户 | varchar | 20 | 否 |  |
| usersPass | 会员密码 | varchar | 32 | 是 |  |
| usersEmail | E-mail | varchar | 50 | 是 |  |
| usersInfo | 信息 | text |  | 是 |  |
| realName | 真实姓名 | varchar | 20 | 是 |  |
| sex | 性别 | varchar | 10 | 是 |  |
| phone | 电话号码 | varchar | 20 | 是 |  |
| idNumber | 身份证号 | varchar | 30 | 是 |  |

### 4.2.3 文章信息表

文章信息表newsinfo包含着文章发表时候的全部信息，以及文章Id设为主键用于在查找文章时可以根据此标识来查找出文章的所有信息。同时还在表中设有管理员Id作为外键，用于对管理员信息表进行关联。（如表4.3）

表4.3 文章信息表

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 字段名 | 字段说明 | 类型 | 长度 | 允许空值 | 备注 |
| newsInfoId | 文章Id | int | 11 | 否 | 主键 |
| newsInfoTitle | 文章标题 | varchar | 100 | 是 |  |
| newsInfoDescribe | 文章描述 | text |  | 是 |  |
| newsInfoContent | 文章内容 | longtext |  | 是 |  |
| newsInfoTime | 创建时间 | datetime |  | 是 |  |
| newsAuthor | 文章作者 | varchar | 40 | 是 |  |
| adminId | 管理员Id | int | 11 | 是 | 外键 |
| newsType | 文章分类 | text |  | 是 |  |
| newsInfoState |  | int | 11 | 是 |  |

### 4.2.4 评论信息表(暂时没用)

评论信息表replys包含着会员评论时候的全部信息，以及评论Id设为主键用于在查找评论时可以根据此标识来查找出评论的所有信息。同时还在表中设有会员Id、文章Id作为外键，用于对会员信息表、文章信息表进行关联。（如表4.4）

表4.4 评论信息表

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 字段名 | 字段说明 | 类型 | 长度 | 允许空值 | 备注 |
| replytId | 评论Id | int | 11 | 否 | 主键 |
| replytContent | 评论内容 | longtext |  | 是 |  |
| replytTime | 评论时间 | datetime |  | 是 |  |
| usersId | 会员Id | int | 11 | 是 | 外键 |
| newsInfoId | 文章Id | int | 11 | 是 | 外键 |

### 4.2.5 文章频道表

文章频道表newstype主要是用于对文章的分类，管理员在发布新闻时需要对新闻进行分类存放到不同的频道，以便会员可以通过在查看文章频道来查找相关的文章。（如表4.5）

表4.5 文章频道表

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 字段名 | 字段说明 | 类型 | 长度 | 允许空值 | 备注 |
| newsTypeId | 频道Id | int | 11 | 否 | 主键 |
| newsTypeName | 频道名 | varchar | 200 | 是 |  |
| newsTypeDescripe | 频道描述 | text |  | 是 |  |

### 4.2.6 会员留言表（暂时没用）

会员留言表leavemessage主要是包含会员留言内容，以及留言Id设为主键以便管理员可以通过留言Id来查找留言，有需要时候可进行删除管理，此表还包含有会员Id设为外键用于标识是哪位会员进行的留言。（如表4.6）

表4.6 会员留言表

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 字段名 | 字段说明 | 类型 | 长度 | 允许空值 | 备注 |
| leaveMessageId | 留言Id | int | 11 | 否 | 主键 |
| leaveMessageContent | 留言内容 | text |  | 是 |  |
| leaveMessageTime | 留言时间 | datetime |  | 是 |  |
| userId | 会员Id | int | 11 | 是 | 外键 |

### 4.2.6 新闻附件表

新闻附件表news\_attachment主要用于保存与新闻相关的附件，新闻ID作为外键，与文章信息表关联。当新闻ID为0时，表示保存的是软件信息，其中attachment\_content域为空，实际软件被保存在工程的softwares目录中。

表4.7 新闻附件表

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 字段名 | 字段说明 | 类型 | 长度 | 允许空值 | 备注 |
| attachment\_id | 附件Id | int | 10 | 否 | 主键 |
| news\_id | 新闻Id | text |  | 是 | 外键 |
| attachment\_name | 附件名称 | varchar | 50 | 是 |  |
| attachment\_content | 附件内容 | longblob |  | 是 |  |

## 4.3 数据关系ER图

ER图主要是用于显示出每个对象的属性以及每个对象之间的联系关系，对于从数据表结构到代码的实现有一定的指导作用。本次系统开发的ER图结构。（如图4.1）

 图4.1 ER图

# 代码设计

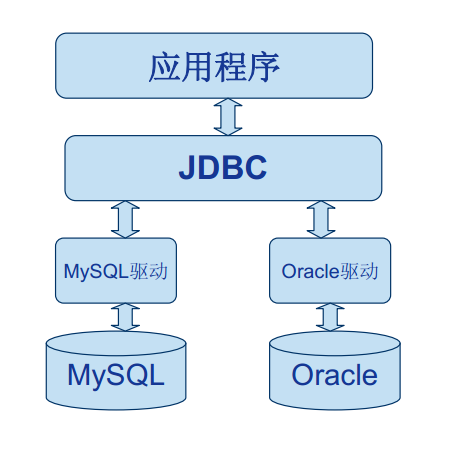
## 5.1 系统开发准备工作

### 5.1.1 JSP文件的导入

在系统开发前我们需要先倒入本项目所需要的JSP模板，这里需要说明的是在一个实际的项目开发过程中会分为负责JSP界面设计的美编，以及负责JAVA软件开发的软件工程师。在本次设计中我们主要是完成软件设计部分的功能，所以开发前需导入一个JSP文件模板。

### 5.1.2 JDBC连接数据库(这部分代码有所不同)

JDBC（Java Data Base Connectivity,java[数据库](http://baike.baidu.com/view/1088.htm)连接）是一种用于执行SQL语句的Java API，可以为多种关系数据库提供统一访问，它由一组用[Java语言](http://baike.baidu.com/view/229611.htm)编写的类和接口组成。JDBC提供了一种基准，据此可以构建更高级的工具和接口，使[数据库](http://baike.baidu.com/view/1088.htm)开发人员能够编写数据库[应用程序](http://baike.baidu.com/view/330120.htm)。（如图5.1）



如图 5.1 JDBC连接模型

由上图可知JDBC的作用是为数据库的连接提供可扩展的接口，当系统选择不同类型的数据库时，只需采用不同的驱动便可连接上不同的数据库。为系统提高更多的灵活性。

在本次系统开发前需编写一个JDBC连接池，当系统需要获取数据库数据时，只需创建一个数据库连接对象，便可与数据库获取连接，然后可以执行SQL语句对数据库进行操作。

本次设计连接池的编写代码：

**package** org.news.utils;

**import** java.sql.Connection;

**import** java.sql.DriverManager;

**import** java.sql.SQLException;

**import** java.util.LinkedList;

**public** **class** Pools {

**private** **final** **static** String *url* = "jdbc:mysql://localhost:3306/news\_all";

**private** **final** **static** String *password* = "";

**private** **final** **static** String *user* = "root";

**private** **static** Connection *con* = **null**;

**private** **static** LinkedList<Connection> *pools* = **new** LinkedList<Connection>();

**static**{

**try** {

Class.*forName*("com.mysql.jdbc.Driver");

**for**(**int** i = 1 ; i <= 2 ; i++){

*pools*.addLast(*getPoolsConnection*());

}

} **catch** (ClassNotFoundException e) {

e.printStackTrace();

}

}

**public** **static** Connection getPoolsConnection(){

**try** {

*con* = DriverManager.*getConnection*(*url*, *user*, *password*);

} **catch** (SQLException e) {

e.printStackTrace();

}

**return** *con*;

}

**public** **static** Connection getPoolsConnectionSingle(){

*con* = *pools*.removeFirst();

**return** *con* ;

}

**public** **static** **void** freeConnection(Connection con) **throws** SQLException{

*pools*.addLast(con);

}

}

## 5.2 MVC模式实现过程

在本次设计中采用的是MVC模式作为系统的搭建框架，采用这种模式的好处是可以使每一层次的作用分工明确，它强制性的使[应用程序](http://baike.baidu.com/view/330120.htm)的[输入](http://baike.baidu.com/view/700406.htm)、[处理](http://baike.baidu.com/view/989420.htm)和[输出](http://baike.baidu.com/view/700402.htm)分开。使用MVC应用程序被分成三个核心部件：[模型](http://baike.baidu.com/view/96500.htm)（model）、[视图](http://baike.baidu.com/view/71981.htm)（model）、[控制器](http://baike.baidu.com/view/122229.htm)（controller）。本文采用的是JSP+Servlet+Service+DAO四层模式来搭建整个框架。

JSP为view层，作为视图，用于在浏览器显示页面。Servlet为controller层，作为控制器，用于实现前台后台的业务数据交互以及逻辑设计。Service实现业务逻辑，为控制器提供各种服务。DAO为model层，作为模型用于从数据库获取数据，这里要指出DAO模式与传统的model层不同，它不仅承载数据，而且包含了内在的逻辑和操作。

本次设计的总体实现过程（如图：4.2）。



图4.2 实现过程

## 5.3系统功能具体实现思想（具体代码有所不同）

在系统运行时会涉及到一个通过浏览器对网页的操作，而后产生一系列的请求通过底层代码去调用所需的数据去完成对每个请求的响应的过程。在本系统中这一个过程凡是涉及到与数据库交互数据时都会按照MVC模式流程来完成。这也就使每一层都必须清楚自己所负责的功能。

在本系统的设计过程中我以先后顺序为：model，DAO，Service，Servlet，JSP的分层顺序搭建起MVC模式。然后在数据库中完成增、删、改查等功能。下面我以某些关键功能的例子来说明系统的实现思想。

### 5.3.1 对信息的批量删除

我们以文章的批量删除为例，其他的删除功能，以及单条记录的删除可以仿照此例实现。

（1）model的建立，我们首先把文章的所有所有属性封装在一个类里面，类名为newsInfo。这个类的属性是与数据库中的表的内容相匹配的，其中类中的Id可以做为该类的对象的索引，此外model中除了拥有属性外还包含了get、set方法用于获取和设置该类对象的属性。

（2）DAO层，该层主要是负责逻辑操作以及与数据库的数据交互，所以首先要创建数据库连接对象，与数据库连接。因为要实现删除功能，所以在DAO层中就定义了一个删除的方法，以批量删除为例，因为为批量删除，所以需要用数组对传来需删除的Id数据进行接收，然后返回boolean类型进行判断，实现代码如下：

**public** **boolean** deleteNewsInfo(**int**[] newsInfoIds) {

**boolean** b = **false**;

sql = "delete from newsInfo where newsInfoId = ?";

*con* = DB\_UTILS.*getConnection*();

**try** {

*pstmt* = *con*.prepareStatement(sql);

**for** (**int** i = 0; i < newsInfoIds.length; i++) {

*pstmt*.setInt(1, newsInfoIds[i]);

**int** j = *pstmt*.executeUpdate();

**if** (j > 0) {

b = **true**;

}

}

} **catch** (SQLException e) {

e.printStackTrace();

} **finally** {

DB\_UTILS.*close*(*con*, *pstmt*, *rs*);

}

**return** b;

}

1. Servlet层，本层的作用在于从前台网页获取参数再调用DAO层中所提供的方法进行逻辑操作，然后操作结果返回前台网页。需要说明的是在Servlet中存在着几个域可以存放数据与前台网页进行交流。其中常用的域有request域和session域。request域所存放的内容在一个请求过程中会一直存在，当请求结束或重定向后域中的内容会被才被释放。而session域则会在整个会话中存在只有关闭浏览器后才会消失。在本例中批量删除的实现代码如下：

public boolean deleteNewsInfoService(HttpServletRequest request,

HttpServletResponse response) throws ServletException, IOException {

String[] newsInfo = request.getParameterValues("newsInfoId");

int[] newsInfoIds = new int[newsInfo.length];

for (int i = 0; i < newsInfoIds.length; i++) {

newsInfoIds[i] = Integer.parseInt(newsInfo[i]);

}

boolean b = News\_NewsType\_Factory.instance\_News\_NewsType\_Factory()

.getNewsDAO().deleteNewsInfo(newsInfoIds);

return b;

}

1. JSP层，在这层中与Servlet层有着相同的域来存放参数和内容彼此进行交互。JSP层的作用主要是在HTML网页页面设计的基础上添加了可以实现编写JAVA代码的功能，同时也能插入javascript脚本，通过该脚本可以编写方法各种功能，在事件促发后可调用该方法完成操作。在本例中因为要将所需删除的Id传给Servlet层，所以要使用一个for循环进行遍历，将所有的遍历出来的Id以及告诉Servlet所需执行的方法的参数op形成一个字符串传递出去。实现代码如下：

**function** **dels**(){

alert("确定全部删除吗？");

//获取所有checkbox对象

**var** checkBoxs = document.getElementsByName("checkbox2");

**var** url="/news\_all/newsService.action?op=deleteNewsInfo";

//获取被选中的checkbox

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

**if**(checkBoxs[i].checked){

url = url+"&newsInfoId="+checkBoxs[i].value;

}**else**{

checkBoxs[i].checked = **false**;

}

}

window.location = url;

}

如此一来，一个批量删除的功能就能通过前台网页JSP页面获取删除操作经过Servlet层、DAO层对数据库中的文章进行删除。

### 5.3.2 对信息的修改

我们以文章的修改为例，其他的修改、添加、注册、登陆功能的实现可以仿照该例来实现。需要说明的是添加和注册操作需要用到SQL语言的插入操作，登陆用到SQL语言的查找验证操作。而修改需要先从数据库中查找出来再进行插入更新。所以选用文章修改为例供大家参考。

（1）model的建立，此处的model其实与上一例中的文章删除所建的model相同。

（2）DAO层，因为要实现修改功能，所以在DAO层中就定义了查询文章信息的方法，以及一个插入文章信息的方法，同样首先要创建数据库连接对象，与数据库连接。实现代码如下：

// 通过新闻Id查询文章内容

**public** NewsInfo searchNewsInfo(**int** newsInfoId) {

sql = "select \* from newsInfo where newsInfoId = ?";

*con* = DB\_UTILS.*getConnection*();

NewsInfo newsInfo = **null**;

**try** {

*pstmt* = *con*.prepareStatement(sql);

*pstmt*.setInt(1, newsInfoId);

*rs* = *pstmt*.executeQuery();

**if** (*rs*.next()) {

String newsInfoTitle = *rs*.getString("newsInfoTitle");

String newsInfoDescribe = *rs*.getString("newsInfoDescribe");

String newsInfoTime = *rs*.getString("newsInfoTime");

String newsAuthor = *rs*.getString("newsAuthor");

String newsInfoContent = *rs*.getString("newsInfoContent");

**int** adminId = *rs*.getInt("adminId");

**int** newsTypeId = *rs*.getInt("newsTypeId");

**int** newsInfoState = *rs*.getInt("newsInfoState");

newsInfo = **new** NewsInfo(newsInfoId, newsInfoTitle,

newsInfoDescribe, newsInfoContent, newsInfoTime,

newsAuthor, adminId, newsTypeId, newsInfoState);

}

} **catch** (SQLException e) {

e.printStackTrace();

} **finally** {

DB\_UTILS.*close*(*con*, *pstmt*, *rs*);

}

**return** newsInfo;

}

// 修改文章信息

**public** NewsInfo updateNewsInformation(NewsInfo newsInfo) {

*con* = DB\_UTILS.*getConnection*();

sql = "update newsInfo set newsInfoTitle = ? ,newsInfoDescribe = ?,newsInfoContent = ?,"

+ "newsInfoTime = ?,newsAuthor = ?,adminId = ? , newsTypeId = ? ,newsInfoState = ? where newsInfoId = ?";

**try** {

*pstmt* = *con*.prepareStatement(sql);

String newsInfoTitle = newsInfo.getNewsInfoTitle();

String newsInfoDescribe = newsInfo.getNewsInfoDescribe();

String newsInfoContent = newsInfo.getNewsInfoContent();

String newsInfoTime = newsInfo.getNewsInfoTime();

String newsAuthor = newsInfo.getNewsAuthor();

**int** adminId = newsInfo.getAdminId();

**int** newsTypeId = newsInfo.getNewsTypeId();

**int** newsInfoId = newsInfo.getNewsInfoId();

**int** newsInfoState = newsInfo.getNewsInfoState();

*pstmt*.setString(1, newsInfoTitle);

*pstmt*.setString(2, newsInfoDescribe);

*pstmt*.setString(3, newsInfoContent);

*pstmt*.setString(4, newsInfoTime);

*pstmt*.setString(5, newsAuthor);

*pstmt*.setInt(6, adminId);

*pstmt*.setInt(7, newsTypeId);

*pstmt*.setInt(8, newsInfoState);

*pstmt*.setInt(9, newsInfoId);

**int** i = *pstmt*.executeUpdate();

**if** (i == 0) {

newsInfo = **null**;

}

} **catch** (SQLException e) {

e.printStackTrace();

} **finally** {

DB\_UTILS.*close*(*con*, *pstmt*, *rs*);

}

**return** newsInfo;

}

（3）Servlet/JSP层，同样是通过域中传过来的参数获取操作数，然后调用DAO的方法，完成操作回复响应。

如此一来，一个文章修改的功能就能通过前台网页JSP页面获取删除操作经过Servlet层、DAO层对数据库中的文章进行修改。

# 6. 系统调试

## 6.1前台页面调试

新闻发布系统有两种用户：会员用户（包含游客）和管理员，分别有不同的权限，但都是从同一界面登陆，此系统首页采用DIV+CSS框架，使得页面看起来美观舒适。能自动获取系统当前时间，展示新闻列表内容和软件等信息。

用户只要在浏览器地址栏输入改项目的地址，即可访问该系统的相关信息；需要说明的是：因为该系统并未真正发布成正规的网站，只是将项目部署到了Tomcat服务器上面；其他机子需要访问必须在同一个局域网中访问，在浏览器地址栏中输入：http://服务器机子地址：8080/MVCNews/ 既可访问项目。其中登录URL为8080/ MVCNews /login.jsp，还可以进行注册。

登录时进行验证，用户名以字母开头，长度在6到18之间；密码长度在6到18之间。验证失败弹出提示框。



图6.1 前台会员登录



图6.2 登录验证



图6.3登录成功



图6.4 登录失败

在登陆界面下方有个链接可以进行注册，注册界面如下图所示。在填写时，系统会自动对每一项进行验证。

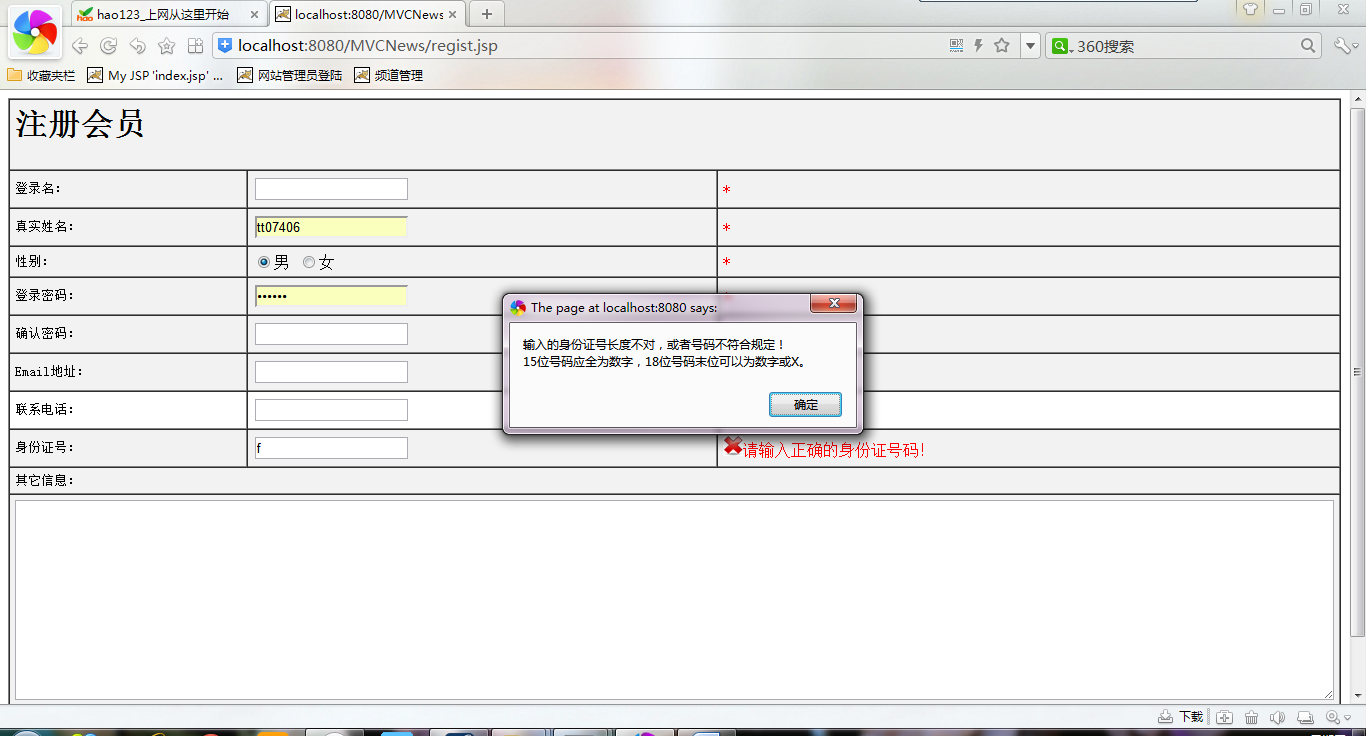


图6.5 会员注册页面

## 6.2 后台管理员页面调试



图6.6 后台登录

后台登录界面地址为8080/MVCNews/manage/login.jsp。在该界面输入用户名密码后，完成登录验证后就能进入管理员操作界面（如图6.7）。



图6.7 管理员操作界面

在管理员界面中点击左边栏添加新闻操作可以跳转到添加新闻页面（如图6.8）可以添加新闻，成功后可以在将新闻发布到前台页面，可供用户访问。可以上传多个附件，只需点击右边的“添加”按钮。



图6.8 添加新闻

点击新闻列表便可以查看所有的新闻，还可以进行新闻的查找和翻页操作。点击标题链接可以查看新闻详情，同样的，修改和删除链接可以实现相应的操作。

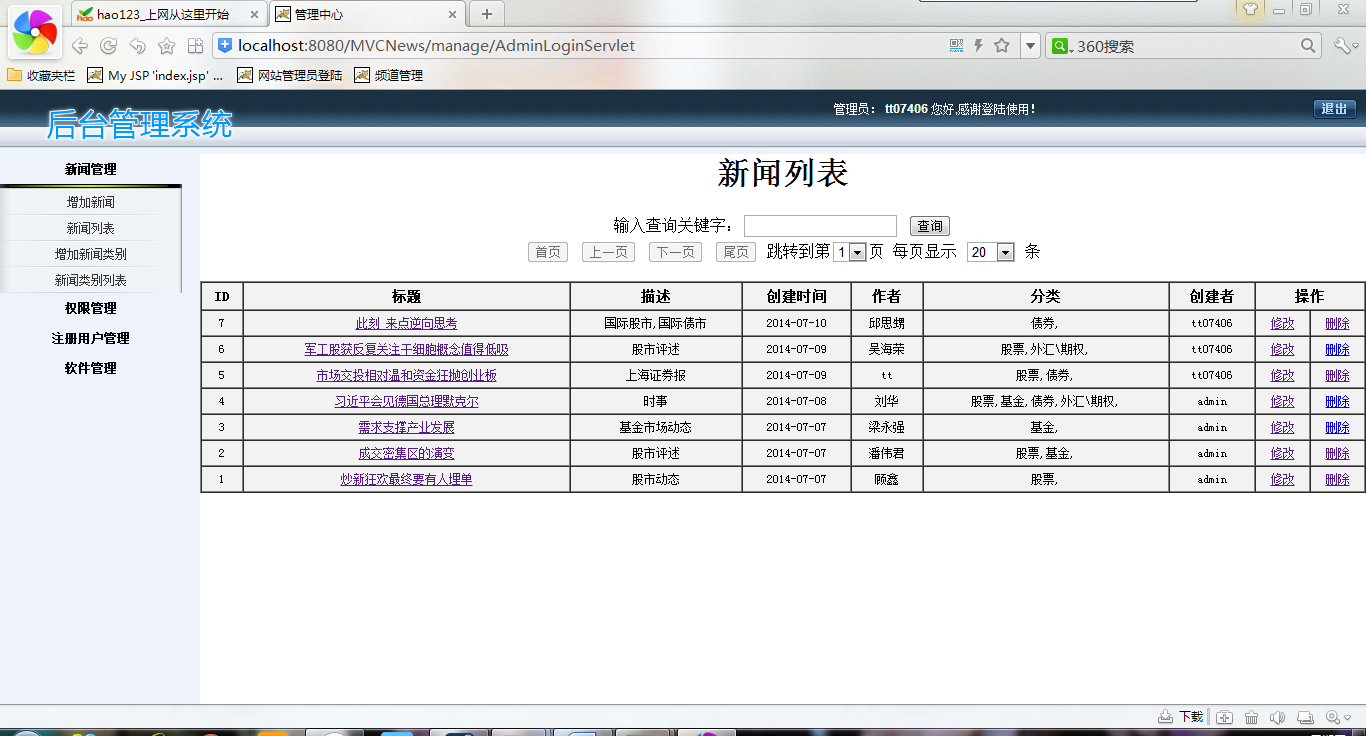


图6.9 新闻列表

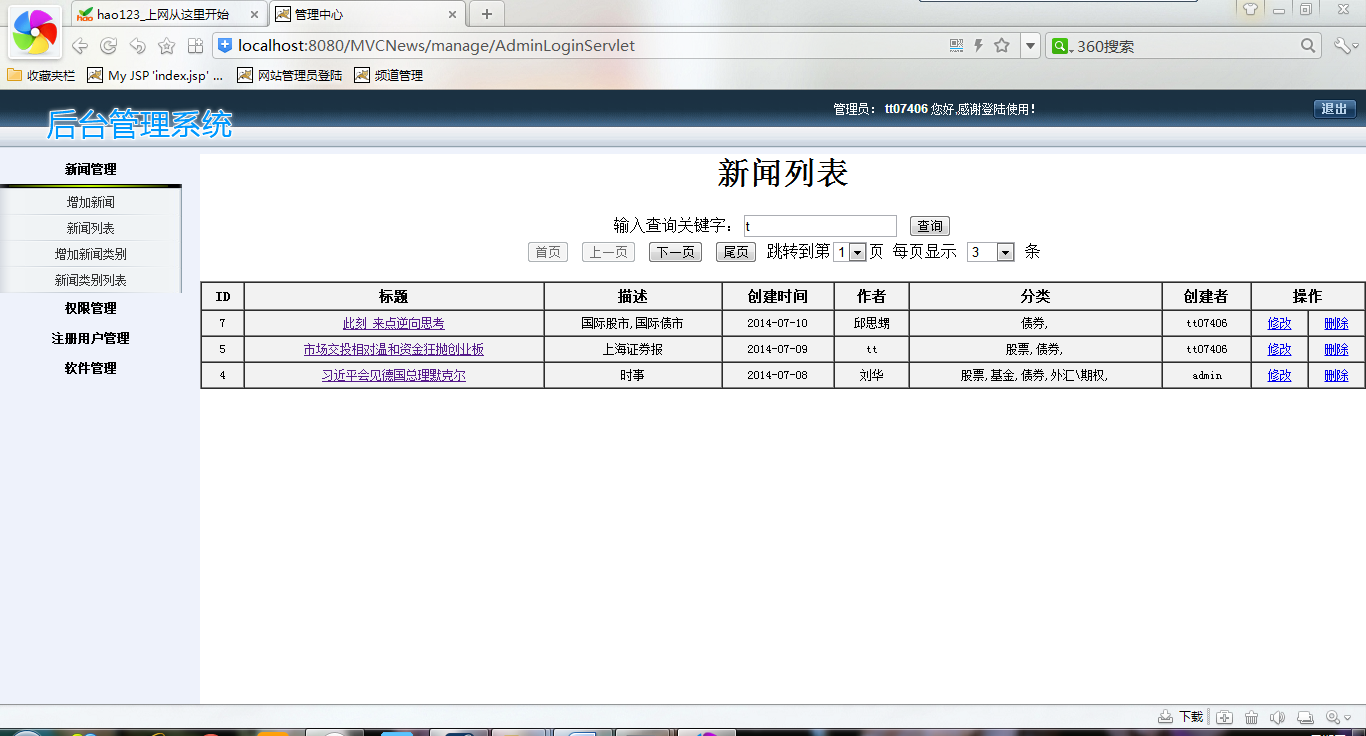


图6.10 翻页和查找

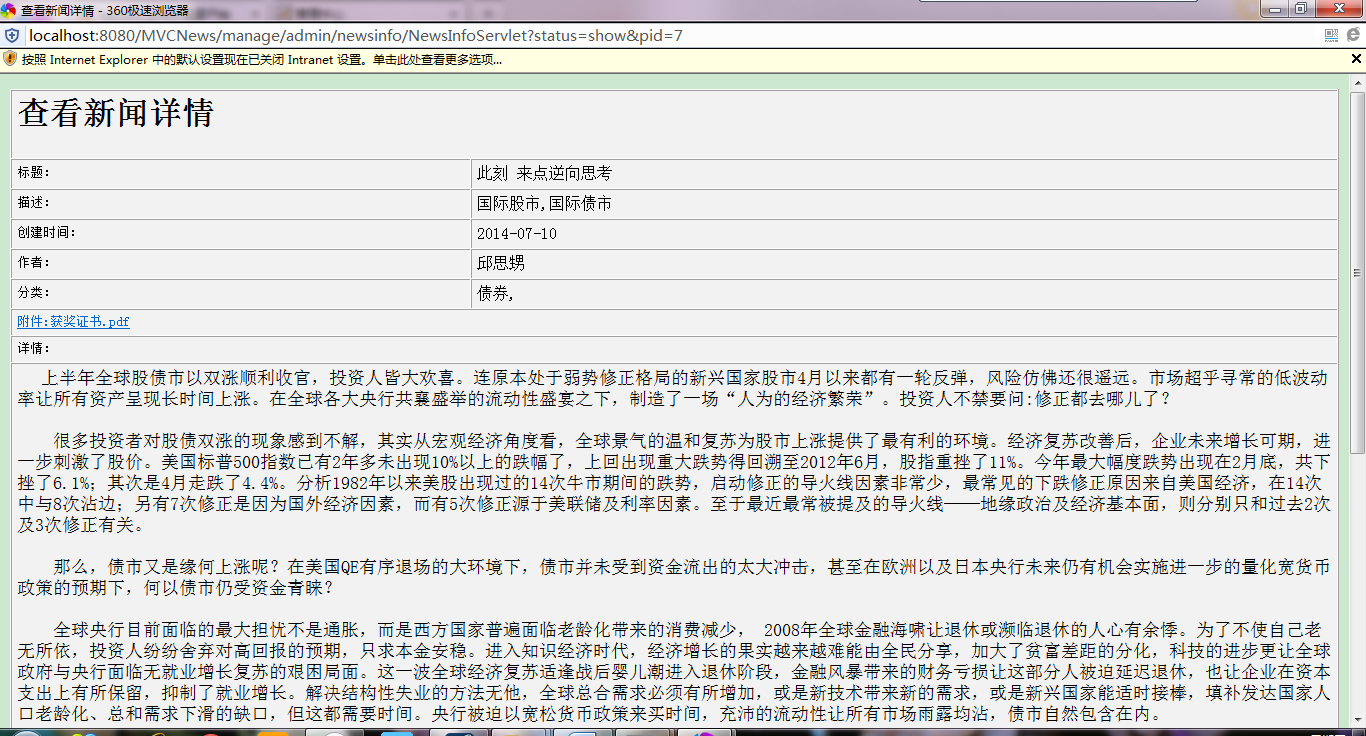


图6.11 新闻详情

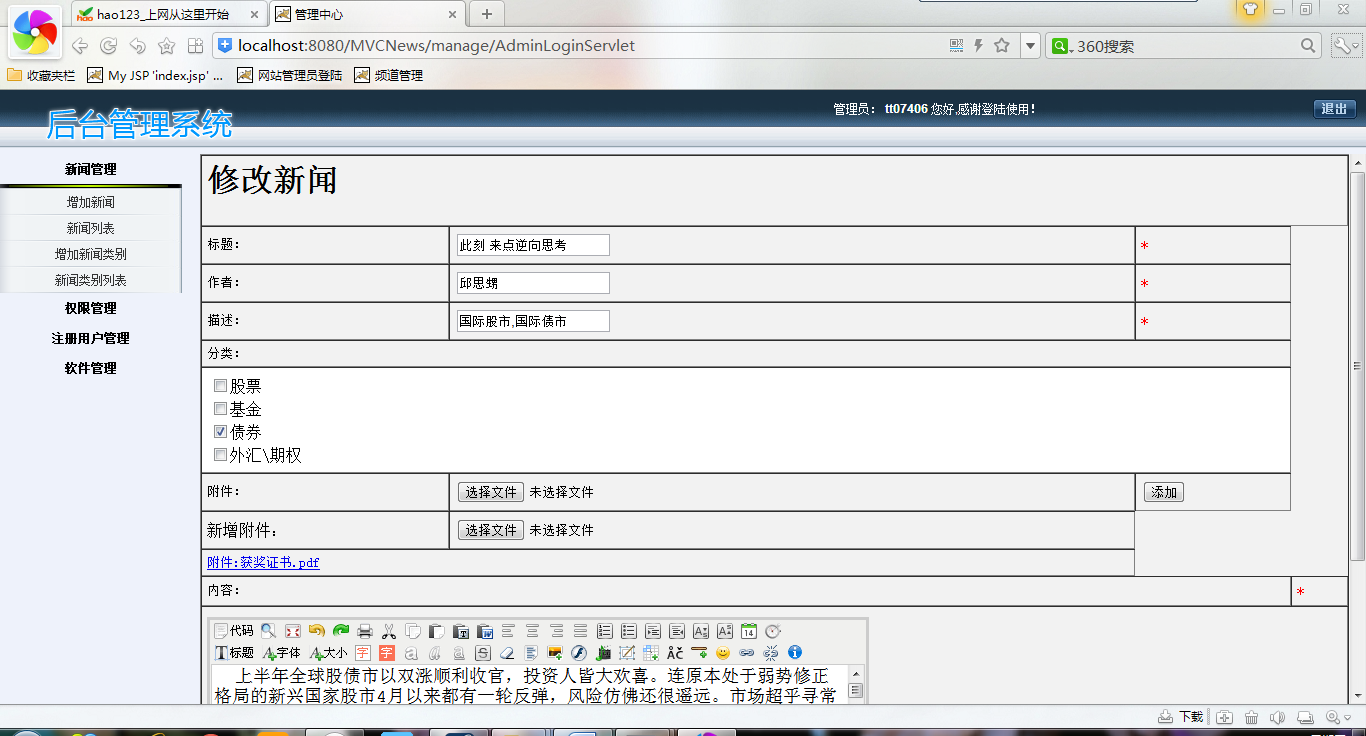
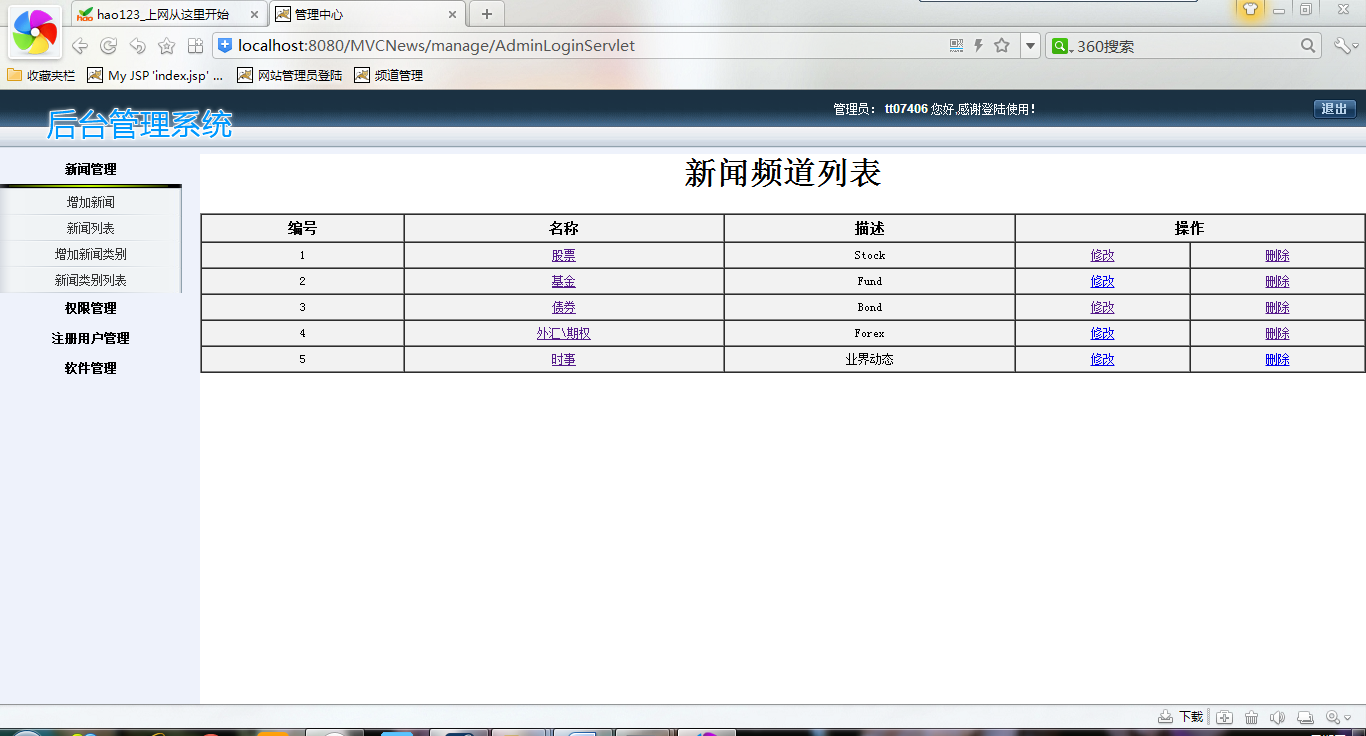


图6.12 修改新闻信息

同样的，可以对新闻类别，管理员进行相应的操作，如下图所示。其中，新闻类别与新闻中的分类项一致，不建议随意增改，修改时也不能修改名称。









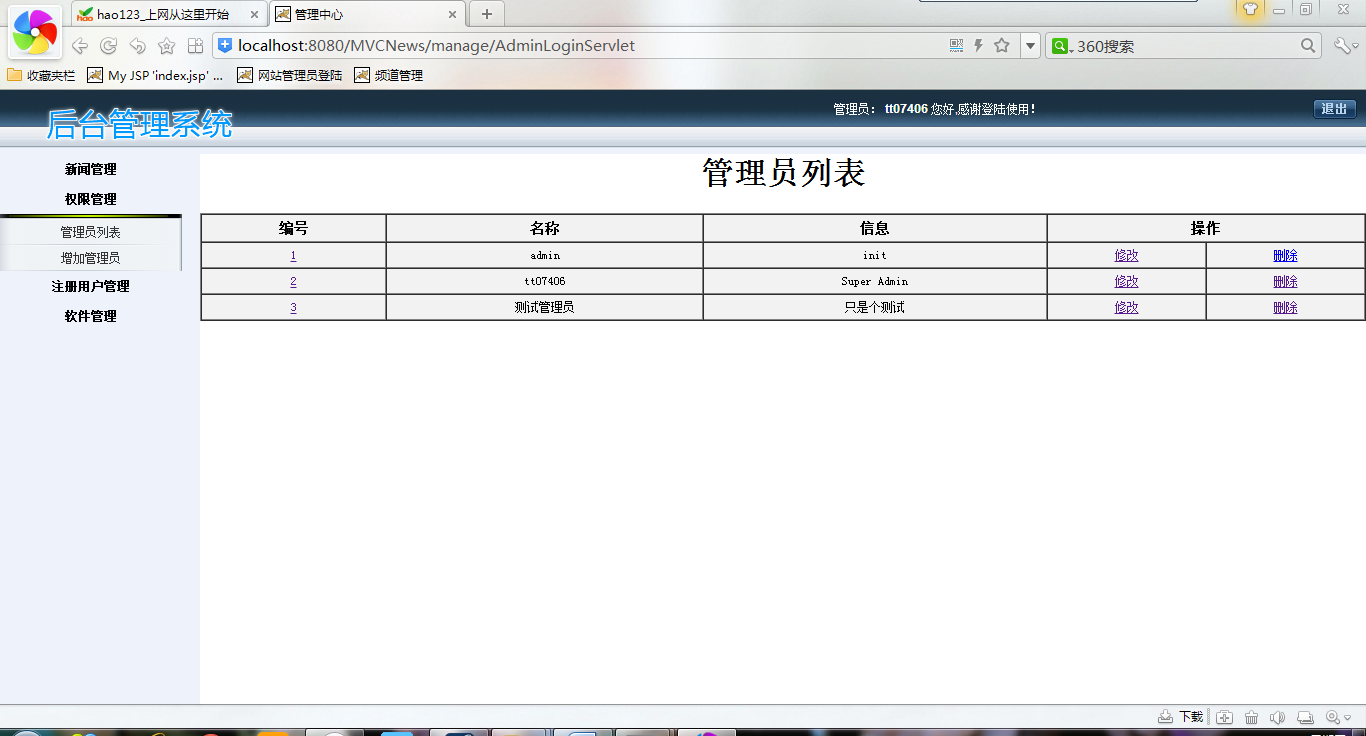


图6.13 新闻类别和管理员操作

对于会员管理，目前只支持展示会员详情、修改会员信息和删除会员操作。

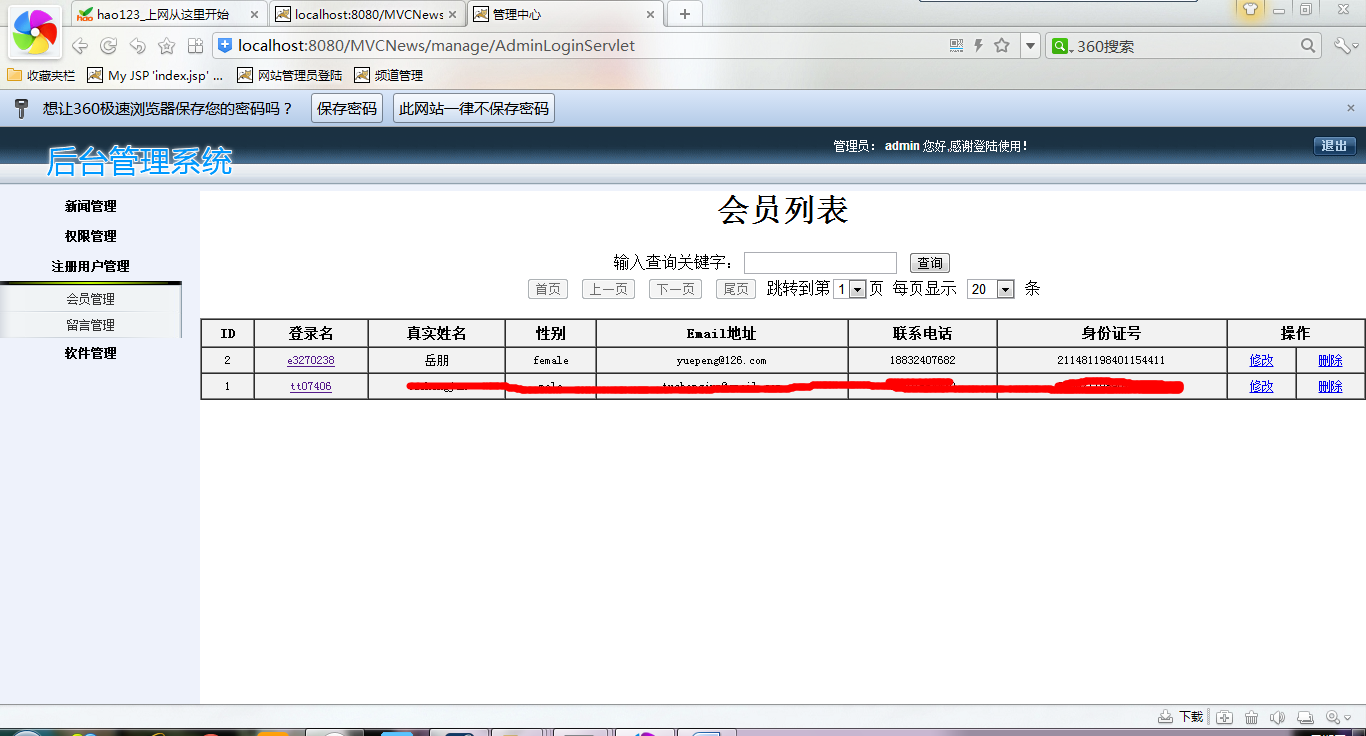


图6.14 会员模块

本系统还可以批量上传软件，软件保存在应用服务器，可以点击链接下载或删除。



图6.15 批量上传软件



图6.16 软件列表



图6.17 软件下载

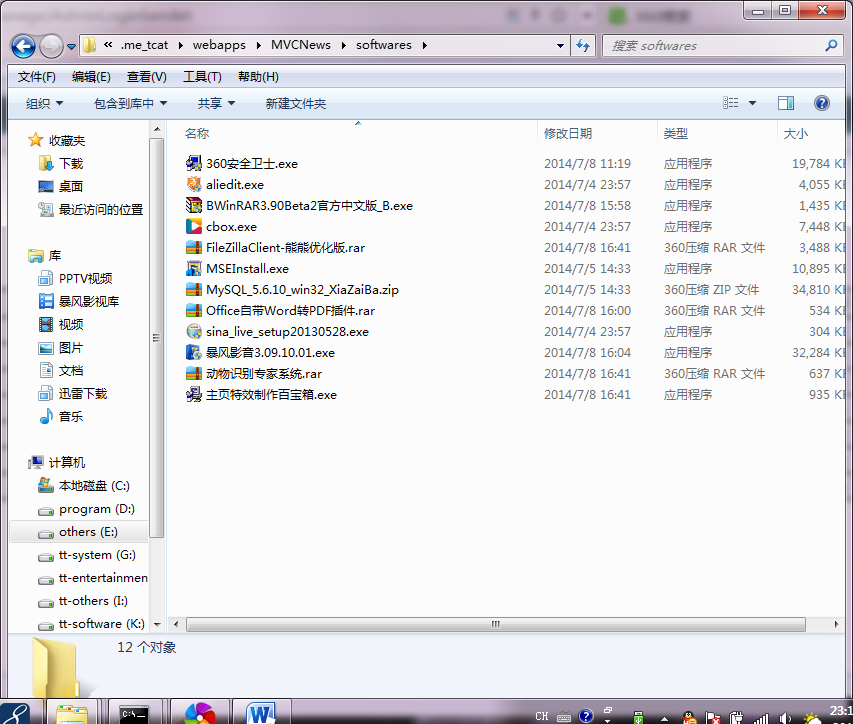


图6.18 软件所在目录