

# **软件技术专业网站**

## **总体方案设计和功能实现报告**

2013 年 12 月 15 日

# 目 录

一、引言.....	1
1.1 编写目的 .....	1
1.2 适用范围 .....	1
1.3 文档概述 .....	1
1.4 参考资料 .....	1
1.5 术语和缩写词 .....	2
二、网站总体概述.....	2
2.1 建设目标 .....	2
2.2 功能需求 .....	2
三、网站设计方案.....	3
3.1 网站设计原则 .....	3
3.2 网站整体设计 .....	4
3.2.1 网站技术平台.....	4
3.2.2 网站用例模型描述.....	4
3.2.3 网站数据库设计 .....	5
3.3 网站安全措施 .....	8
四、网站详细设计.....	8
4.1 网站前台详细设计 .....	8
4.2 网站后台详细设计 .....	9
五、 网站功能实现.....	10
5.1 采用 ASP.NET 三层架构体系实现网站功能.....	12
5.2 三层架构体系中数据访问及调用 .....	12
5.3 三层结构中核心类的定义及说明 .....	13
5.3.1 数据库访问组件基础类（DBUtility） .....	13
5.3.2 业务是实体类（Model） .....	15
5.3.3 数据访问层（DAL） .....	16
5.3.4 业务逻辑层（BLL） .....	21
5.3.5 表示层（WEB） .....	24
六、运行环境.....	30
6.1 硬件平台 .....	30
6.2 网络平台 .....	30
6.3 软件平台 .....	30
七、工作任务及进度计划 .....	30
7.1 项目组成员.....	30
7.2 项目进度计划 .....	31
八、网站推广和维护 .....	31

# 一、引言

## 1.1 编写目的

本报告为软件技术专业网站项目（以下简称专业网站）的总体方案设计报告，目的是形成软件开发者和任务提出者对系统需求理解达成一致共识的基础文件，作为界定项目范围基础，也作为本项目测试验收的主要依据。同时，本文档也作为软件开发人员、测试人员进行后继工作开展的基础，供项目负责人、技术开发人员、测试人员等理解需求之用。

## 1.2 适用范围

本文档适用于所有与本项目有关的软件开发阶段及其相关人员，其中：项目负责人、技术开发人员、测试人员应重点阅读本文档各部分，其他人员可选择性阅读本文档。

## 1.3 文档概述

本文档主要描述了专业网站的总体设计方案及功能实现。

本文档首先从建设目标、网站需求及功能等方面概要描述系统，其次从设计原则、数据结构设计、功能设计、运行环境等方面描述系统的总体设计情况，然后进一步详细描述了网站采用的技术策略、功能实现及项目实施。

## 1.4 参考资料

《设计模式--可复用面向对象软件的基础》，伽玛等著，机械工业出版社，2005-6

《UML 用户指南》第 2 版，（美）布奇，人民邮电出版社 2006-6

《ASP.NET Web 开发教程》，程不功，清华大学出版社 2011-09

《ASP.NET MVC 4 高级编程》（美）加洛韦，清华大学出版社 2013-8-1

《ASP.NET 开发实战范例宝典》章立民，科学出版社 2010-10-1

## 1.5 术语和缩写词

表 1 术语和缩写词

序号	术语	说明性定义
1	SQL	一种用于访问查询数据库的语言
2	SqlCommand	数据库命令对象
3	SqlConnection	数据源连接对象
4	SqlDataAdapter	数据适配器对象
5	DBUtility	数据库访问组件基础类
6	Model	数据实体类
7	DAL	数据访问层
8	BLL	业务逻辑层
9	WEB	显示层
10	三层架构体系	整个业务应用划分为：表现层（WEB）、业务逻辑层（BLL）、数据访问层（DAL）

## 二、网站总体概述

### 2.1 建设目标

“软件技术专业网站”，作为学校专业网站之一，应能达到以下目标：

1. 具有宣传软件技术专业的功能
2. 具有服务于在校生的功能

### 2.2 功能需求

1. 通过网站中专业简介、展示专业人才培养目标、就业方向、合作机构、合作模式等相关内容，达到对外宣传软件技术专业的目的；
2. 通过行业动态让同学们了解到本行业的最新信息。
3. 通过在线学习为在校生提供在线学习功能；
4. 通过素质拓展栏目展示在校生良好的精神风貌；
5. 通过新闻中心了解本专业的最新动态及最新技术；
6. 通过求职面试，就业准备为在校生提供最新的本行业内就业信息；
7. 通过毕业生风采展示栏目为本专业学生提供优秀毕业生榜样力量。

## 三、网站设计方案

### 3.1 网站设计原则

专业网站是展示本专业建设的一个重要窗口，针对整个网站进行精心的形象设计，使之在视觉效果上更美观、大方、简捷，内容上更加完整、实用、丰富、统一。主要设计原则如下：

1. 网站色彩：以黑白色为主，配合中国风的特色。
2. 网站规格：网站采用 1024\*768。
3. 内容与形式相统一：网站设计过程中力求通过网页内容向浏览者传达的有效信息及文字，综合运用网站的排版布局、色彩、图形等增强网站外在的视觉效果。
4. 主题鲜明：本网站的主题就是通过网站宣传，树立专业形象，为专业的招生就业服务，为在校生服务。设计过程中力求围绕主题，突出重点，突出自己的个性和特色。
5. 风格统一：整个网站的设计采取统一的风格，使网站看起来更专业。确立风格时，力争突出自身的个性，无论是文字、色彩的运用，还是版式的设计都要给人一种鲜明的印象。
6. 导航清晰：网站给浏览者提供一个清晰的导航系统，以便于浏览者能够清楚目前所处的位置，同时能够方便地转到其他页面。导航系统出现在每一个页面上，标志明显，便于用户使用。
7. 栏目设置合理：栏目设置是否清晰、合理、科学，往往在很大程度上影响网站的访问量。本专业网站主要客户群是：即将填报志愿的学生、在校学生和已毕业学生。学生的计算机水平参差不齐，栏目设置合理的网站，使学生能很容易地找到需要的东西，这样的网站才能让学生喜欢。
8. 良好的兼容性：对于网页来说，它不同于其它印刷品，制作完成后就一成不变了，它随着用户浏览器的不同而出现变化，因此在设计过程中注意网页的兼容性，使它适用于大多数主流的浏览器，不致于出现差别很大的浏览效果。
9. 经常更新：本网站自建成后保持经常维护更新，给学生提供最新的信息，增强网站的吸引力。

## 3.2 网站整体设计

### 3.2.1 网站技术平台

#### 1. Asp.net 技术

ASP.NET 是由微软在 .NET Framework 框架中所提供，开发 Web 应用程序的类库，封装在 System.Web.dll 文件中，并提供 ASP.NET 网页处理、扩充以及 HTTP 通道的应用程序与通信处理等工作，以及 Web Service 的基础架构。

ASP.NET 优点主要表现在：可管理性、安全、易于部署、增强的性能、灵活的输出缓存、国际化、移动设备支持、扩展性和可用性、跟踪和调试、与 .NET Framework 集成和与现有 ASP 应用程序的兼容性。

#### 2. 三层架构体系

网站整体采用 Asp.net 的三层架构(3-tier application)进行设计，通常意义上的三层架构就是将整个业务应用划分为：表现层（web）、业务逻辑层（BLL）、数据访问层（DAL）。区分层次的目的即为了“高内聚，低耦合”目的。

### 3.2.2 网站用例模型描述

在明确网站建设目标和功能需求基础上，对系统需求进行细化，为更深入的系统设计打好的基础。通过分析和分解系统建设和功能目标中关键操作，下面使用 UML 先对网站角色进行建模。

#### 1. 网站角色用例模型

网站角色分为超级管理员、内容管理员。超级管理员具有所有权限；内容管理员角色，主要完成各个栏目内容的添加与修改的功能。用户角色用例图如图 1 所示：

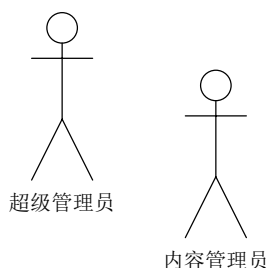


图 1 用户角色用例图

## 2. 网站超级管理员角色用例模型

网站超级管理员具有所有权限，能完成后台管理的所有功能, 角色用例图如图 2 所示：

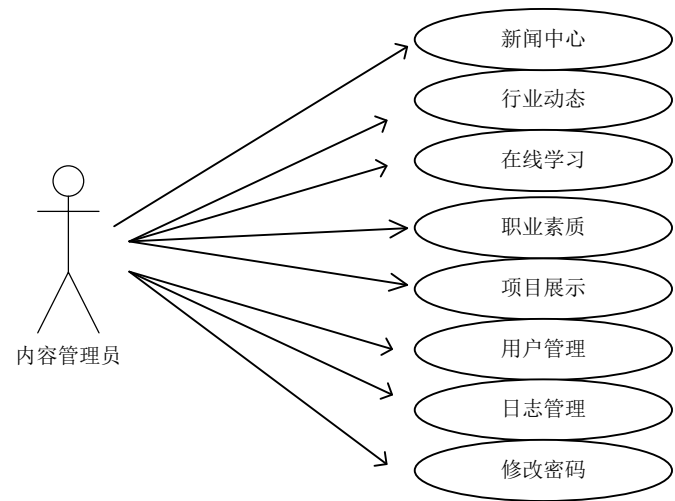


图 2 超级管理员角色用例图

## 3. 网站内容管理员角色用例模型

内容管理员角色，主要完成各个栏目内容的添加与修改的功能，其角色用例图如图 3 所示：

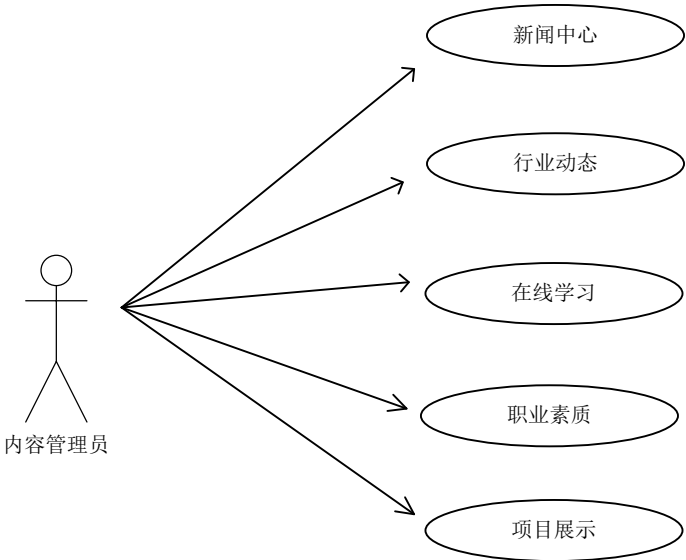


图 3 内容管理员角色用例图

### 3.2.3 网站数据库设计

数据库设计是系统设计中必不可少的组成部分，也是进行系统开发的基础。为完成本网站的功能，设计数据库 MyDB 数据库，在本数据库中一共有 5 个数据表，每个数据表的名称及作用如表 2 所示。

表 2 数据库中数据表简介

序号	表名	作用
1	hnf_adminuser (管理员表)	每一条记录代表一个不同角色的管理员，角色分为四类，一个是高级管理员角色，具有所有的权限，能完成后台管理的所有功能；一个是添加内容管理员角色，主要完成各个栏目内容的添加与修改的功能；一个是学习资料管理员角色，主要用于完成学生学习资料的管理；一个是统计人员角色，主要完成当月内容添加的统计功能。
2	hnf_baseinfo (基本信息表)	用于保存本网站基本信息，如单位名称、单位地址、单位联系电话，网站备案号。
3	hnf_news (新闻表)	主要用于存放在前台以文字为主题内容展示的栏目，同时也可以包含图片的内容。
4	hnf_n_type (新闻类型表)	因为“hnf_news”中存放多种栏目的内容，本表主要用于表达多种栏目对应的类型，将来通过“hnf_news”与“hnf_n_type”关联能够清楚的查询出每一个栏目对应的记录数及记录的详细内容。
5	hnf_loginlog (登录日志表)	用于保存任何时刻，任意一个角色登陆后台的操作日志信息。

具体表结构如表 2-1 至表 2-12 所示：

表 2-1：管理员表

表名	序号	列名	数据类型	长度	标识	主键	允许空	默认值	说明
hnf_adminuser	1	user_id	int		是	是	否		主键
	2	user_name	varchar	50			是		用户名
	3	user_password	varchar	50			是		密码
	4	user_createtime	datetime				是		创建时间
	5	user_ip	varchar	50			是		登录 ip
	6	user_logintime	datetime				是		最后登录时间
	7	user_role	int						角色

表 2-2 基本信息表

表名	序号	列名	数据类型	长度	标识	主键	允许空	默认值	说明
hnf_baseinfo	1	base_id	int		是	是	否		主键
	2	base_dianhua	varchar	20			是		电话
	3	base_mobile	varchar	20			是		手机
	4	base_qq	varchar	20			是		QQ
	5	base_email	varchar	20			是		Email
	6	base_lianxiren	varchar	50			是		联系人
	7	base_address	varchar	100			是		地址
	8	base_seotitle	varchar	50			是		网站标题



	9	base_keyword	varchar	100			是		网站关键字
	10	base_describ	varchar	100			是		网站描述
	11	base_beianhao	varchar	50			是		备案号
	12	base_content	varchar	0			是		简介

表 2-8 新闻表

表名	序号	列名	数据类型	长度	标识	主键	允许空	默认值	说明
hnf_news	1	news_id	int		是	是	否		主键
	2	news_seotitle	varchar	50			是		优化标题
	3	news_keyword	varchar	100			是		关键字
	4	news_describ	varchar	100			是		优化描述
	5	news_name	varchar	50			是		新闻标题
	6	news_content	varchar	0			是		内容
	7	news_createtime	datetime				是		创建时间
	8	news_alertime	datetime				是		修改时间
	9	news_comefrom	varchar	50			是		来源
	10	news_seetime	int				是	0	点击次数
	11	news_isshow	int				是		审核
	12	news_recommand	int				是		推荐
	13	news_type	int				是		类型 id
	14	news_image	varchar	50			是		图片
	15	news_isimage	int				是	0	是否图片新闻
	16	news_sort	int				是		排序

表 2-9 新闻类型表

表名	序号	列名	数据类型	长度	标识	主键	允许空	默认值	说明
hnf_n_type	1	n_type_id	int		是	是	否		主键
	2	n_type_name	varchar	255			是		类型名称
	3	n_type_parentid	int				是		父类 id
	4	n_type_strsort	varchar	50			是		排序数组
	5	n_type_depth	int				是		深度
	6	n_type_root	int				是		节点
	7	n_type_sort	int				是		排序
	8	n_type_isshow	int				是		是否显示

表 2-11 登录日志表

表名	序号	列名	数据类型	长度	标识	主键	允许空	默认值	说明
hnf_loginlog	1	log_id	int		是	是	否		自动编号
	2	log_type	int				是	0	类型
	3	log_note	varchar	200			是		内容
	4	log_ip	varchar	50			是		登录 IP
	5	log_time	datetime				是		登录时间

### 3.3 网站安全措施

网络安全有网络中心的通过防火墙配置实现。

服务器安全，开启自动更新，开启防火墙，封掉不必要的端口，安装杀毒软件。

网站安全，前后分别发布，前台访问数据库只给只读访问权限，后台访问数据库给读取和写入权限。并且在做的时候考虑网站的页面安全，包括错误信息进行处理，防 sql 注入等。

## 四、网站详细设计

本网站主要是由“网站前台”子模块和“网站后台”子模块两部分组成，其系统框架如图 4 所示：

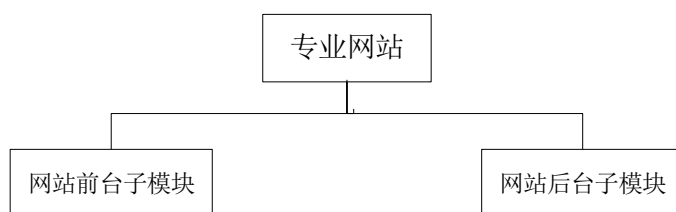


图 4 系统框架图

### 4.1 网站前台详细设计

网站前台设计力求视觉效果上更美观、大方、简捷，内容上完整、实用，功能上为在校生提供一个展示的平台、一个服务的平台以及对校外提供一个专业自我宣传的平台、一个信息资源供给的平台，其功能模块图如图 5 所示：

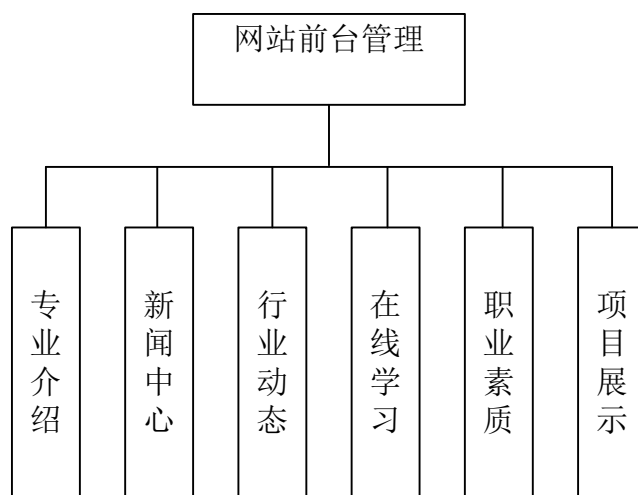


图5 “网站前台子模块”功能模块图

各个模块具体功能描述如下：

专业简介：专业培养目标、就业方向、专业特色等内容。以文字的方式展示。

新闻中心：及时报道本专业的新闻动态。以文字和图片的方式展示并且信息目录按照时间倒序显示，使浏览者看到的永远都是网站上的最新信息。（动态内容可时时更新，修改）

行业动态：对国内、国际的行业资讯进行及时的报道，采用文字的方式展示，并且信息目录按照时间倒序显示，使浏览者看到的永远都是网站上的最新信息。（动态内容可时时更新，修改）

在线学习：该栏目展示了学生的学习心得，使浏览的学生学到新的学习方法，增强学习的动力。（动态内容可时时更新，修改）

职业素质：该栏目展示了软件人员应该具备的职业素质和道德修养。（动态内容可时时更新，修改）

项目展示：部分学生的优秀作品展示，图片方式显示并且信息目录按照时间倒序显示，使浏览者看到的永远都是网站上的最新信息。（动态内容可时时更新，修改）

## 4.2 网站后台详细设计

网站后台在界面设计上力求简洁大方，网站后台栏目与网站前台栏目一一对应，保证信息管理的准确与便利；在功能上力求信息的添加、删除和编辑操作简单，其功能模块图如图6所示：

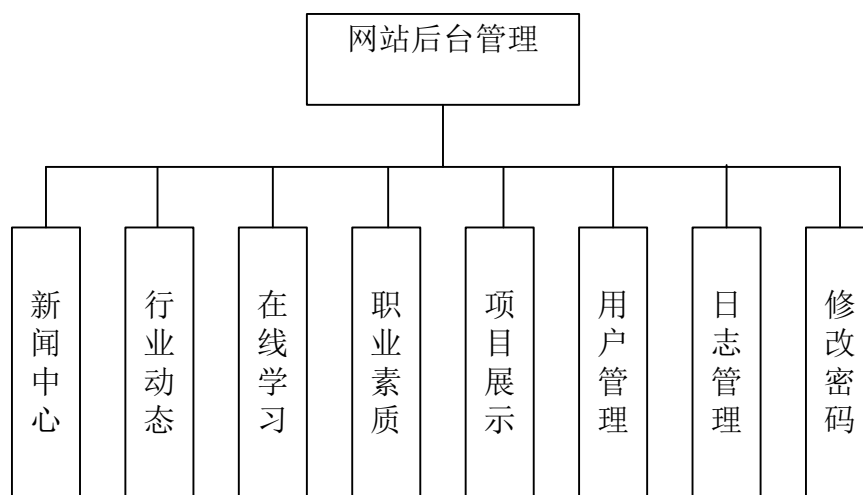


图6 “网站后台子模块”功能模块图

各个功能模块具体描述如下：

行业动态、新闻中心、专业简介、在线学习等功能模块：主要用于存放在前台以文字为主题内容展示的栏目，同时也可以包含图片的内容，可以动态添加与修改记录的内容。

项目展示模块：主要用于上传图片类的内容，前台以图片为主题内容的栏目，同时也包含文字内容。

管理员：管理员管理模块将角色分为两类，一个是超级管理员角色，具有所有的权限，能完成后台管理的所有功能；一个是内容管理员角色，主要完成各个栏目内容的添加与修改的功能。

## 五、 网站功能实现

网站前台实现了“行业动态”、“新闻中心”、“专业简介”、“在线学习”、“职业素质”、“项目展示”等栏目的展示功能，界面如图7所示。



图 7 网站前台页面

网站后台实现了与前台栏目一一对应的后台管理模块。每个管理模块通过 web 页面对相应的信息进行添加、修改与删除等操作。并且具有用户权限设置与维护，日志跟踪等功能。后台管理界面如图 8 所示。



图 8 后台管理界面

## 5.1 采用 ASP.NET 三层架构体系实现网站功能

网站整体采用 Asp.net 的三层架构体系(3-tier application)进行设计，本网站是再标准的三层架构体系中添加了 Model、DBUtility 两层，实际都是在这三层基础上的一种扩展和应用。因此整个网站业务功能划分为：表现层(web)、业务逻辑层(BLL)、数据访问层(DAL)、业务实体类(Model)、数据访问组件基础类(DBUtility) 5 个层次。

## 5.2 三层架构体系中数据访问及调用

数据访问层(DAL)只能被业务逻辑层(BLL)访问，业务逻辑层只能被表示层(web)访问，用户通过表示层将请求传送给业务逻辑层，业务逻辑层完成相关业务规则和逻辑，并通过数据访问层访问数据库获得数据，然后按照相反的顺序依次返回将数据显示在表示层。三层架构间数据访问及调用关系如表 3 所示。

表 3 三层架构间数据访问及调用关系

序号	项目	描述	用途	项目引用关系
1	Web	表现层	Web 页和控件	引用 BLL, Model

2	BLL	业务逻辑层	业务逻辑组件	引用 DAL, Model, 使用 DAL 创建实例
3	DAL	数据访问层	创建反射, 用来确定加载哪一个数据库访问程序集的类。每个 DAL 中的类都要实现的一组接口	引用 Model 、DBUtility , 通过读取 web.config 里设置的程序集, 加载类的实例, 返回给 BLL 使用。
4	Model	业务实体类	传递各种数据的容器	无引用
5	DBUtility	数据库访问组件基础类	GetSqlConnectionString 得到数据库连接字符, 也可以在 SQLServerDAL.SqlHelper, 或 oledbhelp 中用 static readonly string SqlConnectionString 代替。	无引用

## 5.3 三层结构中核心类的定义及说明

web 页面中看到的大量的数据信息是用户通过表示层将请求传送给业务逻辑层, 业务逻辑层完成相关业务规则和逻辑, 并通过数据访问层访问数据库获得数据, 然后按照相反的顺序依次返回将数据显示在表示层。代码的实现过程与用户的使用过程是一个相反的过程, 即数据库访问组件类 (DBUtility)、业务实体类 (Model)、数据访问层 (DAL)、业务逻辑层 (BLL)、表示层 (WEB) 的一个顺序实现的。以下详细介绍整个三层架构体系中核心类的实现方法。

### 5.3.1 数据库访问组件基础类 (DBUtility)

数据库访问组件基础类 (DBUtility) 中有一个属性, 8 种方法。详细内容如下。

#### 1. 属性描述:

(1) 属性 connectionString: 获取数据库连接字符串, 其属于静态变量且只读, 项目中所有文档可以直接使用, 但不能修改。

类型: string

约束: public static readonly

#### 2. 方法描述:

(1) 方法: ExecuteSql () 执行一个不需要返回值的 SqlCommand 命令, 通过指定专用的连接数据库字符串 connectionString 连接数据库。

方法参数: string SQLString //一个 SQL 语句, 如 insert、delete、update 语句的字符串

方法返回值: int //返回一个数值表示此 SqlCommand 命令执行后影响的行数

(2) 方法: ExecuteSql (): 执行一个不需要返回值的 SqlCommand 命令, 通过指定专用的连接数据库字符串 connectionString 连接数据库。

方法参数: string SQLString //一个 SQL 语句, 如 insert、delete、update 语句的字符串

string content //参数内容, 比如一个字段是格式复杂的文章, 有特殊符号, 可以通过这个方式添加

方法返回值: int //返回一个数值表示此 SqlCommand 命令执行后影响的行数

(3) 方法名称: GetSingle () 执行一条计算查询结果语句, 返回查询结果, 通过指定专用的连接数据库字符串 connectionString 连接数据库

方法参数: string SQLString //一个包含计算的查询语句

方法返回值: object //返回一个 object 类型的数据, 可以通过 Convert.To{Type} 方法转换类型

(4) 方法名称: ExecuteReader () 执行一条查询语句, 返回 SqlDataReader 类型的查询结果, 通过指定专用的连接数据库字符串 connectionString 连接数据库

方法参数: string strSQL //查询语句

方法返回值: SqlDataReader //返回一个 SqlDataReader 类型的查询结果

(5) 方法名称: ExecuteReader () 执行一条带参数的查询语句, 返回查询结果, 通过指定专用的连接数据库字符串 connectionString 连接数据库

方法参数: string strSQL //查询语句

SqlParameter[] commandParameters 以数组形式提供 SqlCommand 命令中用到的参数列表

方法返回值: SqlDataReader //返回一个 SqlDataReader 类型的查询结果。

(6) 方法名称: Query () 执行一条查询语句, 返回 DataSet 类型的查询结果, 通过指定专用的连接数据库字符串 connectionString 连接数据库

方法参数: string SQLString //查询语句



方法返回值: DataSet //返回一个 DataSet 类型的查询结果

(7) 方法名称: Query () 执行一条带参数的查询语句, 返回查询结果, 通过指定专用的连接数据库字符串 connectionString 连接数据库

方法参数: string SQLString //查询语句

SqlParameter[] commandParameters 以数组形式提供 SqlCommand 命令中用到的参数列表

方法返回值: DataSet //返回一个 DataSet 类型的查询结果

(8) 方法名称: ExecuteSqlTran () 执行多条 SQL 语句, 实现数据库事务, 通过指定专用的连接数据库字符串 connectionString 连接数据库

方法参数: SQLStringList // SQL 语句的哈希表, 其中 key 为 sql 语句, value 是该语句的 SqlParameter[]

方法返回值: void //返回值为空

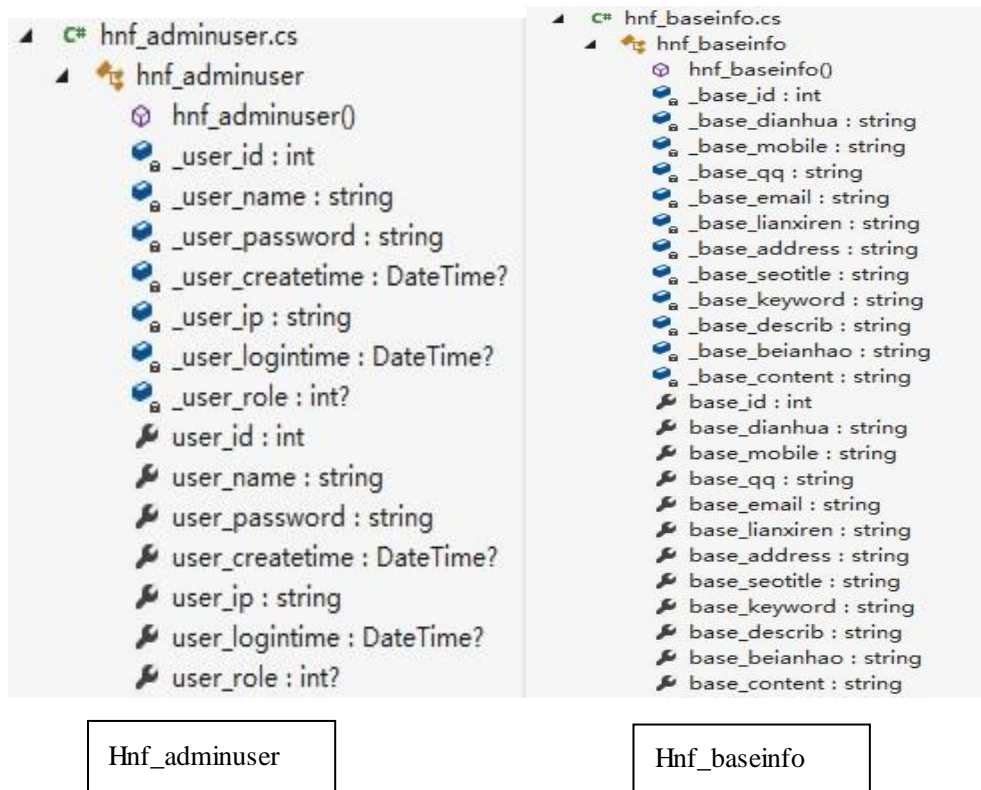
### 5.3.2 业务是实体类 (Model)

主要用于封装数据的, 就是把数据库中的每一个数据表定义一个类。每一个数据字段就是 model 中的每一个属性。每一个属性对应一个 get、set 方法。业务实体类中所有的类及对应的数据表如表 4 所示。

表 4 业务实体类列表

序号	实体类名称	对应的表名
1	实体类 hnf_adminuser	hnf_adminuser (管理员信息表)
2	实体类 hnf_baseinfo	hnf_baseinfo (基本信息表)
3	实体类 hnf_news	hnf_news (新闻表)
4	实体类 hnf_n_type	hnf_n_type (新闻类型表)
5	实体类 hnf_loginlog	hnf_loginlog (登录日志表)

以 hnf\_adminuser hnf\_baseinfo 两个表为例展示 model 中的业务实体类。



### 5.3.3 数据访问层（DAL）

主要是负责 MyDB 数据库的访问，该层所做事务直接操作 MyDB 数据库中的所有表，如针对数据表的增加、删除、修改、更新、查找等。数据访问类及类中方法的功能详细说明参看表 5。

表 5 数据访问类及类中方法的说明

序号	数据访问类	类中的方法名	功能
1	hnf_baseinfo	GetMaxId ( )	得到 hnf_baseinfo 最大 ID
		Exists(System. Int32)	hnf_baseinfo 是否存在该记录
		Add(Model. hnf_baseinfo)	增加 hnf_baseinfo 一条数据
		Update(Model. hnf_baseinfo)	更新 hnf_baseinfo 一条数据
		Delete(System. Int32)	删除 hnf_baseinfo 一条数据
		DeleteList(System. String)	删除 hnf_baseinfo 一条数据
		GetModel(System. Int32)	得到 hnf_baseinfo 一个对象实体
		GetModelByCache(System. Int32)	得到 hnf_baseinfo 一个对象实体，从缓存中
		GetList(System. String)	获得 hnf_baseinfo 数据列表
		GetList(System. Int32, System. String, System. String)	获得 hnf_baseinfo 前几行数据
		GetModellist(System. String)	获得 hnf_baseinfo 数据列表

序号	数据访问类	类中的方法名	功能
		DataTableToList(System.Data.DataTable)	获得 hnf_baseinfo 数据列表
		GetAllList()	获得 hnf_baseinfo 数据列表
2	hnf_adminuser	GetMaxId()	得到 hnf_adminuser 中记录的最大 ID
		Exists(System.Int32)	检测 hnf_adminuser 表中是否存在该记录
		Add(Model.hnf_adminuser)	向 hnf_adminuser 表 增加一条记录
		Update(Model.hnf_adminuser)	修改 hnf_adminuser 一条记录
		Delete(System.Int32)	删除 hnf_adminuser 一条记录
		DeleteList(System.String)	批量删除 hnf_adminuser 中的记录
		GetModel(System.Int32)	得到一个 hnf_adminuser 对象实体
		GetList(System.String)	获得 hnf_adminuser 数据列表
		GetList(System.Int32, System.Int32)	获得 hnf_adminuser 前几行数据
3	hnf_news	GetMaxId()	得到 hnf_news 最大 ID
		Exists(System.Int32)	hnf_news 中是否存在该记录
		Add(Model.hnf_news)	向 hnf_news 增加一条数据
		Update(Model.hnf_news)	在 hnf_news 更新一条数据
		Delete(System.Int32)	在 hnf_news 删除一条数据
		DeleteList(System.String)	在 hnf_news 删除一条数据
		GetModel(System.Int32)	向 hnf_news 得到一个对象实体
		GetModelByCache(System.Int32)	向 hnf_news 得到一个对象实体, 从缓存中
		GetList(System.String)	获得 hnf_news 数据列表
		GetList(System.Int32, System.String, System.String)	获得 hnf_news 前几行数据
		GetModelList(System.String)	获得 hnf_news 数据列表
		DataTableToList(System.Data.DataTable)	获得 hnf_news 数据列表
		GetAllList()	获得 hnf_news 数据列表
		getnum(System.Int32, System.Int32, System.Int32)	统计新闻发布
4	hnf_n_type	GetMaxId()	得到 hnf_n_type 最大 ID
		Exists(System.Int32)	hnf_n_type 是否存在该记录
		Add(Model.hnf_n_type)	向 hnf_n_type 增加一条数据
		Update(Model.hnf_n_type)	更新 hnf_n_type 一条数据
		Delete(System.Int32)	删除 hnf_n_type 一条数据
		DeleteList(System.String)	删除 hnf_n_type 一条数据
		GetModel(System.Int32)	得到 hnf_n_type 一个对象实体
		GetModelByCache(System.Int32)	得到 hnf_n_type 一个对象实体, 从缓存中
		GetList(System.String)	获得 hnf_n_type 数据列表
		GetList(System.Int32, System.String, System.String)	获得 hnf_n_type 前几行数据
		GetModelList(System.String)	获得 hnf_n_type 数据列表
		DataTableToList(System.Data.DataTable)	获得 hnf_n_type 数据列表
		GetAllList()	获得 hnf_n_type 数据列表
		UpdOrder(System.Int32, System.Int32)	hnf_n_type 根据 id 修改排序号
5	hnf_loginlog	GetMaxId()	得到 hnf_loginlog 最大 ID
		Exists(System.Int32)	hnf_loginlog 是否存在该记录
		Add(Model.hnf_loginlog)	向 hnf_loginlog 中增加一条数据
		Delete(System.Int32)	在 hnf_loginlog 中删除一条数据
		DeleteList(System.String)	在 hnf_loginlog 中删除一条数据
		Update(Model.hnf_loginlog)	在 hnf_loginlog 中更新一条数据
		GetModel(System.Int32)	在 hnf_loginlog 中得到一个对象实体
		GetModelByCache(System.Int32)	在 hnf_loginlog 中得到一个对象实体,

序号	数据访问类	类中的方法名	功能
			从缓存中
		GetList(System.String)	获得 hnf_loginlog 中数据列表
		GetList(System.Int32, System.String, System.String)	获得 hnf_loginlog 中前几行数据
		GetModelList(System.String)	获得 hnf_loginlog 中数据列表
		DataTableToList(System.Data.DataTable)	获得 hnf_loginlog 中数据列表

以下以 hnf\_adminuser 数据访问类为例展示在 DAL 中如何操作实体类编写操作数据库的增加、删除、修改、更新、查找等操作，返回值用于 BLL 调用。

```

/// <summary>
/// 向hnf_adminuser表中增加一条数据
/// </summary>
public int Add(Model.hnf_adminuser model)
{
    StringBuilder strSql=new StringBuilder();
    strSql.Append("insert into hnf_adminuser(");
    strSql.Append("user_name,user_password,user_createtime,user_ip,user_logintime,u");
    strSql.Append("ser_role)");
    strSql.Append(" values (");
    strSql.Append("@user_name,@user_password,@user_createtime,@user_ip,@user_logint");
    strSql.Append("ime,@user_role");
    strSql.Append(");select @@IDENTITY");
    SqlParameter[] parameters = {
        new SqlParameter("@user_name", SqlDbType.NVarChar, 50),
        new SqlParameter("@user_password", SqlDbType.NVarChar, 50),
        new SqlParameter("@user_createtime", SqlDbType.SmallDateTime),
        new SqlParameter("@user_ip", SqlDbType.NVarChar, 50),
        new SqlParameter("@user_logintime", SqlDbType.SmallDateTime),
        new SqlParameter("@user_role", SqlDbType.Int, 4) };
    parameters[0].Value = model.user_name;
    parameters[1].Value = model.user_password;
    parameters[2].Value = model.user_createtime;
    parameters[3].Value = model.user_ip;
    parameters[4].Value = model.user_logintime;
    parameters[5].Value = model.user_role;

    object obj = DbHelperSQL.GetSingle(strSql.ToString(), parameters);
    if (obj == null)
    {
        return 0;
    }
    else
    {
        return Convert.ToInt32(obj);
    }
}

```

```

/// <summary>

```

```

/// 更新hnf_adminuser表中一条数据
/// </summary>
public bool Update(Model.hnf_adminuser model)
{
    StringBuilder strSql=new StringBuilder();
    strSql.Append("update hnf_adminuser set ");
    strSql.Append("user_name=@user_name,");
    strSql.Append("user_password=@user_password,");
    strSql.Append("user_createtime=@user_createtime,");
    strSql.Append("user_ip=@user_ip,");
    strSql.Append("user_logintime=@user_logintime,");
    strSql.Append("user_role=@user_role");
    strSql.Append(" where user_id=@user_id");
    SqlParameter[] parameters = {
        new SqlParameter("@user_name", SqlDbType.NVarChar, 50),
        new SqlParameter("@user_password", SqlDbType.NVarChar, 50),
        new SqlParameter("@user_createtime", SqlDbType.SmallDateTime),
        new SqlParameter("@user_ip", SqlDbType.NVarChar, 50),
        new SqlParameter("@user_logintime", SqlDbType.SmallDateTime),
        new SqlParameter("@user_role", SqlDbType.Int, 4),
        new SqlParameter("@user_id", SqlDbType.Int, 4) };
    parameters[0].Value = model.user_name;
    parameters[1].Value = model.user_password;
    parameters[2].Value = model.user_createtime;
    parameters[3].Value = model.user_ip;
    parameters[4].Value = model.user_logintime;
    parameters[5].Value = model.user_role;
    parameters[6].Value =model.user_id;

    int rows=DbHelperSQL.ExecuteSql(strSql.ToString(), parameters);
    if (rows > 0)
    {
        return true;
    }
    else
    {
        return false;
    }
}

```

```

/// <summary>
/// 删除hnf_adminuser表中一条数据
/// </summary>
public bool Delete(int user_id)
{
    StringBuilder strSql=new StringBuilder();
    strSql.Append("delete from hnf_adminuser ");
    strSql.Append(" where user_id=@user_id");
    SqlParameter[] parameters = {
        new SqlParameter("@user_id", SqlDbType.Int, 4)};

```

```

        parameters[0].Value = user_id;

        int rows=DbHelperSQL.ExecuteSql(strSql.ToString(),parameters);
        if (rows > 0)
        {
            return true;
        }
        else
        {
            return false;
        }
    }
}

```

```

    /// <summary>
    /// 获得hnf_adminuser表中数据列表
    /// </summary>
    public DataSet GetList(string strWhere)
    {
        StringBuilder strSql=new StringBuilder();
        strSql.Append("select user_id,user_name,user_password,user_createtime,user_ip,
                        user_logintime,user_role ");
        strSql.Append(" FROM hnf_adminuser ");
        if(strWhere.Trim()!="")
        {
            strSql.Append(" where "+strWhere);
        }
        return DbHelperSQL.Query(strSql.ToString());
    }
}

```

```

    /// <summary>
    /// 获得hnf_adminuser表中前几行数据
    /// </summary>
    public DataSet GetList(int Top,string strWhere,string filedOrder)
    {
        StringBuilder strSql=new StringBuilder();
        strSql.Append("select ");
        if(Top>0)
        {
            strSql.Append(" top "+Top.ToString());
        }
        strSql.Append(" user_id,user_name,user_password,user_createtime,
                        user_ip,user_logintime,user_role ");
        strSql.Append(" FROM hnf_adminuser ");
        if(strWhere.Trim()!="")

```

```

{
    strSql.Append(" where "+strWhere);
}
strSql.Append(" order by " + filedOrder);
return DbHelperSQL.Query(strSql.ToString());
}

```

### 5.3.4 业务逻辑层（BLL）

业务逻辑层（BLL）在体系架构中的位置很关键，它处于数据访问层与表示层中间，起到了数据交换中承上启下的作用。本层主要是针对具体问题的操作，也可以说是对数据层的操作，对数据业务的逻辑处理等。在本网站中业务逻辑层（BLL）主要是用于接收表示层（web）传入的实参，根据用户的具体需求，调用数据访问层（DAL）中对应的具体方法，将返回的结果显示到表示层（web）中。业务逻辑层(BLL)中所有的类及功能如表 6 所示。

表 6 业务逻辑类及类中的方法说明

序号	业务逻辑类	类中的方法名	功能
1	hnf_baseinfo	GetMaxId ( )	得到 hnf_baseinfo 最大 ID
		Exists(System. Int32)	hnf_baseinfo 是否存在该记录
		Add(Model. hnf_baseinfo)	增加 hnf_baseinfo 一条数据
		Update(Model. hnf_baseinfo)	更新 hnf_baseinfo 一条数据
		Delete(System. Int32)	删除 hnf_baseinfo 一条数据
		DeleteList(System. String)	删除 hnf_baseinfo 一条数据
		GetModel(System. Int32)	得到 hnf_baseinfo 一个对象实体
		GetModelByCache(System. Int32)	得到 hnf_baseinfo 一个对象实体，从缓存中
		GetList(System. String)	获得 hnf_baseinfo 数据列表
		GetList(System. Int32, System. String, System. String)	获得 hnf_baseinfo 前几行数据
		GetModelList(System. String)	获得 hnf_baseinfo 数据列表
		DataTableToList(System. Data. DataTable)	获得 hnf_baseinfo 数据列表
		GetAllList ( )	获得 hnf_baseinfo 数据列表
2	hnf_adminuser	GetMaxId ( )	得到 hnf_adminuser 中记录的最大 ID
		Exists(System. Int32)	检测 hnf_adminuser 表中是否存在该记录
		Add(Model. hnf_adminuser)	向 hnf_adminuser 表 增加一条记录
		Update (Model. hnf_adminuser)	修改 hnf_adminuser 一条记录
		Delete(System. Int32)	删除 hnf_adminuser 一条记录
		DeleteList(System. String)	批量删除 hnf_adminuser 中的记录
		GetModel(System. Int32)	得到一个 hnf_adminuser 对象实体
		GetList(System. String)	获得 hnf_adminuser 数据列表

序号	业务逻辑类	类中的方法名	功能
		GetList(System. Int32, System. Int32)	获得 hnf_adminuser 前几行数据
3	hnf_news	GetMaxId ( )	得到 hnf_news 最大 ID
		Exists(System. Int32)	hnf_news 中是否存在该记录
		Add(Model. hnf_news)	向 hnf_news 增加一条数据
		Update(Model. hnf_news)	在 hnf_news 更新一条数据
		Delete(System. Int32)	在 hnf_news 删除一条数据
		DeleteList(System. String)	在 hnf_news 删除一条数据
		GetModel(System. Int32)	向 hnf_news 得到一个对象实体
		GetModelByCache(System. Int32)	向 hnf_news 得到一个对象实体, 从缓存中
		GetList(System. String)	获得 hnf_news 数据列表
		GetList(System. Int32, System. String, System. String)	获得 hnf_news 前几行数据
		GetModelList(System. String)	获得 hnf_news 数据列表
		DataTableToList(System. Data. DataTable)	获得 hnf_news 数据列表
		GetAllList ( )	获得 hnf_news 数据列表
		getnum(System. Int32, System. Int32, System. Int32)	统计新闻发布
4	hnf_n_type	GetMaxId ( )	得到 hnf_n_type 最大 ID
		Exists(System. Int32)	hnf_n_type 是否存在该记录
		Add(Model. hnf_n_type)	向 hnf_n_type 增加一条数据
		Update(Model. hnf_n_type)	更新 hnf_n_type 一条数据
		Delete(System. Int32)	删除 hnf_n_type 一条数据
		DeleteList(System. String)	删除 hnf_n_type 一条数据
		GetModel(System. Int32)	得到 hnf_n_type 一个对象实体
		GetModelByCache(System. Int32)	得到 hnf_n_type 一个对象实体, 从缓存中
		GetList(System. String)	获得 hnf_n_type 数据列表
		GetList(System. Int32, System. String, System. String)	获得 hnf_n_type 前几行数据
		GetModelList(System. String)	获得 hnf_n_type 数据列表
		DataTableToList(System. Data. DataTable)	获得 hnf_n_type 数据列表
		GetAllList ( )	获得 hnf_n_type 数据列表
		UpdOrder(System. Int32, System. Int32)	hnf_n_type 根据 id 修改排序号
5	hnf_loginlog	GetMaxId ( )	得到 hnf_loginlog 最大 ID
		Exists(System. Int32)	hnf_loginlog 是否存在该记录
		Add(Model. hnf_loginlog)	向 hnf_loginlog 中增加一条数据
		Delete(System. Int32)	在 hnf_loginlog 中删除一条数据
		DeleteList(System. String)	在 hnf_loginlog 中删除一条数据
		Update(Model. hnf_loginlog)	在 hnf_loginlog 中更新一条数据
		GetModel(System. Int32)	在 hnf_loginlog 中得到一个对象实体
		GetModelByCache(System. Int32)	在 hnf_loginlog 中得到一个对象实体, 从缓存中
		GetList(System. String)	获得 hnf_loginlog 中数据列表
		GetList(System. Int32, System. String, System. String)	获得 hnf_loginlog 中前几行数据
		GetModelList(System. String)	获得 hnf_loginlog 中数据列表
		DataTableToList(System. Data. DataTable)	获得 hnf_loginlog 中数据列表



以下以 hnf\_adminuser 业务逻辑类为例展示引用 DAL 和实体类编写操作数据库的增添、删除、修改、更新、查找等操作，返回值用于 WEB 层调用。

```
/// 增加hnf_adminuser表中一条数据
public int Add(NewsSolution.Model.hnf_adminuser model)
{
    return dal.Add(model);
}
```

```
/// 更新hnf_adminuser表中一条数据
/// </summary>
public bool Update(NewsSolution.Model.hnf_adminuser model)
{
    return dal.Update(model);
}
```

```
/// <summary>
/// 删除hnf_adminuser表中一条数据
/// </summary>
public bool Delete(int user_id)
{
    return dal.Delete(user_id);
}
```

```
/// 获得hnf_adminuser表中前几行数据
/// </summary>
public DataSet GetList(int Top, string strWhere, string filedOrder)
{
    return dal.GetList(Top, strWhere, filedOrder);
}
```

```
/// <summary>
/// 获得hnf_adminuser表中数据列表
/// </summary>
public List< NewsSolution.Model.hnf_adminuser> GetModelList(string strWhere)
{
    DataSet ds = dal.GetList(strWhere);
    return DataTableToList(ds.Tables[0]);
}
```

### 5.3.5 表示层（WEB）

WEB 是展现给用户的网页。以下以 hnf\_adminuser 表为例展示引用 BLL 和实体类编写操作数据库的增添、删除、修改、更新、查找等操作，返回值用于网页的显示及调用。

向 hnf\_adminuser 表中添加用户界面, 如图 9、图 10 所示。



图 9 添加用户界面

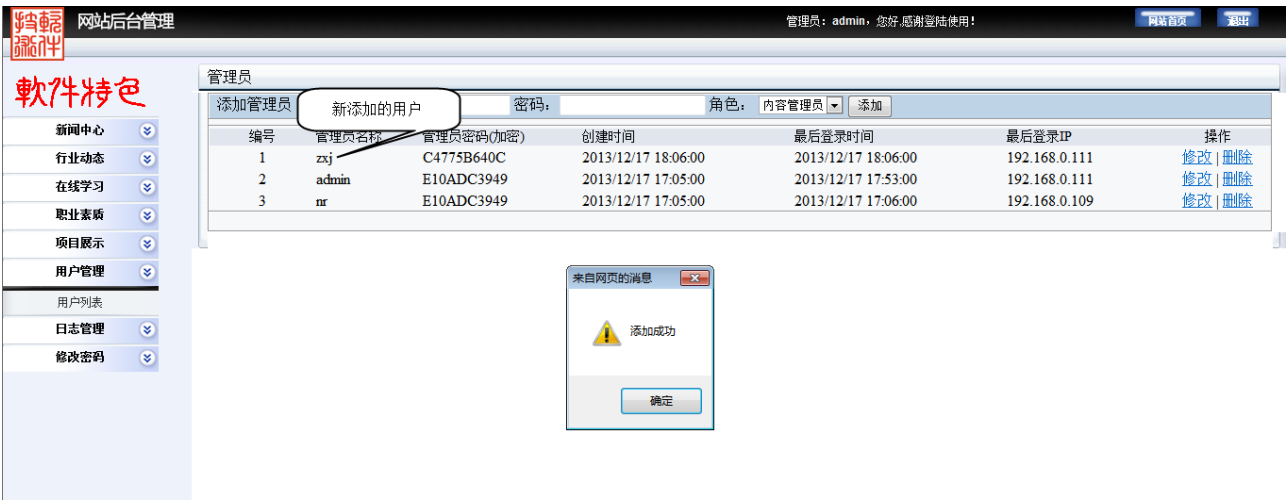


图 10 添加用户成功界面

```

///向 hnf_adminuser 表中添加用户的代码
protected void btnAdd_Click(object sender, EventArgs e)
{
    if (!string.IsNullOrEmpty(HiddenField1.Value) && HiddenField1.Value != "1")
    {
        Page.ClientScript.RegisterStartupScript(GetType(), "", "<script>alert(' 您没有此权限! ')</script>");
        return;
    }
    if (txtname.Value == "" || txtpwd.Value == "")
    {
        Page.ClientScript.RegisterStartupScript(GetType(), "", "<script>alert(' 用户名或密码不能为空 ')</script>");
    }
    else
    {
        madmin.user_name = txtname.Value.Trim();
        madmin.user_password =
            System.Web.Security.FormsAuthentication.HashPasswordForStoringInConfigFile(txtpwd.Value.Trim(), "md5");
        madmin.user_logintime = DateTime.Now;
        madmin.user_createtime = DateTime.Now;
        madmin.user_ip = Request.ServerVariables.Get("Remote_Addr").ToString();
        madmin.user_role=Convert.ToInt32(ddrole.SelectedValue.ToString());
        admin.Add(madmin);
        Page.ClientScript.RegisterStartupScript(GetType(), "", "<script>alert(' 添加成功 ')</script>");
        bindate();
        basepage bp = new basepage();
        string sql = "insert into hnf_loginlog (log_type,log_note,log_ip,log_time)
values(1,'【管理员】+Session["Mname"]+'添加用户成功! ',''+bp.GetIp()+'','"+
+ DateTime.Now + "')";
        Maticsoft.DBUtility.DbHelperSQL.ExecuteSql(sql);

    }
    txtname.Value = "";
    txtpwd.Value = "";
}

```

修改 hnf\_adminuser 表中一个用户信息的界面，如图 11、图 12 所示

图 11 修改用户姓名和密码界面

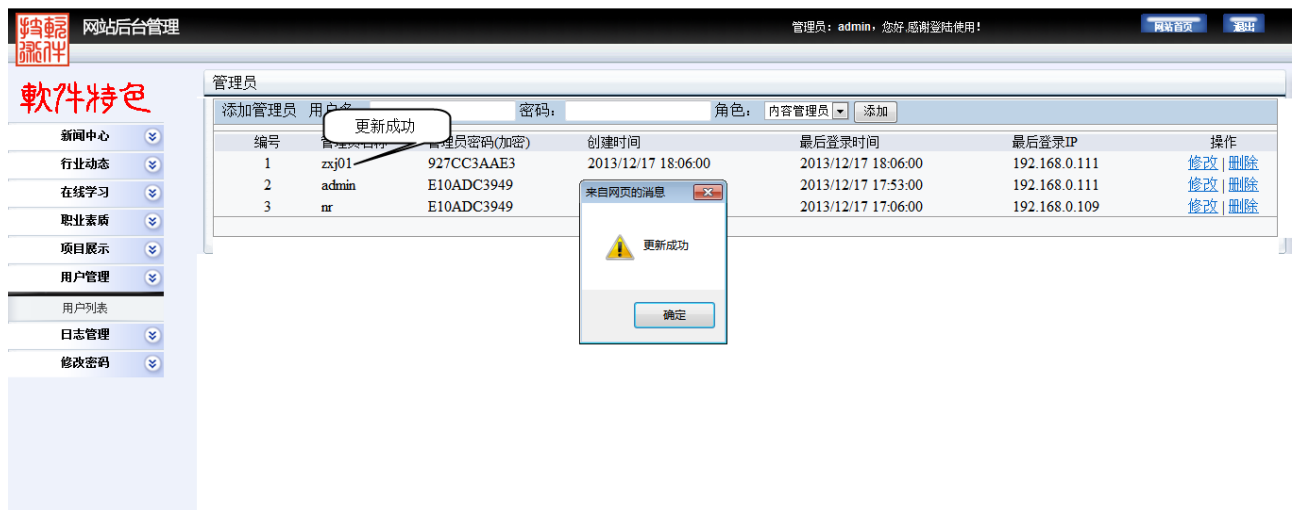
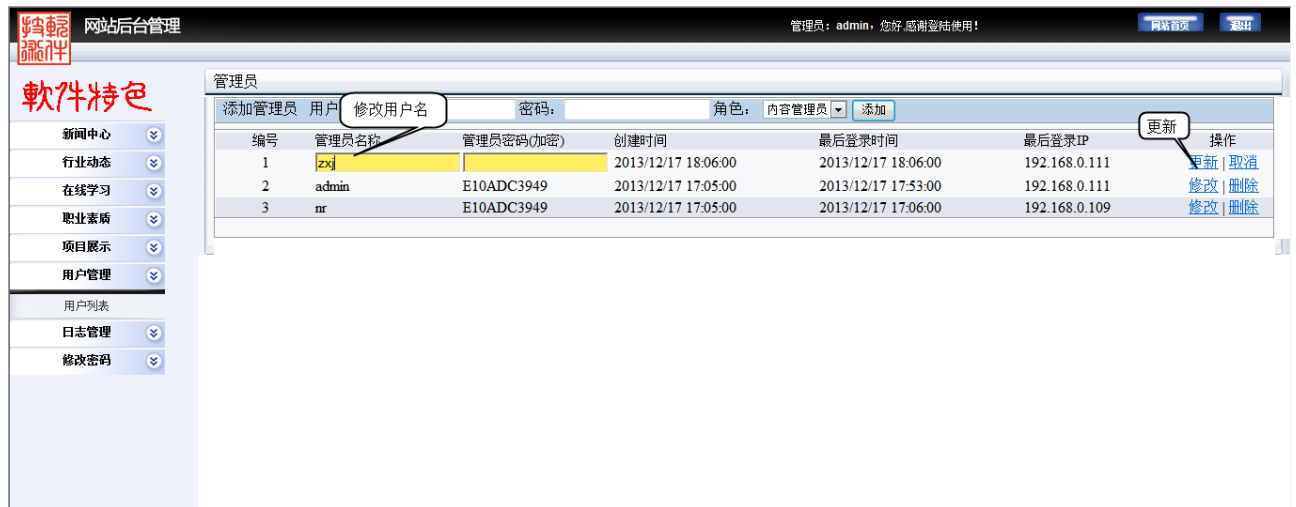


图 12 更新成功界面

```

///修改 hnf_adminuser 表中一个用户信息的代码
protected void dldtype_UpdateCommand(object source, DataListCommandEventArgs e)
{
    int id = Convert.ToInt32(dldtype.DataKeys[e.Item.ItemIndex]);
    string u = ((TextBox)e.Item.FindControl("txtuser")).Text.Trim();
    string p = ((TextBox)e.Item.FindControl("txtpass")).Text.Trim();
    if (p == "")
    {
        Page.ClientScript.RegisterStartupScript(GetType(), "", "<script>alert('密码不能为空')</script>");
        return;
    }
    madmin = admin.GetModel(id);
    madmin.user_name = u;
    madmin.user_password =

```

```

System.Web.Security.FormsAuthentication.HashPasswordForStoringInConfigFile(p,
"md5");
admin.Update(madmin);
Page.ClientScript.RegisterStartupScript(GetType(), "", "<script>alert('更新成功
')</script>");
dldtype.EditItemIndex = -1;
bindate();
basepage bp = new basepage();
string sql = "insert into hnf_loginlog (log_type,log_note,log_ip,log_time) values(1,'
【管理员】+ Session["Mname"] + "修改用户成功! "','"+ bp.GetIp() + "','"+ DateTime.Now
+ "')";
Maticsoft.DBUtility.DbHelperSQL.ExecuteSql(sql);
}

```

在 hnf\_adminuser 表删除一个用户的界面如图 13、图 14 所示。

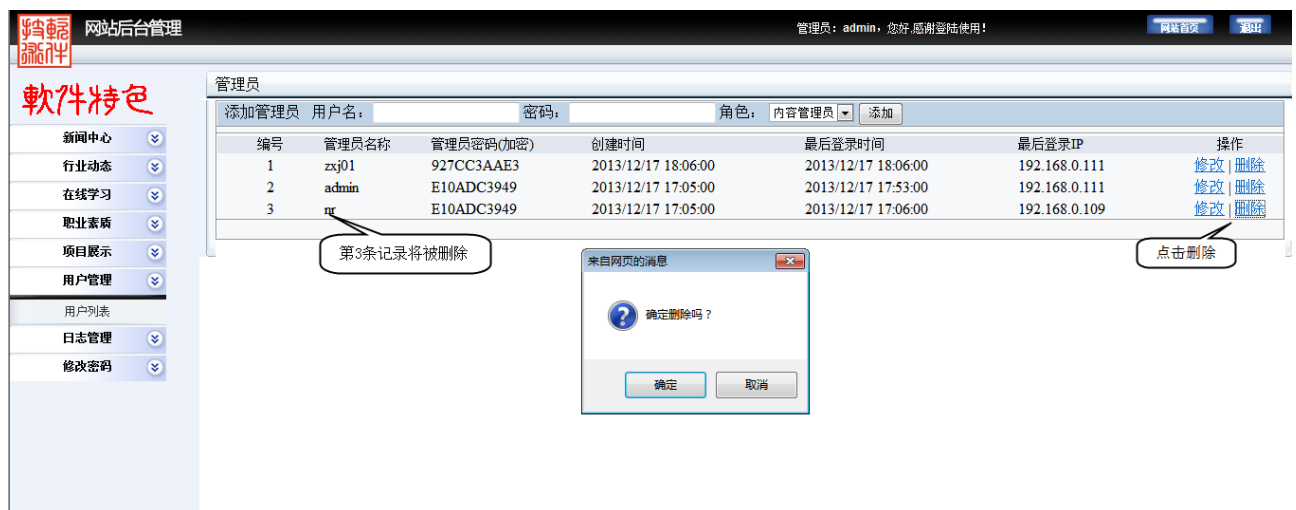


图 13 删除用户界面

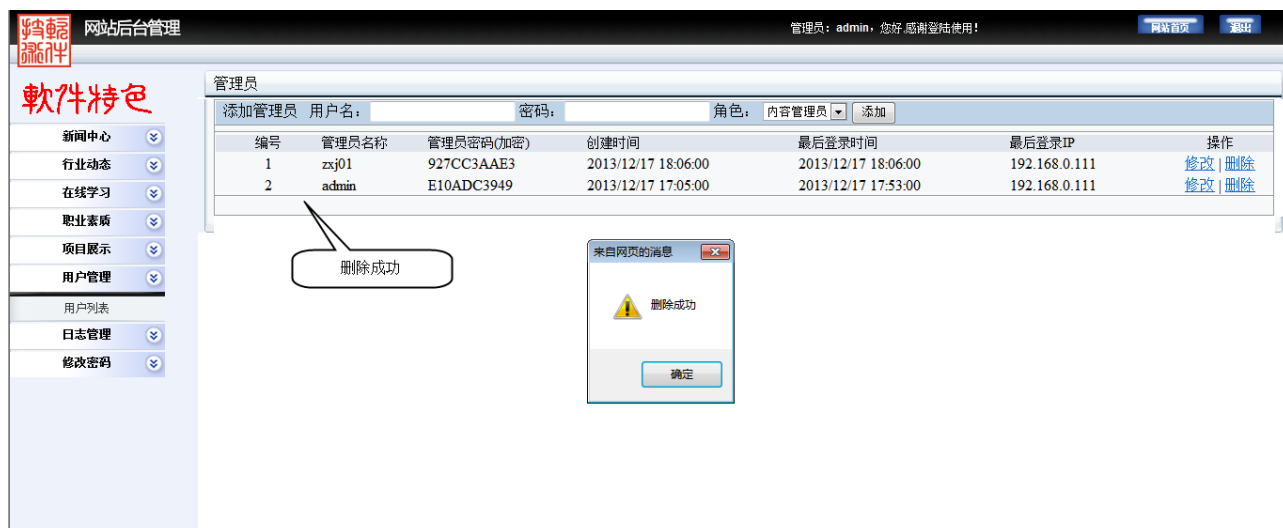


图 14 删除成功界面

```

///在 hnf_adminuser 表中删除用户的代码
protected void dldtype_DeleteCommand(object source, DataListCommandEventArgs e)
{
    int id = Convert.ToInt32(dldtype.DataKeys[e.Item.ItemIndex]);
    if (HiddenField1.Value == "1")
    {
        if (id == 1)
        {
            Page.ClientScript.RegisterStartupScript(GetType(), "", "<script>alert(' 管理员admin不能删除! ')</script>");
            return;
        }
        admin.Delete(id);
    }
    else if (HiddenField1.Value != "")
    {
        ds = admin.GetList("user_id=" + HiddenField1.Value + " order by user_logintime desc");
        if (id.ToString() == ds.Tables[0].Rows[0][0].ToString())
        {
            Page.ClientScript.RegisterStartupScript(GetType(), "", "<script>alert(' 管理员自己不能删除自己! ')</script>");
            return;
        }
    }
    else
    {
        Page.ClientScript.RegisterStartupScript(GetType(), "", "<script>alert(' 管理员admin不能删除! ')</script>");
    }
}

```

```

        return;
    }
    Page.ClientScript.RegisterStartupScript(GetType(), "", "<script>alert('删除成功')</script>");
    bindate();
}

```

显示 hnf\_adminuser 表中所有用户的界面，如图 15 所示。

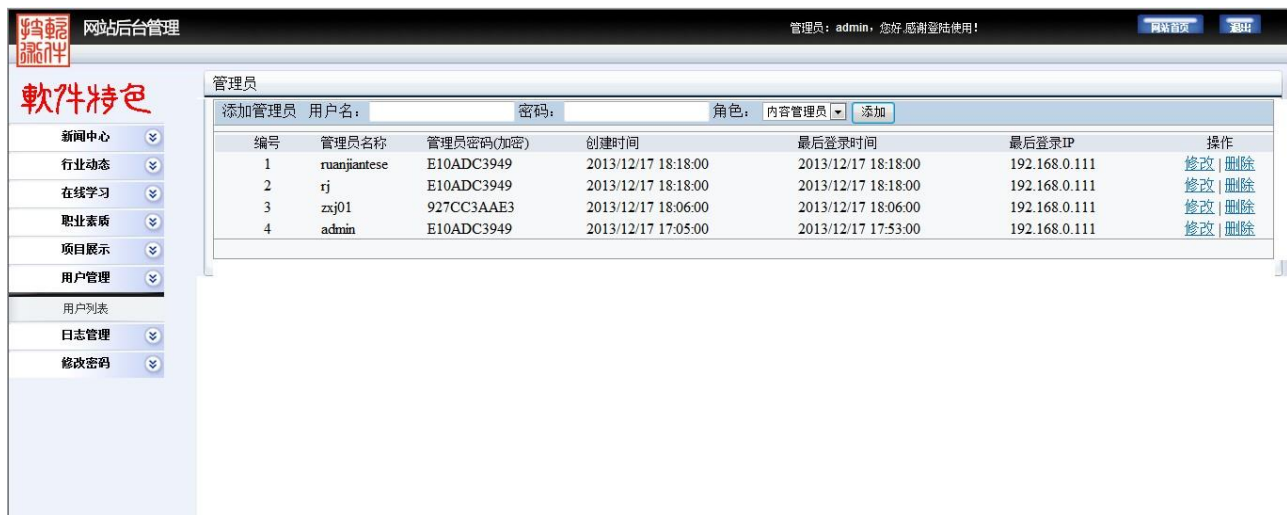


图 15 显示所用用户的界面

///查询 hnf\_adminuser 表中所有用户的代码

```

protected void bindate()
{
    if (!string.IsNullOrEmpty(HiddenField1.Value) && HiddenField1.Value != "1")
    {
        dldtype.DataSource = admin.GetList("user_id=" + HiddenField1.Value + " order by user_logintime desc");
        dldtype.DataBind();
    }
    else
    {
        dldtype.DataSource = admin.GetList("1=1 order by user_logintime desc");
        dldtype.DataBind();
    }
}

```

## 六、运行环境

### 6.1 硬件平台

最低要求：1.6 GHz CPU、384 MB RAM、1024x768 显示器、5400 RPM 硬盘

建议配置：2.2 GHz 或速度更快的 CPU、1024 MB 或更大容量的 RAM、1280x1024 显示器、7200 RPM 或更高转速的硬盘

### 6.2 网络平台

Internet 环境中采用独立的服务器放在学院网络中心机房，采用独立域名 rjts.hbcit.edu.cn 进行访问。

### 6.3 软件平台

1. 服务器环境：操作系统 windows2003 server
2. 客户端：ie6、ie7、ie8、ie9、火狐、谷歌等主流浏览器。
3. 语言支持：Web 发布 IIS，数据库 SQL Server2012

## 七、工作任务及进度计划

### 7.1 项目组成员

根据本项目的工作内容和范围，我们将成立一个 3 个人左右的项目工作组来负责本项目的开发。具体职责如下：

1. 项目负责人 段志磊

项目负责人负责项目管理、组织、协调，对项目资源进行控制，是项目能够按照计划实施，满足项目规定的业务需求。项目负责人对项目的质量、进度和成本负责。项目负责人负责和客户沟通。并负责对整个项目中的数据库结构及功能程序的设计。

2. 程序员（3 人）段志磊、张晓静、韩鑫鹏




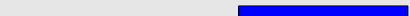

负责外部网站和内部服务系统的程序及多媒体的开发。



3. 前台制作（3 人）段志磊、张晓静、韩鑫鹏

负责网页的模板制作及 Html 搭建。

## 7.2 项目进度计划

ID	任务名称	开始时间	完成	持续时间	2013年 11月			2013年 12月			完成人
					11/10	11/17	11/24	12/1	12/8		
1	调研及分析	2013/11/11	2013/11/12	2d							段志磊、张晓静、韩鑫鹏
2	前台美工	2013/11/13	2013/11/18	4d							段志磊、张晓静、韩鑫鹏
3	各栏目前台及后台代码实现	2013/11/19	2013/12/16	20d							段志磊、张晓静、韩鑫鹏
4	代码整合验收测试	2013/12/3	2013/12/18	12d							段志磊、张晓静、韩鑫鹏
5	调试运行	2013/12/18	2013/12/19	2d							段志磊、张晓静、韩鑫鹏

## 八、网站推广和维护

网站开发过后的维护和资料的上传交给本专业的在校生来维护（2 名学生班长和学委）。这两名学生负责网站各个部分内容收集和撰写。

1. 通过搜索引擎推广网站：网站做好后，可以到百度、谷歌、必应等网站上进行搜索引擎登记，让更多检索、查找同行业的人查找到我们的网站。

2. 利用自己的网站推广自己的网站：网站建好后，不断更新自己的网站内容，这样会给访问者留下好的印象，增加回头率，让浏览者产生访问兴趣。

3. 利用<meta>标签，斟酌设置网站的描述和关键字，也可以提高搜索引擎自动搜索到的几率。