基于webservice的BBS系统

1. 需求分析

BBs论坛允许来自不同地区的用户在这里交流信息的平台。

1.1 业务需求

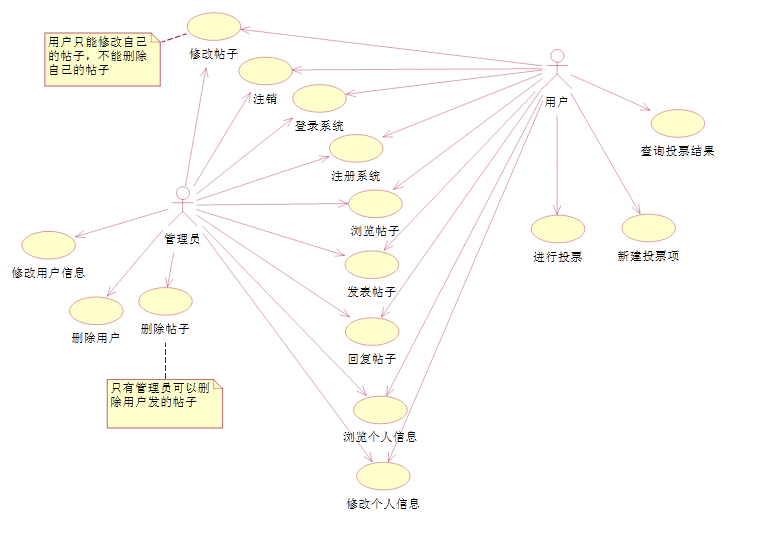
作为一个BBS论坛，其中的角色分为管理员与用户两类，用户用来进行论坛帖子的浏览、发布，修改本人的信息，投票的发布以及回复其他用户的帖子。管理员除了拥有用户的所有权限之外，还拥有可以删除帖子与投票的权力。

1.2 功能需求

从功能上来看，该BBS论坛有发布帖子与发布投票的功能。通过上述分析，在本例中把BBS系统分为以下几个模块：

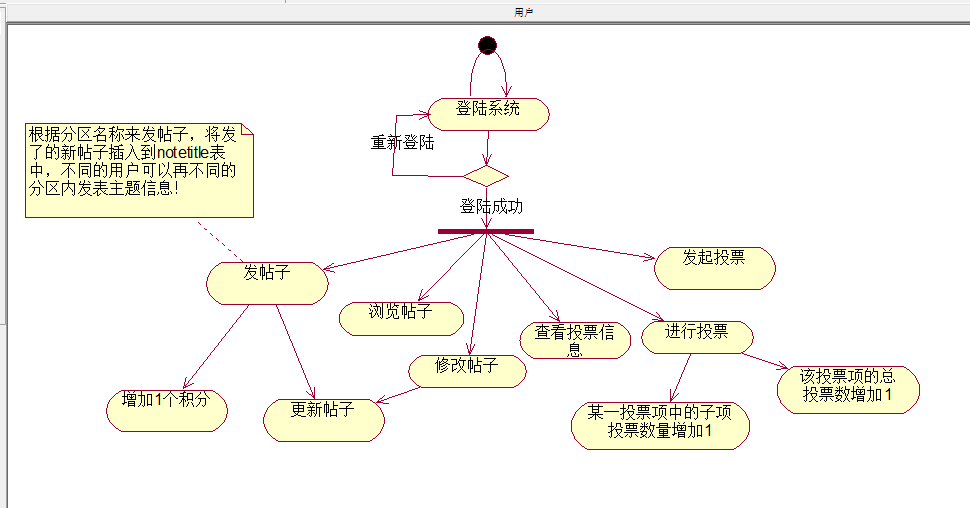
1. 主页面与登录模块。该模块主要用来进行用户的注册于登录操作，把用户注册的信息存入数据库中，在用户登录时从数据库中取出数据。
2. 发表主题模块。该模块主要用来进行用户对帖子的发布、修改、浏览操作。把帖子与其他用户的回复信息存入数据库中。
3. 投票模块。该模块主要用来进行用户对投票的发布、浏览操作，并把投票的票数信息存入数据库。
4. 管理模块。该模块主要用来进行数据的整理，其中管理员可以对帖子与投票进行删除操作，用户可以修改个人信息。

1.2．1 业务实体描述



注意：只有管理员才有权限来删除帖子，每个人的帖子只能由自己来修改，其他用户不能修改自己的帖子。

1.2．2 业务场景描述



1. 系统设计（详细设计）

2.1 文件功能设计

通过上述的模块分析，可以设计出网站的大致结构，每个模块由若干网页构成，具体设计如下表，这就是与BBS论坛的模块文件相关

|  |  |  |
| --- | --- | --- |
| 模块名 | 文件名 | 说明 |
| 注册与登录模块 | Welcome.aspx | 欢迎页面 |
| Regedit.aspx | 注册页面 |
| ErrMsg.aspx | 出错信息页面 |
| 发表主题模块 | Bbs.aspx | BBS主页面 |
| Content.aspx | 浏览主题页面 |
| EditContent.aspx | 编辑主题页面 |
| 投票模块 | Ballot.aspx | 投票页面 |
| NewSubject.aspx | 发表新投票页面 |
| 管理员模块 | Modify.aspx | 修改个人信息页面 |
| Admin.aspx | 管理员页面 |
| Passport.aspx | 个人中心页面 |
| Webservice模块 | Webservice.asmx | Webservice页面 |
| Webservice.cs | Webservice后台代码文件 |

2.2 数据库设计

本模块使用SQL Server2005作为后台数据库。设计数据库的一般步骤为：

1. SQL Server2005中新建一个名为BBS\_DB的数据库
2. 在BBS\_DB数据库中新建一张名为UserInfo的表，用来记录有关用户的详细信息。

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段 | 中文描述 | 数据类型 | 是否为空 | 备注 |
| UserID | 用户ID | Int | 否 | 主键，自增长 |
| userName | 用户名 | Char(10) | 否 |  |
| userNick | 用户昵称 | Char(10) | 否 |  |
| userPass | 用户密码 | Char(10) | 否 |  |
| userEmail | 用户电子邮件 | Varchar(50) | 否 |  |
| userMark | 用户备注 | Varchar(50) | 否 |  |
| userGroup | 用户组 | Int | 否 |  |
| userLever | 用户级别 | int | 否 |  |

（分析：用户id为表的主键用来标识这张表，用户级别用来记录该用户是普通用户还是管理员，最后用户级别是通过用户发表的论坛内容来确定的，发表的越多积分也就越高，相应的级别也就越高）

创建该表的SQL语句如下：

create table UserInfo

(

userID int identity(1,1) primary key not null,---用户ID，主键，identity(1,1)意思就是该列自动增长，由开始每次增加是，

userName char(10) not null,---------------------用户名称

userNick char(10) not null,---------------------用户昵称

userPass char(10) not null,---------------------用户密码

userEmail varchar(50) not null,------------------用户电子邮件

userMark varchar(500) not null,-----------------用户备注

userGroup int , -------------------------用户组

userLevel int -------------------------用户等级

);

1. 在BBS\_DB数据库中新建一张名为TitleList的表。在这个项目当中，通过这张表获得论坛主题的标题列表。该表的字段结构如下：

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 字段 | 中文描述 | 数据类型 | | 是否为空 | 备注 |
| NoteId | 主题ID | int | 否 | | 主键，自增长 |
| userNick | 发帖用户昵称 | Char(10) | 否 | |  |
| noteTitle | 帖子标题 | Text | 否 | |  |
| contTime | 发帖时间 | Datetime | 否 | |  |
| noteGroup | 帖子所属分区 | Char(10) | 否 | |  |
| noteContent | 帖子正文 | Text | 否 | |  |
| username | 发帖用户名称 | Char(10) | 否 | |  |

（分析：此表记录bbs论坛中用户主动发表内容，包括发表文章的标题、内容、发帖的时间、帖子所属的分类、帖子的发起者是谁等）

创建该表的SQL语句：

---------------------------------------------------------------------

--建论坛主题的标题列表

---------------------------------------------------------------------

create table TitleList

(

noteId int identity(1,1) primary key not null ,----论坛主题的标题id

userNick char(10) not null,-----------------------发帖用户昵称

noteTitle text not null,---------------------------帖子标题

contTime datetime not null,------------------------发帖时间

noteGroup char(10) not null,-----------------------帖子所属的分区

noteContent text not null,-------------------------帖子的正文内容

username char(10) not null,-----------------------发帖用户名称

)

1. 在BBS\_DB数据库中创建一张名为NoteContent的表，用来记录帖子的内容与回复信息。该表的字段如下：

|  |  |  |  |
| --- | --- | --- | --- |
| 字段 | 中文描述 | 数据类型 | 是否为空 |
| returnId | 回复列表ID | Int | 主键，自增长 |
| NoteId | 主题ID | Int | 否，外键【TitleList. NoteId】 |
| noteContent | 回复内容 | Varchar（500） | 否 |
| userNick | 回复用户昵称 | Char(10) | 否 |
| noteTime | 回复时间 | Datetime | 否 |
| userName | 回复用户名 | Char(10) | 否 |

（分析：该表用来记录帖子的内容与回复信息，与主题内容不同的是，这张表主要是记录用户对某主题的回复信息，包括回复的内容、回复的时间、回复者的名称等）

创建该表的SQL语句为：

注意：如果noteTime改为其他格式（char）可能会乱码，不会出现应有的效果

\*我们这里的外键是NoteId，我们通过主题的编号来进行回复，并将回复的信息保存在回复的表中

go

--------------------------------------------------------------------------

--建论坛帖子的内容与回复信息列表

---------------------------------------------------------------------------

create table NoteContent

(

noteId int primary key not null,----------------------主题ID

noteContent varchar(500) not null,----------------回复内容

userNick char(10) not null,----------------------回复用户昵称

noteTime datetime(10) not null,----------------------回复时间

userName char(10) not null,----------------------回复用户名

)

1. 在BBS\_DB数据库中建一张名为Subject的表，用来记录投票主题的信息。该表字段结构如下图：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段 | 中文描述 | 数据类型 | 是否为空 | 备注 |
| SubjectId | 投票主题ID | int | 否 | 主键，自增长 |
| SubjectName | 发起投票用户昵称 | Varchar(50) | 否 |  |
| SumNumber | 帖子标题 | int | 是 |  |

创建该表的SQL语句为：

---------------------------------------------------------------------

--建论坛帖子的内容与回复信息列表

---------------------------------------------------------------------

create table Subject

(

SubjectId int primary key identity(1,1) not null,--投票主题ID

SubjectName varchar(50) not null,------------------发起投票用户昵称

SumNumber int ,-----------------------------------帖子标题

)

1. 在BBS\_DB数据库中新建一张名为Ballot的表，用来记录投票项目的信息。该表为：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段 | 中文描述 | 数据类型 | 是否为空 | 备注 |
| OptionID | 主题ID | Int | 否 | 主键，自增长 |
| SubjectId | 投票项目id | Int | 否 | 外键，【Subject.subjectid】 |
| OptionName | 投票项目名 | Varchar（50） | 否 |  |
| CountNumber | 投票数 | Int | 是 |  |

create table Ballot

(

OptionID int primary key not null ,----------------主题ID

SubjectId int not null,----------------------------投票项目id

OptionName varchar(50) not null, ------------------投票项目名称

CountNumer int ,-----------------------------------投票数

)

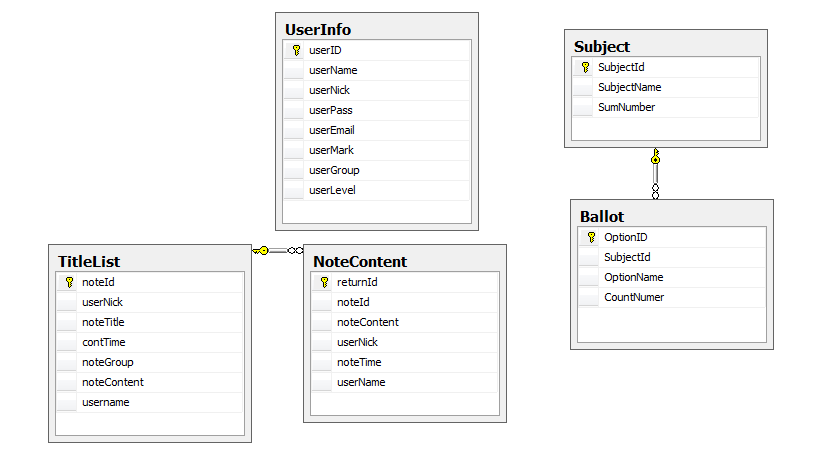
表建完之后，可以需要手动通过SQL的查询分析器来添加一个管理员，在用户列表中添加数据

insert into UserInfo values('Admin','管理员','root','user@sina.com','This is Admin',0,0)

注意：因为设置了用户编号为自增长类型，因此在初始化时不需要对其赋值。空值也不需要

注意：

1. 在SQL中，使用getdate（）方法可以获取当前的时间
2. 如果字段没有设置为“自增长”，那么该字段即使为null，也得在插入语句中写入null。
3. 以下是数据库的结构关系



注意：可能自己的电脑有问题，因此，我们在建立的时候一定要建立网站，不然不会出现App-code这个文件夹，不能建项目。但是还有一点，我们建的网站或者项目仍然不能用，此时应该选择开发工具为c#，可选项有几个，因此一定要好好选一选，这是由于安装vs的时候没搞好的原因。

<%@ WebService Language="C#" CodeBehind="~/App\_Code/Service.cs" Class="WebService.BBS\_DataBase" %>

其中，Language指的是使用的语言，CodeBehind指的是后台代码存在的位置，Class说明该服务的类名。

1. 编写WebService服务系统

3.1 主页面与登录相关的Web Service数据访问模块

本节开始进行模块编写。首先编写的是主页面与登录模块的代码，主页面与登录模块的作用为编写用户登录与注册过程。这里使用Web Service 来构建服务端对用户登录与注册的实现。首先进行该模块数据访问层的的设计。

由于时间原因这里没有将所有利用到面向对象的设计思想，利用“三层架构”的理念来将数据数据库连接、查询的操作与数据访问模块相分离（即利用DBHelper与DAL来实现数据的封装）

【第一部分】登录和注册模块后台模块设计

1. 用户登录模块（Login（）方法）

该方法写在webservice.Cs文件中

[WebMethod(Description = "该方法用于实现用户登录")]

public DataSet Login(string name, string pass)

{

//获得数据库连接

SqlConnection conn = new SqlConnection(DBconn);

//创建数据库操作命令

SqlCommand SqlCmd = conn.CreateCommand();

//使用pd\_Login存储过程来验证是否合法用户

SqlCmd.CommandType = CommandType.StoredProcedure;

SqlCmd.CommandText = "pd\_Login";

//传入用户名与密码参数

//参数1，用户名

SqlParameter parInput1 = SqlCmd.Parameters.Add("@Name",SqlDbType.Char);

parInput1.Direction = ParameterDirection.Input;

parInput1.Value = name.Trim();

//参数2，密码

SqlParameter parInput2 = SqlCmd.Parameters.Add("@Pass",SqlDbType.Char);

parInput2.Direction = ParameterDirection.Input;

parInput2.Value = pass.Trim();

//连接数据库

if (conn.State == ConnectionState.Closed)

{

conn.Open();

}

//执行SQL命令

SqlDataAdapter myDataAdapter = new SqlDataAdapter(SqlCmd);

DataSet myDataSet = new DataSet();

//填充数据集，并为数据集取别名

myDataAdapter.Fill(myDataSet,"UserInfor");

//获得用户等级与昵称信息

if (conn.State == ConnectionState.Open)

{

conn.Close();

}

//返回数据集

return myDataSet;

}

其中对应的存储过程为：

USE [MYBBS\_DB]

GO

/\*\*\*\*\*\* 对象: StoredProcedure [dbo].[pd\_Login] 脚本日期: 05/29/2013 15:53:28 \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

----------------------------------------------------创建登录存储过程-------------------------

CREATE PROCEDURE [dbo].[pd\_Login]

(

@Name char(10),

@Pass char(10)

)

AS

SELECT

userNick,userLevel

FROM

UserInfo

WHERE

(userPass=@Pass)and(userName=@Name)

1. CheckUser（）方法

该方法用来在注册的过程中检查用户名是否存在，用户向Web Service传递用户名，服务器返回用户是否存在的信息。以下为该方法的代码清单：

/// <summary>

///

/// </summary>

/// <param name="name">客户端输入的用户名</param>

/// <returns>返回为1，则用户在数据库中，返回为0，则用户不在数据库中</returns>

[WebMethod(Description = "检测用户是否存在")]

public int CheckUser(string name)

{

//获得数据库连接

SqlConnection conn = new SqlConnection(DBconn);

//创建数据库操作命令

SqlCommand SqlCmd = conn.CreateCommand();

//使用pd\_checkUser存储过程来验证是否合法用户

SqlCmd.CommandType = CommandType.StoredProcedure;

SqlCmd.CommandText = "pd\_checkUser";

//传入用户名参数

SqlParameter parInput1 = SqlCmd.Parameters.Add("@Name", SqlDbType.Char);

parInput1.Direction = ParameterDirection.Input;

parInput1.Value = name.Trim();

//获得返回值

SqlParameter parReturn1 = SqlCmd.Parameters.Add("returnvalue", SqlDbType.Int);

parReturn1.Direction = ParameterDirection.ReturnValue;

//打开数据库

if (conn.State == ConnectionState.Closed)

{

conn.Open();

}

int state;

//执行Sql命令

try

{

//获得命令的返回值

SqlCmd.ExecuteNonQuery();

//输出参数结果为0表明没有受影响，该用户不在数据库中，该参数为1表明受到影响，该用户在数据库中

state = Convert.ToInt32(SqlCmd.Parameters["returnvalue"].Value);

}

catch

{

//若查找失败 置state变量为0

state = 0;

}

if (conn.State == ConnectionState.Open)

conn.Close();

return state;

}

其中对应的存储过程为：

USE [MYBBS\_DB]

GO

/\*\*\*\*\*\* 对象: StoredProcedure [dbo].[pd\_ChceckUser] 脚本日期: 05/29/2013 15:23:11 \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE PROCEDURE [dbo].[pd\_ChceckUser]

(

@Name char(10)

)

as

begin

select userName=@Name from UserInfo where userName=@Name

if @@Rowcount<1

return 0

else

return 1

end

1. Register()方法

该方法用来在注册模块中把用户的信息加入到数据库中。以下为该方法的代码清单：

/// <summary>

/// 返回的是插入数据的结果，如果返回的是1，则插入成功；返回的是0，则插入失败

/// </summary>

/// <param name="regedit">用字符串数组来记录注册用户的信息</param>

/// <returns>返回的是一个输出参数</returns>

[WebMethod(Description = "注册")]

public int Register(string[] regedit)

{

//连接数据库

SqlConnection coon = new SqlConnection(DBconn);

SqlCommand SqlCmd = conn.CreateCommand();

//使用pd\_regedit存储过程

SqlCmd.CommandType = CommandType.StoredProcedure;

SqlCmd.CommandText = "pd\_regedit";

//设置注册的用户默认的组为1 为用户

//数据库中有默认的用户为0组 为管理员

//将用户信息作为参数传入

SqlParameter parInput1 = SqlCmd.Parameters.Add("@name",SqlDbType.Char);

parInput1.Direction = ParameterDirection.Input;

parInput1.Value = regedit[0].Trim();

SqlParameter parInput2 = SqlCmd.Parameters.Add("@nick", SqlDbType.Char);

parInput2.Direction = ParameterDirection.Input;

parInput2.Value = regedit[1].Trim();

SqlParameter parInput3 = SqlCmd.Parameters.Add("@pass", SqlDbType.Char);

parInput3.Direction = ParameterDirection.Input;

parInput3.Value = regedit[2].Trim();

SqlParameter parInput4 = SqlCmd.Parameters.Add("@email", SqlDbType.VarChar);

parInput4.Direction = ParameterDirection.Input;

parInput4.Value = regedit[3].Trim();

SqlParameter parInput5 = SqlCmd.Parameters.Add("@mark", SqlDbType.VarChar);

parInput5.Direction = ParameterDirection.Input;

parInput5.Value = regedit[4].Trim();

//获得存储过程的返回值

SqlParameter parReturn1 = SqlCmd.Parameters.Add("returnvalue", SqlDbType.Int);

parReturn1.Direction = ParameterDirection.ReturnValue;

//打开数据库连接

if (coon.State == ConnectionState.Closed)

{

coon.Open();

}

int state;

try

{

//执行插入用户信息命令，并获得返回值

SqlCmd.EndExecuteNonQuery();

//获取输出参数的值

state = Convert.ToInt32(SqlCmd.Parameters["returnvalue"].Value);

}

catch (Exception)

{

state = 0;

}

finally

{

//关闭数据库

if (coon.State == ConnectionState.Open)

{

coon.Close();

}

}

//返回插入数据是否成功的状态值

return state;

}

}

这段代码的作用是把用户在注册页面中所输入的信息打包成一个字符串数组传入webservice的注册函数中，接着从该数组中依次取出用户的各项信息，把该信息数据作为参数分别传入pd\_regedit的存储过程中，然后进行数据插入命令。在插入完成之后，如果返回值正确无误，则服务器端告知用户的注册过程顺利完成。

其中涉及到的存储过程为：

CREATE PROCEDURE pd\_regedit

(

@Name char(10),

@nick char(10),

@Pass char(10),

@email char(50),

@mark varchar(500)

)

AS

begin

insert into UserInfo values(@Name,@nick,@Pass,@email,@mark,1,0)

if @@Rowcount<1

return 0

else

return 1

end

GO

注意：这里默认的注册的是用户，而不是管理员（设置管理员的时候有特定的权限管理模块，只有超级用户才能够创建或者删除数据库管理员）

【第二部分】发表主题相关的Web Service访问模块

该模块用来在BBS系统中发布新的主题帖，并对已发布的主题进行浏览、修改、回复，在这Web Service完成对数据库中主题进行查看、修改和回复作用。

1. 前台业务逻辑设计

4.1 主页面与登录模块

主页面与登录模块通过webservice的相应方法实现了用户对BBS系统的登录与注册过程，同时包含了整个BBS系统的页面模块的编写。下面开始进行页面设计。

4.1.1 界面设计

①功能界面设计

该页面主要用于用户的登录和注册操作。

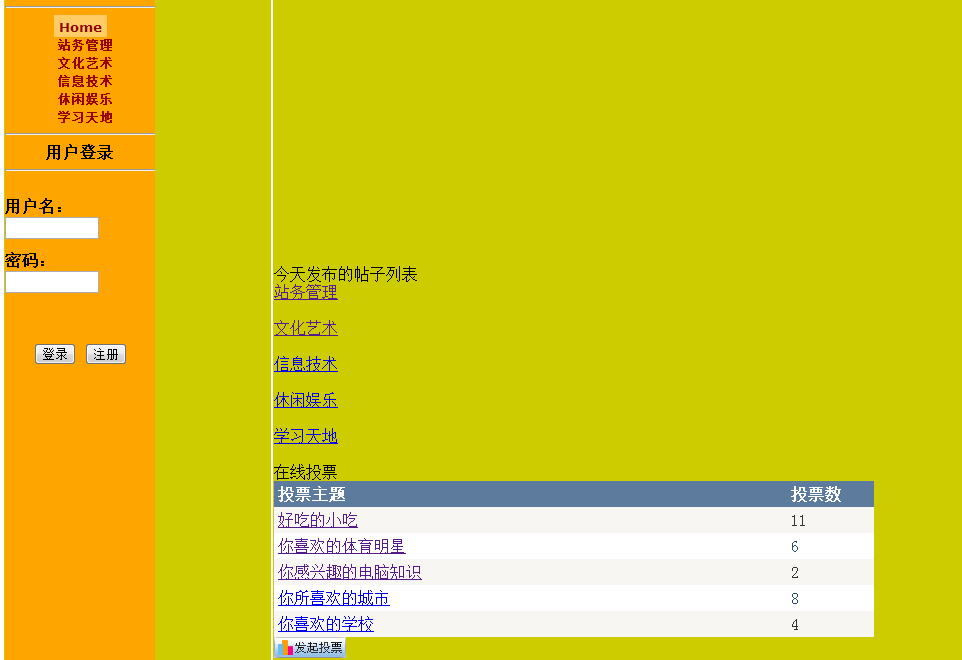
②功能文件关系

主页面与登录模块文件关系

|  |  |
| --- | --- |
| 文件名 | 作用 |
| BbsHeader.ascx | 页面导航条 |
| BbsSide.ascx | 页边导航与登录控件 |
| BbsFooter.ascx | 页尾导航条 |
| StyleSheet.css | 页面样式文件 |
| Site.master | 母版页 |
| Web.sitemap | 网站地图 |
| ErrMsg.aspx | 出错页面 |
| BBS\_DataBase.cs | 数据访问类 |
| Regedit.aspx | 注册页面 |

在这些页面中页首导航条、页边导航条与页尾导航条同时位于母版页中BBS系统中所有页面的模板，即所有的页面都是在该模板的基础上编写的，网站地图用来表示网站的结构，在页边与页尾的导航控件中被用来表明用户的位置。

首页为welcome.aspx页面

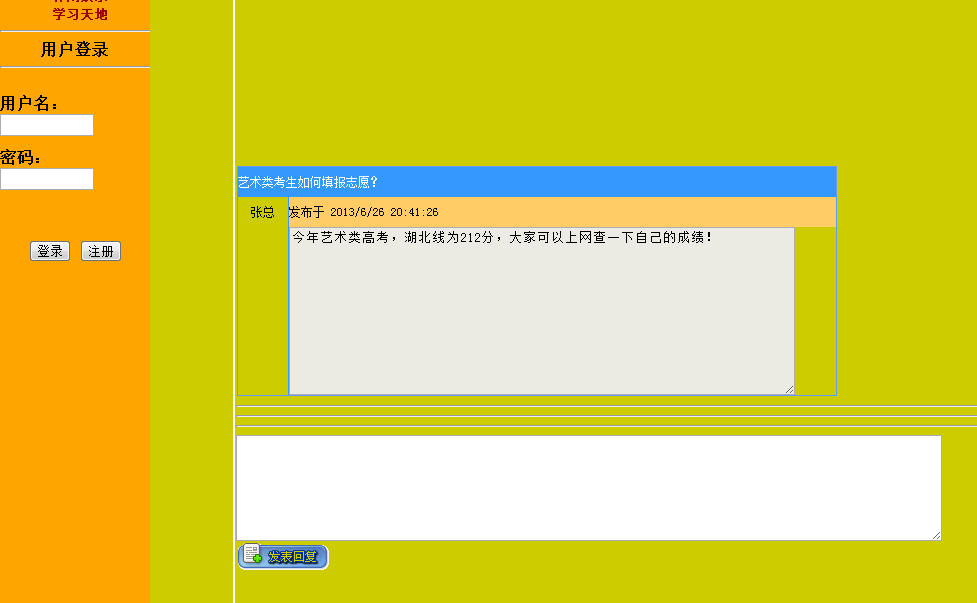


发帖管理模块

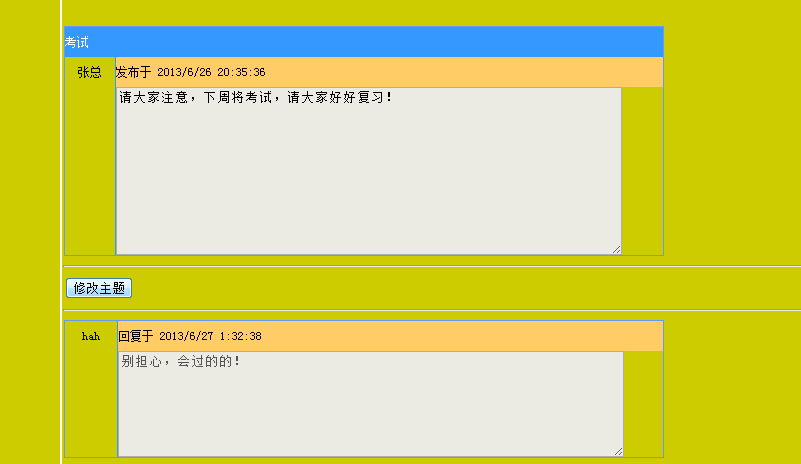
点击分区的标题如“站务管理”，用户添加“新帖”



点击其中某一项帖子，进行回复：

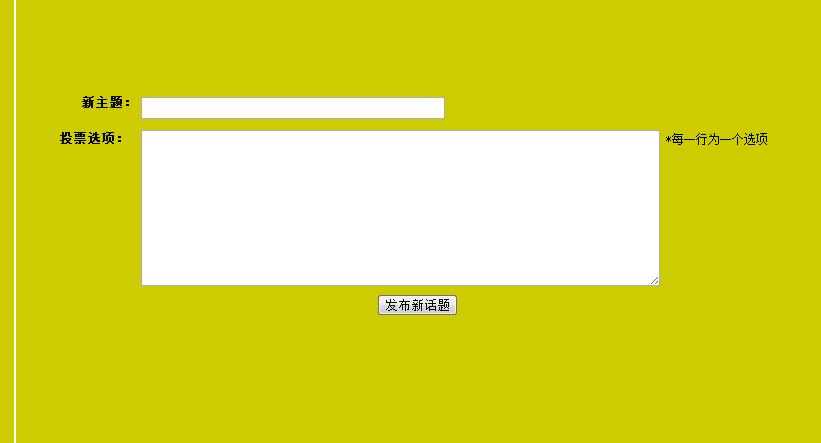


用户自己发的帖子自己可以进行修改：

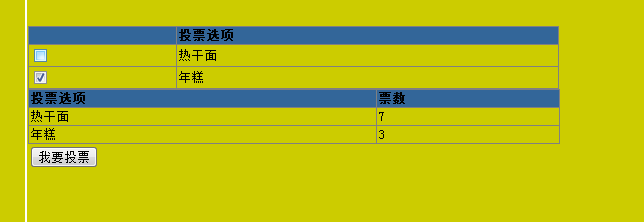


某一个用户为了对某一个话题进行调查，可以对某一个事件发起一个投票：

增加投票项：



选择某个话题后，可以进行投票：



管理员管理模块：

发帖管理：



实际上管理员还可以进行用户管理，删除或者添加用户。

个人信息管理：

