# 一、注册登录

1、起始页:Q\_Zone.aspx是起始页，包括用户登录、用户注册、验证码、自动登录、忘记密码。

1.1用户注册：点击注册后，进入Register.aspx页面。

1.2用户登录：触发事件后，对填入的密码进行哈希，执行sql查询语句，与数据库对比。

判断是否存在此用户。并对验证码对比，判断是否正确。

代码如下;

string mail = txtUserMail.Text;

string pwd = Md5Hash(txtUserPwd.Text) ;

string[] column = new string[2] { "mail", "pwd" };

string[] columnvalue = new string[2] { mail, pwd };

string sql = "select \* from Users ";

sql = sql + " where userMail =@" + column[0] + " and userPwd =@" + column[1];

System.Data.DataTable dt = new DataTable();

dt = SqlHelper.SqlSelect(sql,column,columnvalue,2);

1.3忘记密码：触发此事件后，进入UserFind.aspx页面。

1.4自动登录：触发此事件后，在登录成功后保存用户id，保存在名为Mycook的cooki里。等到用户再次加载此页面时，判断Mycook是否为null,不为空则进入。

代码如下：

HttpCookie cookie = new HttpCookie("Mycook"); //初始化cooki

DateTime time = DateTime.Now;

cookie.Expires = time.AddDays(3); //设置过期时间

cookie.Values.Add("userid", Session["userID"].ToString());

cookie.Values.Add("username", Session["userName"].ToString());

cookie.Values.Add("userpic", Session["userPic"].ToString());

Response.AppendCookie(cookie);

2.Register.aspx:

该页面是注册页面。

首先调用验证码生成函数，其次点击注册，判断所输入的信息格式，无误后发送验证码。

2.1 验证码生成函数，该函数返回验证码

代码如下：

private string validateCodeType = "0";

/// <summary>

/// 验证码字符个数

/// </summary>

private int validateCodeCount = 4;

/// <summary>

/// 验证码的字符集，去掉了一些容易混淆的字符

/// </summary>

char[] character = { '2', '3', '4', '5', '6', '8', '9', 'A', 'B', 'C', 'D', 'E', 'F', 'G', 'H', 'J', 'K', 'L', 'M', 'N', 'P', 'R', 'S', 'T', 'W', 'X', 'Y' };

private string GenerateCheckCode()

{

char code;

string checkCode = String.Empty;

System.Random random = new Random();

for (int i = 0; i < validateCodeCount; i++)

{

code = character[random.Next(character.Length)];

// 要求全为数字或字母

if (validateCodeType == "1")

{

if ((int)code < 48 || (int)code > 57)

{

i--;

continue;

}

}

else if (validateCodeType == "2")

{

if ((int)code < 65 || (int)code > 90)

{

i--;

continue;

}

}

checkCode += code;

}

Response.Cookies.Add(new System.Web.HttpCookie("CheckCode", checkCode));

Session["CheckCode"] = checkCode;

return checkCode;

}

2.2判断内容格式

代码如下：

string emailStr = @"^[\u4E00-\u9FA5A-Za-z0-9\_]+$";

Regex emailReg = new Regex(emailStr);

if (emailReg.IsMatch(userName.Text.Trim())) //判断邮箱格式

{

}

else

{

flag = 8;

}

string selStr = "select \* from Users where userMail=@"; //查看数据库中是否已注册

selStr = selStr + column[0];

DataTable dt = new DataTable();

dt = SqlHelper.SqlSelect(selStr,column,columnvalue,1);

if (dt.Rows.Count != 0)

{

flag = 2;

}

if (name.IndexOf(" ") > -1 || password.IndexOf(" ") > -1) //判断是否有空格

{

flag = 3;

}

if(password != passwords) //判断前后密码是否一致

{

flag = 4;

}

if (lenth < 4 || lenth > 20) //判断用户名和密码长度

{

flag = 6;

}

if (lenths < 4 || lenths >20)

{

flag = 7;

}

if (lenth == 0 || lenths == 0)

{

flag = 0;

}

Regex reg = new Regex(@"^\w+$"); //判断密码格式

if (reg.IsMatch(password))

{

}

else

{

flag = 9;

}

2.3发送邮件，

代码如下：

string txtEmailAddress = userMail.Text;

string validCode = GenerateCheckCode(); //获取系统生成的验证码

string smtpServer = "smtp.qq.com"; //SMTP服务器

string mailFrom = "330953853@qq.com"; //登陆用户名

string userPassword = "ulldpxudytplcbee";//客户端授权码

string mailTo = txtEmailAddress;

string mailSubject = "验证码";

string mailContent = "你的验证码是" + validCode;

SmtpClient smtpClient = new SmtpClient();

smtpClient.DeliveryMethod = SmtpDeliveryMethod.Network;//指定电子邮件发送方式

smtpClient.Host = smtpServer; //指定SMTP服务器

smtpClient.UseDefaultCredentials = false;

smtpClient.EnableSsl = true;

smtpClient.Credentials = new System.Net.NetworkCredential(mailFrom, userPassword);//用户名和密码

MailMessage mailMessage = new MailMessage(mailFrom, mailTo); // 发送人和收件人

mailMessage.Subject = mailSubject;//主题

mailMessage.Body = mailContent;//内容

mailMessage.BodyEncoding = Encoding.UTF8;//正文编码

mailMessage.IsBodyHtml = false;//设置为HTML格式

mailMessage.Priority = MailPriority.Normal;//优先级

smtpClient.Send(mailMessage); // 发送邮件

2.4 注册之数据库

输入验证码后判断输入正确后写入数据库

SQL使用了参数化查询，在此介绍一下。

例如：

string[] column = new string[3] { "name","mail","password" };

string[] columnvalue = new string[3] { name,mail,password };

string sql = "insert into Users(userName ,userMail,userPwd,picture) values(@";

sql = sql + column[0] + ",@" + column[1] + ",@" + column[2] + ",'" + picture + "')";

SqlHelper.InsertInto(sql,column,columnvalue,3);  
SqlHelper.InsertInto使用了封装，sql是SQL语句，column和columnvalue是字符串数组。

3代表column有三个变量需要声明。该段代码写入了用户的基本信息。

2.5 获取用户id

string[] column2 = new string[1] { "name1" };

string[] columnvalue2 = new string[1] { name };

string sql2 = "select \* from Users where userName =@";

sql2 = sql2 + column2[0];

System.Data.DataTable dt = new DataTable();

dt = SqlHelper.SqlSelect(sql2,column2,columnvalue2,1);

Session["username"] = userName.Text;

string id = dt.Rows[0][0].ToString();

2.6 自动填写

//用户个人信息，默认为未填写。

string[] column4 = new string[2] { "R\_userid","nothing" };

string[] columnvalue4 = new string[2] {Session["id"].ToString(), "未填写" };

string sql4 = "insert into UserInformation(userID,sex,age,birthday,blood,marry,adress,job,hobby,constellation,hometown) values(@";

sql4 = sql4 + column4[0] + ",@" + column4[1] + ",@" + column4[1] + ",@" + column4[1] + ",@" + column4[1] + ",@" + column4[1] + ",@" + column4[1] + ",@" + column4[1] + ",@" + column4[1] + ",@" + column4[1] + ",@"+column4[1]+")";

SqlHelper.InsertInto(sql4, column4, columnvalue4, 2);

//默认自己为自己好友。

string[] column3 = new string[2] { "userID","friendID" };

string[] columnvalue3 = new string[2] { id,id };

string sql3 = "insert into UserFriend(userID,friendID) values('" + id + "','" + id + "')";

sql = sql + column3[0] + ",@" + column3[1] + ")";

SqlHelper.InsertInto(sql3,column3,columnvalue3,2);

//提示注册成功

Response.Write("<script>alert('注册成功！');location='Q\_Zone.aspx'</script>");

# 二、用户使用

母版页共有两个，分别为MasterPage.master和VisitFriend.master。

Master Page.asps为用户个人空间母版页，VisitFriend.master为访问好友空间母版页。

1.1MasterPage.aspx

用户头像显示，昵称显示。

主页、日志、相册、好友、个人档案、留言板、设置等按钮。

点击分别进入各自页面。

1.2判断用户是否登录

代码如下：

if (Session["userID"] == null)

{

Response.Redirect("Wrong.aspx");

}

1.3主页

查询用户及好友说说

代码如下：

string userID = Session["userID"].ToString();

string[] column = new string[1] { "userID4" };

string[] columnvalue = new string[1] { userID };

string sql = "select text,time,userName,picture,userId,talkID from (select talkID,userId,text,time from Talk where userId in(select friendID from Users inner join UserFriend on UserFriend.userID=@";

sql = sql + column[0] + " and Users.id=UserFriend.userID)) as tk inner join Users on tk.userId = Users.id Order By talkID Desc";

System.Data.DataTable dt = new DataTable();

Session["dt"] = SqlHelper.SqlSelect(sql,column,columnvalue,1);

1.4说说分页显示及权限

说说设置查看权限

代码如下

void DataBindToRepeater(int currentPage)

{

DataTable dt = (DataTable)Session["dt"];

if(dt.Rows.Count != 0)

{

int count = Convert.ToInt32(dt.Rows.Count);

for(int i = 0;i<count;i++)

{

dt.Rows[i][0] = Server.HtmlEncode(dt.Rows[i][0].ToString());

}

for(int i = 0;i<count;i++)

{

string userid = dt.Rows[i][4].ToString();

string sql = "select \* from Permission where userid = '" + userid + "'";

DataTable da = SqlHelper.SqlSelect(sql, null, null, 0);

string talkType = da.Rows[0][2].ToString();

if (talkType == "仅好友")

{

string sql2 = "select \* from UserFriend where userID = '" + userid + "' and friendID = '" + Session["userID"].ToString() + "'";

DataTable ds = SqlHelper.SqlSelect(sql2, null, null, 0);

if(ds.Rows.Count == 0)

{

dt.Rows[i].Delete();

}

}

if(talkType == "仅自己")

{

if(Session["userID"].ToString() != userid)

{

dt.Rows[i].Delete();

}

}

}

}

PagedDataSource pds = new PagedDataSource();

pds.AllowPaging = true;

pds.PageSize = 5;

pds.DataSource = dt.DefaultView;

lbTotal.Text = pds.PageCount.ToString();

pds.CurrentPageIndex = currentPage - 1;//当前页数从零开始，故把接受的数减一

talklist.DataSource = pds;

talklist.DataBind();

}

1.5评论及取消评论

代码如下：

switch (e.CommandName)

{

case "reply": //点击评论按钮

TextBox replyEdit = (TextBox)e.Item.FindControl("replyEdit");

Button button = (Button)e.Item.FindControl("submit");

Button back = (Button)e.Item.FindControl("back");

Label talk = (Label)e.Item.FindControl("talkid");

replyEdit.Visible = true; //显示评论框

button.Visible = true; //显示发表按钮

back.Visible = true; //显示取消按钮

Session["userids"] = e.CommandArgument.ToString();

Session["talkid"] = talk.Text;

break;

case "back":

TextBox replyEdit2 = (TextBox)e.Item.FindControl("replyEdit");

Button button2 = (Button)e.Item.FindControl("submit");

Button back2 = (Button)e.Item.FindControl("back");

replyEdit2.Visible = false;

button2.Visible = false;

back2.Visible = false;

break;

default:

break;

}

1.6日志、相册

日志和相册查询代码及分页与上文类似

1.7添加好友、好友申请

添加好友使用了模糊搜索

代码如下：

string userName = btn\_search.Text;

string[] column = new string[1] { "userName1" };

string[] columnvalue = new string[1] { "%"+userName+"%" };

string sql = "select \* from Users where userName like ";

sql = sql + "@" + column[0];

DataTable dt = SqlHelper.SqlSelect(sql,column,columnvalue,1);

点击添加后开始申请。

string[] column2 = new string[2] { "mail","friendid" };

string[] columnvalue2 = new string[2] { userid,friendid };

string sql2 = "insert into FriendPass(userID,friendID) values('"+ userid +"','"+ friendid +"')";

sql = sql + "(@" + column2[0] + ",@" + column2[1] + ")";

SqlHelper.InsertInto(sql2,column2,columnvalue2,2);

Response.Write("<script>alert('申请中！')</script>");

2、VisitFriend.master

2.1主页

仅显示好友说说，代码类似。

2.2日志、相册、好友档案代码类似，仅缺少部分功能。

三、数据库介绍

Album表：

albumid:相册id

image:图片路径

Album\_Cover:

albumName:相册名称

introduce:相册介绍

picture:图片路径

blog: 日志相关信息

blogReplies:日志评论相关信息

FriendPass:好友申请表

LiuYan:留言表

Permission:权限表

PhotoComment:照片评论

Talk:说说

TalkReplies：说说评论

UserFriend:好友关系列表

UserInformation：用户个人档案表

Users：用户基本信息