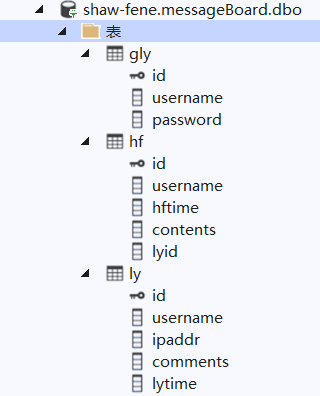
# 数据库部分

## 创建数据库

使用 SQL Server 代替 Access，表结构不变。其中要注意：

* 设置 id 为主键
* 设置 id 为标识列
* 设置lyid 为外键，对应留言的 id 。
* 时间的类型为 datetime ，如 2022-04-11 09:05:06 。





## 连接数据库

1. web.config
   1. VS 数据库连接工具测试：菜单>工具>连接到数据库>
   2. 选择 SQL Server 数据源，然后选择 SQL Server 数据库文件，点击测试连接。
   3. 将连接字符串放在web.config文件中，代码如下：

<connectionStrings>  
 <add name="sqlConnect" connectionString="Data Source=SHAW-FENE;Initial Catalog=messageBoard;Integrated Security=True"/>  
 </connectionStrings>

1. DBHelper.cs
   1. 将原有的命名空间注释，使用新的命名空间。
   2. 更换变量 connStr 中字符串。
   3. 并且将所有原有的类替换为 SqlClient 中的类。

using System;  
using System.Collections.Generic;  
using System.Linq;  
using System.Web;  
using System.Data;  
//using System.Data.OleDb;  
using System.Configuration;  
using System.Data.SqlClient;//SQL server数据提供程序  
  
namespace lyban.App\_Code  
{  
 public class DBHelper  
 {  
 private static readonly string connStr = ConfigurationManager.ConnectionStrings["sqlConnect"].ConnectionString;  
 private SqlConnection conn;//负责打开或连接数据库文件  
 private SqlCommand cmd;//对数据库下达行动查询指令，以及执行SQL Server的存储过程  
 // 构造函数  
 public DBHelper() { }  
  
 public SqlConnection getConn()  
 {  
 conn = new SqlConnection(connStr);  
 return conn;  
 }

# 前后端交互部分

* 回复成功之后清空回复框。
* 优化按钮显示样式。

## 留言

1. 显示数据库中已有留言
   * init.ashx 修改 SQL 语句等

* public void ProcessRequest(HttpContext context)  
  {  
   DBHelper dh = new DBHelper();  
   // 选择留言表的所有数据  
   string sql = "select \* from ly";  
   SqlDataReader reader = dh.queryDb(sql);  
   List<message> lylist = new List<message>();  
   // 判断表是否为空  
   if (reader.HasRows)  
   {  
   while (reader.Read())  
   {  
   message tmp = new message();  
   tmp.id = (int)reader[0];  
   tmp.username = (string)reader[1];  
   tmp.ipaddr = (string)reader[2];  
   tmp.comments = (string)reader[3];  
   tmp.lytime = Convert.ToDateTime(reader[4]);  
   lylist.Add(tmp);  
   }  
   // 关闭数据库连接，释放资源  
   dh.closeDb();  
   // 把留言转换成JSON格式  
   string res = JsonConvert.SerializeObject(lylist);  
   context.Response.Write(res);  
   }  
   else   
   {  
   context.Response.Write("error");  
   }  
  }

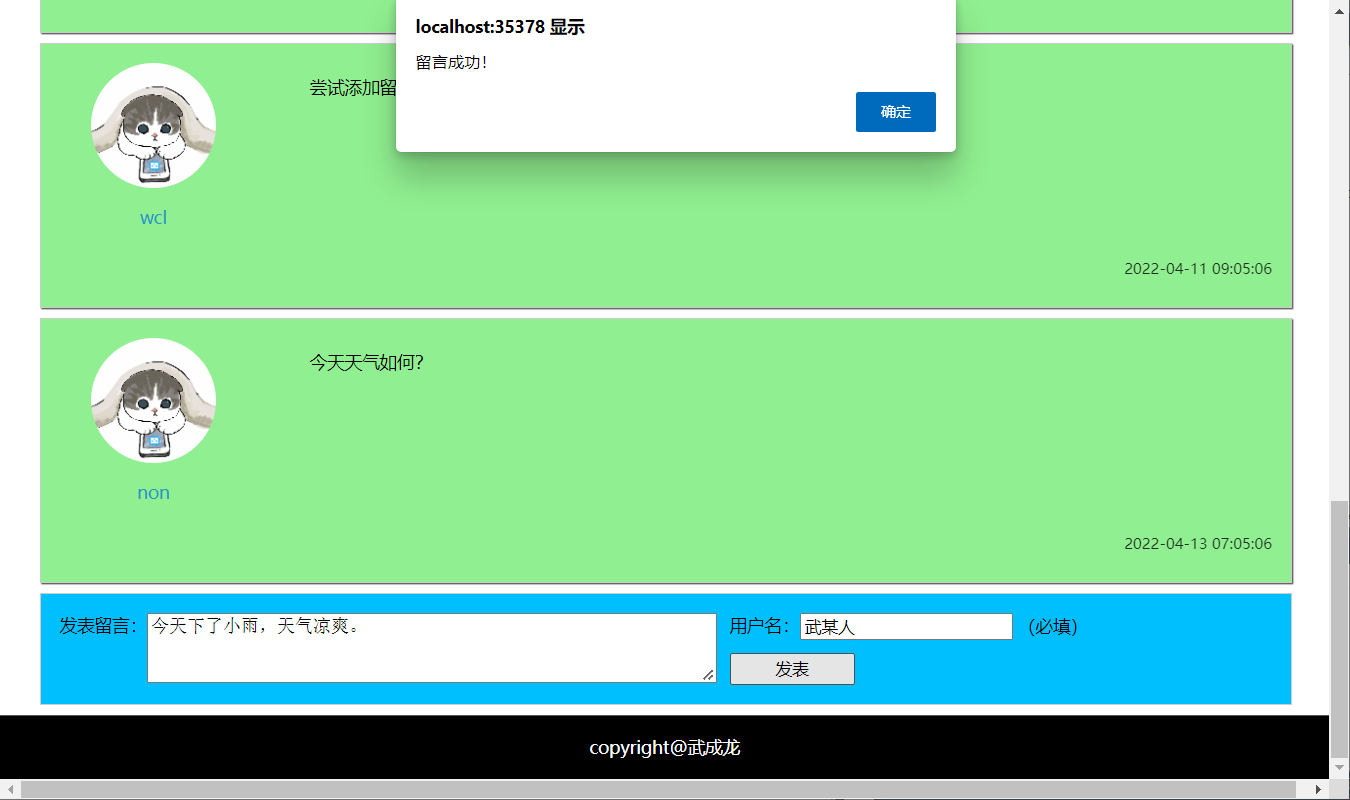
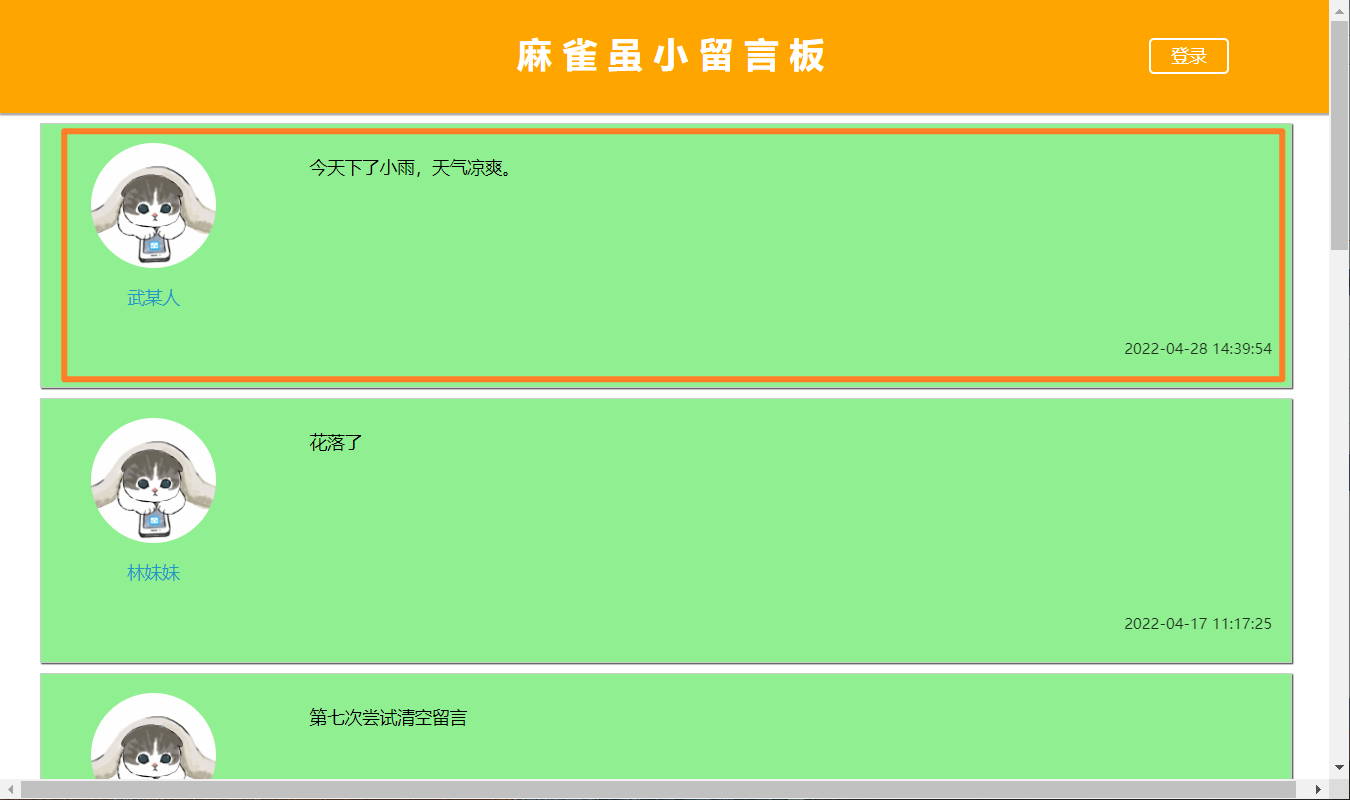
1. 存放新留言到数据库中
   * handleMsg.ashx 修改 SQL 语句等

* public void ProcessRequest(HttpContext context)  
  {  
   if (context.Request["username"] != null && context.Request["comments"] != null)  
   {  
   // 假如用户名和留言不为空，就创建一个 message 对象。  
   message ly = new message();  
   ly.username = context.Request["username"];  
   ly.comments = context.Request["comments"];  
   ly.ipaddr = context.Request.UserHostAddress=="::1"?"127.0.0.1":context.Request.UserHostAddress;  
   ly.lytime = DateTime.Now;   
   DBHelper dh = new DBHelper();  
   // 定义SQL语句  
   string sql = "insert into ly (username,ipaddr,comments,lytime) values ( '" +  
   ly.username + "','" + ly.ipaddr + "','" + ly.comments + "','" + ly.lytime.ToString() + " ')";  
   // 执行SQL语句  
   int res =dh.ExeCurd(sql);  
   if (res > 0)  
   {  
   context.Response.Write("success");  
   }  
   }  
  }

1. 留言成功后清空留言窗口
   * index.js

* function sendMsg(msg) {  
   $.ajax({  
   type:"POST",  
   url: "handleMsg.ashx",  
   data: msg,  
   success: function (res,status,xhr) {  
   if (res = "success") {   
   alert("留言成功！");  
   //添加留言  
   var comments = $(".comments").eq(0).clone();  
   comments.find(".uname").find("label").text($("#uname2").val());  
   comments.find(".ucontents").html($("#ucontents2").val());  
   comments.find("#lytime").text(msg.lytime);  
   $(".contents").prepend(comments);  
   //清空留言窗口内容  
   var message = $("#message").find("#ucontents2");  
   message.val("");  
   }   
   }  
   })  
  }

1. 结果展示

* 
* 
* 

## 登录

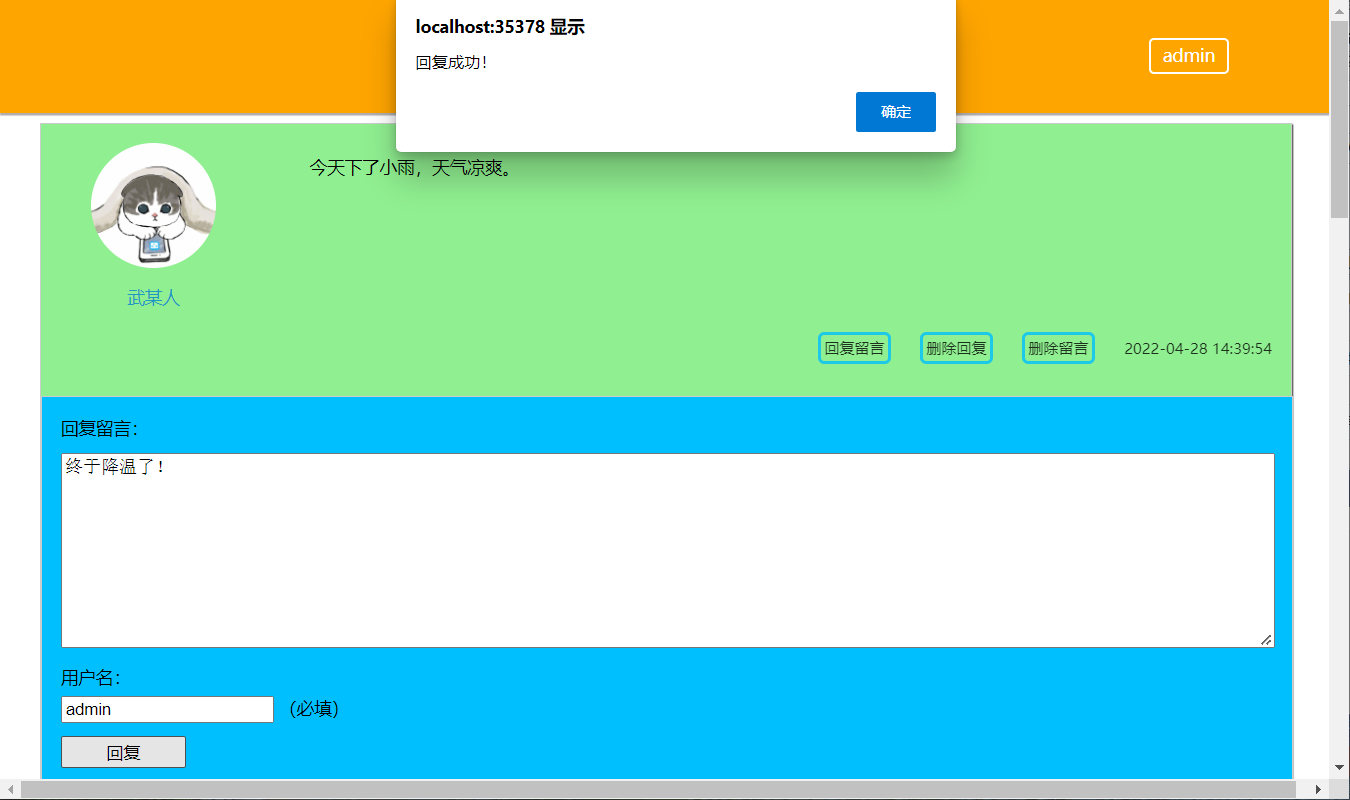
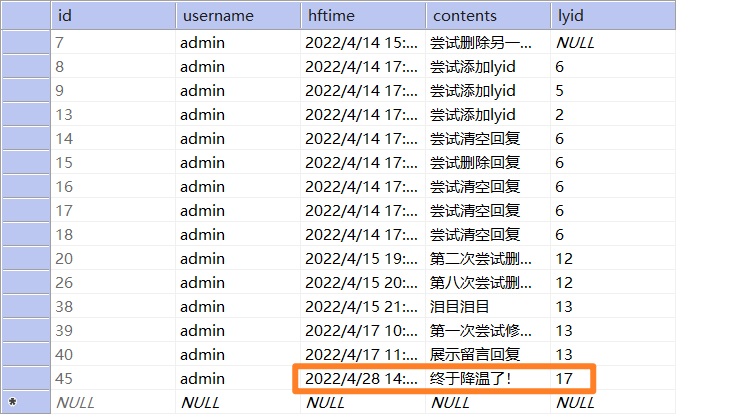
1. 修改样式
2. 添加清空文本函数 clearText() ，绑定在“取消”按钮上。

## 回复

1. 存放新回复到数据库中
   * handleHf.ashx 修改 SQL 语句等
2. 将回复放在 reply-container 块中，方便删除。回复成功后清空回复窗口。
   * manager.js

* function sendHf(msg) {  
   $.ajax({  
   type: "POST",  
   url: "handleHf.ashx",  
   data: msg,   
   success: function (res, status, xhr) {  
   if (res = "success") {  
   alert("回复成功！");  
   //显示回复  
   $("#hf").parent().find(".ucontents").append("<div class='reply-container'><p style='font-size:12px;opacity:0.8;'>"  
   + msg.hfContents + "</p>" + "<label style='font-size:12px;opacity:0.5;'>" + msg.hftime + "</label></div>");  
   //回复存值  
   $("#hf").parent().find(".reply-container").data("lyid", msg.lyid);  
   //清空回复  
   $("#hf").find("#hfContents").val("");  
   //隐藏回复区  
   $("#hf").css("display", "none");  
   }  
   }  
   })  
  }

1. 结果展示

* 
* 
* 

## 删除留言

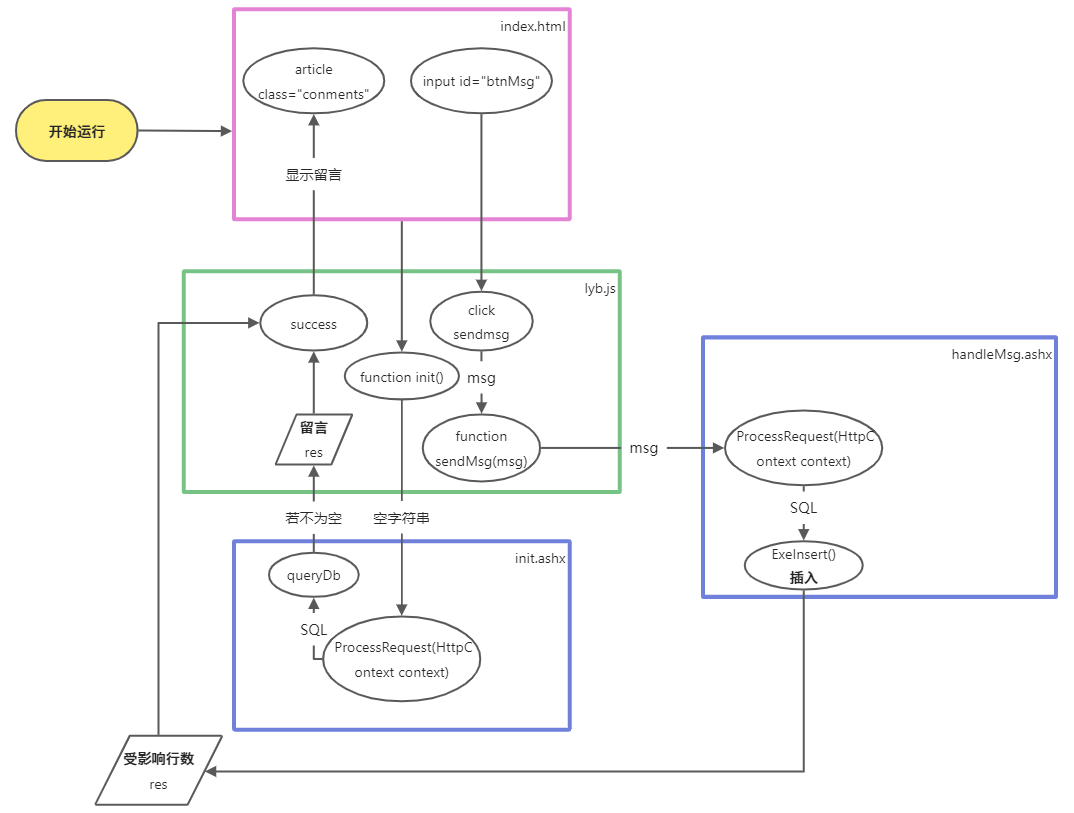
1. 改变点击函数，传递 this 参数。
   * Manager.aspx
   * <label class="rp" onclick="delReplyHandle(this)">删除回复</label>
2. 类似 sendHf.ashx 用 ajax 传递信息给函数，打开数据库连接处理数据。
   1. manager.js
   * //删除留言  
     function delLyHandle(e) {  
      if (confirm("确定要删除留言吗?") == true) {  
      //找到对应的元素  
      const id = $(e).parent().parent().parent().find(".left").find(".h").html();  
      deleteMsg({ ID:id }, e)  
      }  
     }  
     //删除留言  
     function deleteMsg(msgId, e) {  
      $.ajax({  
      type: "POST",  
      url: "deleteMsg.ashx",  
      data: msgId,  
      success: function (res, status, xhr) {  
      if (res = "success") {  
      $(e).parent().parent().parent().remove();  
      alert("删除成功! ");  
      }  
      }  
      })  
     }
   1. deleteMsg.ashx
   * public void ProcessRequest(HttpContext context)  
     {  
      if (context.Request["ID"] != null)  
      {  
      // 假如ID不为空  
      string delID = context.Request["ID"];  
      DBHelper dh = new DBHelper();  
      // 定义SQL语句  
      string sql = "DELETE FROM ly WHERE id = " + delID;  
      // 执行SQL语句  
      int res = dh.ExeCurd(sql);  
      if (res > 0)  
      {  
      context.Response.Write("success");  
      }  
      }  
      else  
      {  
      context.Response.Write("error");  
      }  
       
     }
3. 结果展示

* 
* 

## 删除回复

1. 改变点击函数，传递 this 参数。
   * Manager.aspx
   * <label class="rp" onclick="delLyHandle(this)">删除留言</label>
2. 类似 sendHf.ashx 用 ajax 传递信息给函数，打开数据库连接处理数据。
   1. manager.js
   * function delReplyHandle(e) {  
      if (confirm("确定要删除所有回复吗? ") == true) {  
      //找到对应的元素  
      var lyid = $(e).parent().parent().parent().find(".reply-container").data("lyid");  
      deleteHf({ lyid }, e);  
      }  
     }  
     //删除回复  
     function deleteHf(lyid, e) {  
      $.ajax({  
      type: "POST",  
      url: "deleteHf.ashx",  
      data: lyid,  
      success: function (res, status, xhr) {  
      if (res = "success") {  
      $(e).parent().parent().parent().find(".reply-container").text("");  
      alert("删除成功! ");  
      }  
      }  
      })  
     }
   1. deleteHf.ashx
   * public void ProcessRequest(HttpContext context)  
     {  
      if (context.Request["lyid"] != null)  
      {  
      // 假如ID不为空  
      string delID = context.Request["lyid"];  
      DBHelper dh = new DBHelper();  
      // 定义SQL语句  
      string sql = "DELETE FROM hf WHERE lyid = " + delID;  
      // 执行SQL语句  
      int res = dh.ExeCurd(sql);  
      if (res > 0)  
      {  
      context.Response.Write("success");  
      }  
      }  
      else  
      {  
      context.Response.Write("error");  
      }  
       
     }
   1. 结果展示
   * 
   * 

# 实验小结

以 index.html 加载函数和留言函数为例分析程序前后端交互过程：  


## 当运行程序时，会发生什么？

首先加载 index.html ，然后 index.js 开始执行 init() 函数，即初始化。把 XMLHttpRequest 对象（这里是空字符串）发送到 init.ashx 。

然后开始处理请求，创建 DBHelper 对象，其中构造函数为空。调用 queryDb() 方法，返回一个 SqlDataReader 对象。如果受影响行不为0，即表中有数据，则逐行把数据表读取到一个 message 类的对象 tmp 中，返回一个 res 变量。

最后回到init() 函数，执行回调函数 success ，把已有的留言添加到页面上。

## 如果点击登录，会发生什么？

跳转到 login.aspx ，按下登录按钮就会执行 loadButton() 函数，创建 DBHelper 对象执行查询。  
若用户名和密码正确，会把用户名存放在 Session 中，最后用 Response 重定向到 Manager.aspx 页面。