北方民族大学

学士学位论文

论文题目 基于.NET的学生德育评分管理系统设计与实现

系(部)名 称: 计算机科学与工程学院

学 生 姓 名:

专 业: 软件工程 学 号:

指导教师姓名:

论文提交时间:

论文答辩时间:

学位授予时间:

北方民族大学教务处制

基于.NET的学生德育评分管理系统设计与实现

摘 要

学生德育评分管理系统实现对学生德育信息的管理及实时更新操作，该系统包括以下模块，学生基本信息管理，学生处分管理，学生IPQ增加管理，学生IPQ减少管理，每周考勤管理，每周宿舍检查结果管理。

本系统采用Visual studio 2005进行开发，使用SQL Server 2005作为后台的数据库。

经过测试，本系统基本能够完成所要求的各项功能，实现了对学生IPQ、考勤、处分、宿舍卫生检查结果的增删查改，以及对学生基本信息的管理。该系统的使用能方便辅导员及时掌握学生情况，提高了工作效率。

关键词：德育评分系统，dotnet，IPQ，学生考勤 ，SQL Server

Student IPQ System Based on DotNet

**Abstract**

The Student IPQ System can update student dynamic information on time. The function of the system contains Student information management,Student punish managent,Student AddIPQ managent,Stdent SubtractionIPQ management,Student Attendance management, Student Health management .

The system using Visual Basic 2005 development, the use of SQL Server 2005 database as a background.

After testing, the system can be completed by the basic requirements of the various functions; can be effective management of Student IPQ information.the use of the system can help teacher know more about student , and it can improve the work efficiency of them .

**Key words**: Student IPQ system Base on DotNet,DotNet,IPQ,Attendance, SQL Server

目录

[1 前 言 1](#_Toc292389047)

[2 开发背景及开发环境 1](#_Toc292389048)

[2.1项目研发背景及趋势 1](#_Toc292389049)

[2.2系统运行环境 2](#_Toc292389050)

[2.2.1硬件环境 2](#_Toc292389051)

[2.2.2软件环境 2](#_Toc292389052)

[2.3 开发环境 2](#_Toc292389053)

[2.3.1VB简介 2](#_Toc292389054)

[2.3.2SQL及SQLSERVER 3](#_Toc292389055)

[3 学生德育评分管理系统概述 3](#_Toc292389056)

[3.1系统开发模型 3](#_Toc292389057)

[3.2系统体系结构 4](#_Toc292389058)

[3.3系统功能结构 4](#_Toc292389059)

[3.4系统功能简介 5](#_Toc292389060)

[4 学生德育评分管理系统需求分析 6](#_Toc292389061)

[4.1系统需求分析 6](#_Toc292389062)

[4.2数据库需求分析 8](#_Toc292389063)

[4.2.1数据库建模 8](#_Toc292389064)

[4.2.2 数据字典 9](#_Toc292389065)

[4.2.3数据库脚本 10](#_Toc292389066)

[5 学生德育评价管理系统设计与实现 11](#_Toc292389067)

[5.1 DataAccessLayer 数据访问层 11](#_Toc292389068)

[5.2 DataAccessHelper 数据访问接口层 12](#_Toc292389069)

[5.3 BussinessLogicLayer 业务逻辑类 14](#_Toc292389070)

[5.4 功能列表 15](#_Toc292389071)

[5.5菜单树用户控件 17](#_Toc292389072)

[5.6模板 18](#_Toc292389073)

[5.7样式表 19](#_Toc292389074)

[5.8表示层 22](#_Toc292389075)

[5.9登录页面 27](#_Toc292389076)

[6 系统的测试 31](#_Toc292389077)

[6.1 系统测试目的和原则 31](#_Toc292389078)

[6.2 系统测试 32](#_Toc292389079)

[结束语 33](#_Toc292389080)

[致 谢 33](#_Toc292389081)

[参考文献 33](#_Toc292389082)

# 1 前 言

随着教育体制的的短改革，现在的大学生拥有的更多的活动空间以及更好的学习条件。同时随着教育普及程度的不断提升，大学扩招规模也在不断扩大，随着而来我们学校的管理就开始显得紧张而且复杂。在这种条件下，对于直接和学生交流的辅导员就肩负着更大的压力和责任，即便如此一个辅导员面对几千学生，往往也显得力不从心，特别是在学生信息收集以及各个阶段的表现情况更是不好掌握。这样就会导致部分学生由于长期处于放松状态或者迷失，而得不到有效的开导。从而不能很好的完成学业。也会给我们学校的管理带来不良影响。

综上所述怎么方便有效的管理学生就是我们学校面临的最重要的一个问题，而作为一种手段，电脑往往会给我们带来很多帮助。所以我们开始着手编写实现一款软件让他能够综合一个学院学生的基本信息，同时通过该生的各方面信息收集准确的了解到学生的近况。这样可以大大提高辅导员老师的办公效率，也能在一定程度上加强老师与学生沟通交流。鉴于此我们尝试去设计这款软件，具体情况如下：

该系统为B/S结构，基于asp.net实现。首先建立一个完善而且包含信息全面（比如：考勤，处分，活动参加情况等）的学生信息数据库，允许对该系统做实时更新操作，以便随时更改学生动态信息（比如，刚参加完一个活动获得奖励，可以及时输入到系统）。在此基础上对系统设置自动报警功能，如：当某些学生的主修课门数超过规定门数，旷课超过规定节数，系统会提醒辅导员注意该生最近这方面情况，以便辅导员及时掌握情况，进行相关辅导工作。

# 2 开发背景及开发环境

## 2.1项目研发背景及趋势

学生德育评分系统大部分是基于CS结构的，而网上BS结构的各种数据库处理系统早已成熟，所以尝试做一款BS的学生德育评分系统。

C/S是Client/Server的缩写。服务器通常采用高性能的PC、工作站或小型机，并采用大型数据库系统，如Oracle、Sybase、Informix或 SQL Server。客户端需要安装专用的客户端软件。

B/S是Brower/Server的缩写，客户机上只要安装一个浏览器（Browser），如Netscape Navigator或Internet Explorer，服务器安装Oracle、Sybase、Informix或 SQL Server等数据库。在这种结构下，用户界面完全通过WWW浏览器实现，一部分事务逻辑在前端实现，但是主要事务逻辑在服务器端实现。浏览器通过Web Server 同数据库进行数据交互。

## 2.2系统运行环境

### 2.2.1硬件环境

服务器：Pentium(R) 4处理器2.93GHz以上/1G RAM/40G硬盘/100M网卡

工作站：Pentium(R) 4处理器2.93GHz以上/256M RAM/20G硬盘/100M网卡/显示器16位真彩色以上，分辨率为1024\*768

### 2.2.2软件环境

服务器端操作系统拟采用WINDOWS 2003 SERVER系列产品。

数据库拟采用SQL SERVER 2005。

客户端操作系统采用WINDOWS系列。

客户端应用软件开发工具采用ASP.NET。

## 2.3 开发环境

### 2.3.1VB简介

Visual Basic（VB）是一种由微软公司开发的包含协助开发环境的事件驱动编程语言。从任何标准来说，VB都是世界上使用人数最多的语言——不仅是盛赞VB的开发者还是抱怨VB的开发者的数量。它源自于BASIC编程语言。VB拥有图形用户界面（GUI）和快速应用程序开发（RAD）系统，可以轻易的使用DAO、RDO、ADO连接数据库，或者轻松的创建ActiveX控件。程序员可以轻松的使用VB提供的组件快速建立一个应用程序。VB 2005的“显著”优点是，可以直接编写出XP风格的按钮，以及其他的控件。但是其编写的小程序占用近10MB的内存。通过几年的发展，它已成为一种专业化的开发语言和环境。用户可用Visual Basic快速创建Windows程序，现在还可以编写企业水平的客户端/服务器程序及强大的数据库应用程序。 　　VB会吸引更多的赞誉和批评，也会继续为广大的用户和程序员使用。它非常适合用来开发拥有友好界面的程序（比如针对终端的数据），但是不适合开发其它程序（比如联合计算程序）。

### 2.3.2SQL及SQLSERVER

SQL(Structured Query Language)，结构化查询语言。SQL语言的主要功能就是同各种数据库建立联系，进行沟通。按照ANSI(美国国家标准协会)的规定，SQL被作为关系型数据库管理系统的标准语言。SQL语句可以用来执行各种各样的操作，例如更新数据库中的数据，从数据库中提取数据等。绝大多数流行的关系型数据库管理系统都采用了SQL语言标准。虽然很多数据库都对SQL语句进行了再开发和扩展，但是包括Select, Insert, Update, Delete, Create,以及Drop在内的标准的SQL命令仍然可以被用来完成几乎所有的数据库操作。

SQL Server 是一个关系数据库管理系统。它最初是由Microsoft、 Sybase 和Ashton-Tate三家公司共同开发的，于1988 年推出了第一个OS/2 版本。在Windows NT 推出后，Microsoft与Sybase 在SQL Server 的开发上就分道扬镳了，Microsoft 将SQL Server 移植到Windows NT系统上，专注于开发推广SQL Server 的Windows NT 版本。Sybase 则较专注于SQL Server在UNIX 操作系统上的应用。

# 3 学生德育评分管理系统概述

## 3.1系统开发模型

本系统采用瀑布模型

瀑布模型（Waterfall Model） 是一个项目开发架构，开发过程是通过设计一系列阶段顺序展开的，从系统需求分析开始直到产品发布和维护，每个阶段都会产生循环反馈，因此，如果有信息未被覆盖或者发现了问题，那么最好 “返回”上一个阶段并进行适当的修改，项目开发进程从一个阶段“流动”到下一个阶段，这也是瀑布模型名称的由来。包括软件工程开发、企业项目开发、产品生产以及市场销售等构造瀑布模型。

1970年温斯顿•罗伊斯（Winston Royce）提出了著名的“瀑布模型”，直到80年代早期，它一直是唯一被广泛采用的软件开发模型。

瀑布模型核心思想是按工序将问题化简，将功能的实现与设计分开，便于分工协作，即采用结构化的分析与设计方法将逻辑实现与物理实现分开。将软件生命周期划分为制定计划、需求分析、软件设计、程序编写、软件测试和运行维护等六个基本活动，并且规定了它们自上而下、相互衔接的固定次序，如同瀑布流水，逐级下落。

## 3.2系统体系结构

本系统采用BS的体系结构。

B／S（Browser／Server）结构，即浏览器服务器结构。它是随着Internet技术的兴起，对C／S结构的一种变化或者改进的结构。在这种结构下，用户工作界面通过浏览器来实现，极少部分事务逻辑在前端（Browser）实现，主要事务逻辑在服务器端（Server）实现，形成所谓三层结构。这样就大大简化了客户端电脑载荷，减轻了系统维护与升级的成本和工作量，降低了用户的总体成本。

## 3.3系统功能结构

根据需求分析的结果，基于Web的学生德育评价管理系统主要分为下面几个模块：管理员基本信息管理模块，学生基本信息管理模块，学生处分模块，IPQ管理模块，每周考勤模块，每周卫生检查结果模块

整个系统的结构如图3.2所示。



图3.2 系统的模块图

## 3.4系统功能简介

以下分别介绍系统中的各个系统6个子模块：

管理员基本信息管理

该模块主要是辅导员对管理员信息的增删查改，管理员在这个模块只有修改自己密码的权限

学生基本信息管理

管理人员和辅导员能对德育评价的学生进行增删查改，学生只能查看自己的基本信息

学生处分管理

管理员或者辅导员能对参加德育评价的学生进行处分的添加修改删除，学生只能查看自己的处分信息

ＩＰＱ管理模块

包括ＩＰＱ增加管理和ＩＰＱ扣分管理，可以查询单个或者所有的学生的ＩＰＱ情况，学生只能查看自己的ＩＰＱ情况

考勤管理

对每周学生考勤的管理，可以通过查询某一周的考勤情况，学生只有查看权限

卫生检查管理

辅导员或者管理员有添加修改删除每周卫生检查结果的权限，学生只能查看每周或者所有的卫生检查结果。

# 4 学生德育评分管理系统需求分析

## 4.1系统需求分析

用例图是被称为参与者的外部用户所能观察到的系统功能的模型图。用例是系统中的一个功能单元，可以被描述为参与者与系统之间的一次交互作用。用例模型的用途是列出系统中的用例和参与者，并显示哪个参与者参与了哪个用例的执行。下图4.1为系统用例图。



图4.1 系统用例图

## 4.2数据库需求分析

### 4.2.1数据库建模

本文使用的是关系型数据库

关系数据库，是建立在关系模型基础上的数据库，借助于集合代数等数学概念和方法来处理数据库中的数据。现实世界中的各种实体以及实体之间的各种联系均用关系模型来表示。关系模型是由埃德加·科德于1970年首先提出的，并配合“科德十二定律”。现如今虽然对此模型有一些批评意见，但它还是数据存储的传统标准。标准数据查询语言SQL就是一种基于关系数据库的语言，这种语言执行对关系数据库中数据的检索和操作。 关系模型由关系数据结构、关系操作集合、关系完整性约束三部分组成。

数据库模型图如图4.2所示。



图4.2 数据库模型图

### 4.2.2 数据字典

下列表4.1到表4.7给出7个数据表：

包括数据描述、数据名称、数据类型等。

表4.1 学生信息表StudentList

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 字段名 | 数据类型 | 长度 | 主外键 | 约束 | 备注 |
| SubtractionId | 自动编号 |  | 主键 | 非空 | 编号 |
| StudentId | 文本 | 8 | 外键 | 非空 | 学号 |
| SubtractionReason | 文本 | 25 |  | 非空 | 扣分原因 |
| SubtractionScore | 整型 |  |  | 非空 | 所扣分值 |

表4.2 IPQ扣分管理表（SubtractionIPQList）

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 字段名 | 数据类型 | 长度 | 主外键 | 约束 | 备注 |
| StudentId | 文本 | 8 | 主键 | 非空 | 学号 |
| StudentName | 文本 | 20 |  | 非空 | 姓名 |
| StudentPassword | 文本 | 20 |  | 非空 | 密码 |
| StudentSex | 文本 | 4 |  | 非空 | 性别 |
| DormitoryType | 文本 | 8 |  | 非空 | 宿舍类型 |
| FloorId | 整型 |  |  |  | 宿舍楼号 |
| DormitoryId | 文本 | 10 |  | 非空 | 宿舍号 |

表4.3 IPQ增加管理表（AddIPQList）

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 字段名 | 数据类型 | 长度 | 主外键 | 约束 | 备注 |
| AddId | 自动编号 |  | 主键 | 非空 | 编号 |
| StudentId | 文本 | 8 | 外键 | 非空 | 学号 |
| AddReason | 文本 | 50 |  | 非空 | 加分原因 |
| AddScore | 整型 |  |  | 非空 | 所加分值 |

表4.4 每周考勤信息表（AttendanceList）

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 字段名 | 数据类型 | 长度 | 主外键 | 约束 | 备注 |
| AttendanceId | 自动编号 |  | 主键 | 非空 | 编号 |
| Week | 文本 | 50 |  | 非空 | 周数 |
| StudentId | 文本 | 8 | 外键 | 非空 | 学号 |
| Semester | 数字 | 整型 |  | 非空 | 学期 |
| StayAway | 数字 | 整型 |  |  | 旷课次数 |
| DesertExercises | 数字 | 整型 |  |  | 旷早操次数 |

表4.5 每周卫生检查表（HealthList）

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 字段名 | | 数据类型 | | 长度 | | 主外键 | | 约束 | | 备注 |
| HealthId | | 自动编号 | | 长整型 | | 主键 | | 非空 | | ID |
| Week | | 数字 | | 整形 | |  | | 非空 | | 周数 |
| DormitoryType | | 文本 | | 8 | |  | | 非空 | | 宿舍类型 |
| FloorId | | 数字 | | 整形 | |  | | 非空 | | 宿舍楼号 |
| DormitoryId | | 文本 | | 10 | |  | | 非空 | | 宿舍号 |
| HealthScore | 数字 | | 整型 | |  | | 非空 | | 得分 | |
| HealthGrade | 文本 | | 4 | |  | | 非空 | | 评价等级 | |

表4.6 账户管理表（UserList）

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 字段名 | 数据类型 | 长度 | 主外键 | 约束 | 备注 |
| UserName | 文本 | 25 | 主键 | 非空 | 用户名 |
| Password | 文本 | 15 |  | 非空 | 密码 |
| Role | 文本 | 25 |  | 非空 | 权限 |

表4.7 学生处分信息表（PunishList）

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 字段名 | 数据类型 | 长度 | 主外键 | 约束 | 备注 |
| PunishId | 数字 |  | 主键 | 非空 | 处分编号 |
| StudentId | 文本 | 8 | 外键 | 非空 | 学号 |
| PunishName | 文本 | 20 |  | 非空 | 处分名称 |
| PunishReason | 文本 | 20 |  | 非空 | 处分原因 |
| PunishDateTime | 日期/时间 |  |  | 非空 | 处分时间 |

### 4.2.3数据库脚本

数据库创建表

--使用系统默认的数据库

use master

go

create database IPQ

go

use IPQ

go

--学生信息表StudentList

CREATE TABLE [dbo].[StudentList] (

[StudentId] [varchar] (8) NOT NULL Primary Key,

[StudentName] [varchar] (20) NOT NULL ,

[StudentPassword] [varchar] (20) NOT NULL ,

[StudentSex] [varchar] (4) NOT NULL ,

[DormitoryType] [varchar] (8) NOT NULL ,

[FloorId] [int] NOT NULL ,

[DormitoryId] [varchar] (10) NOT NULL

)

GO

--外键关联字段

--IPQ扣分管理表（SubtractionIPQList）

ALTER TABLE [dbo].[SubtractionIPQList] ADD

CONSTRAINT [SubtractionIPQList\_StudentList\_fk] FOREIGN KEY

(

[StudentId]

) REFERENCES [dbo].[StudentList] (

[StudentId]

)

GO

数据库表附加数据

USE master

GO

USE IPQ

GO

--学生信息表StudentList

INSERT INTO

[dbo].[StudentList]([StudentId],[StudentName],[StudentPassword],[StudentSex],[DormitoryType],[FloorId],[DormitoryId])

VALUES('20072236','邢成玉','xiner','男','公寓',4,'215')

GO

# 5 学生德育评价管理系统设计与实现

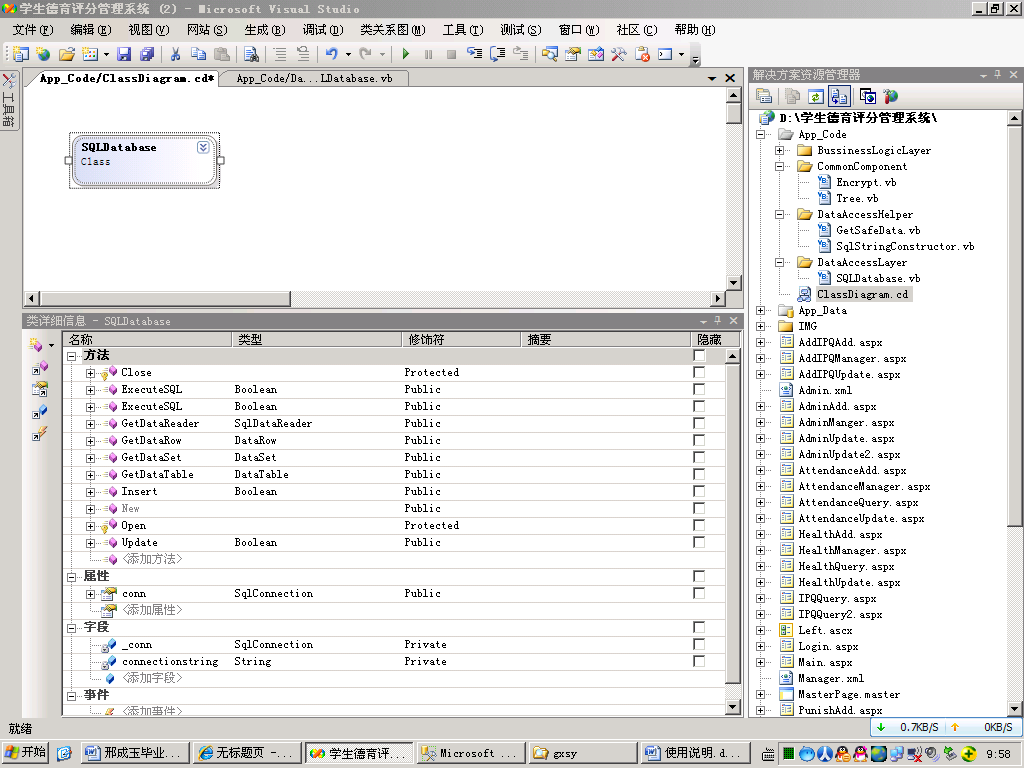
系统采用三层架构

代码主要下面几种

App\_Code文件夹下有下面几个文件夹

## 5.1 DataAccessLayer 数据访问层

数据库类SQLDatabase.vb



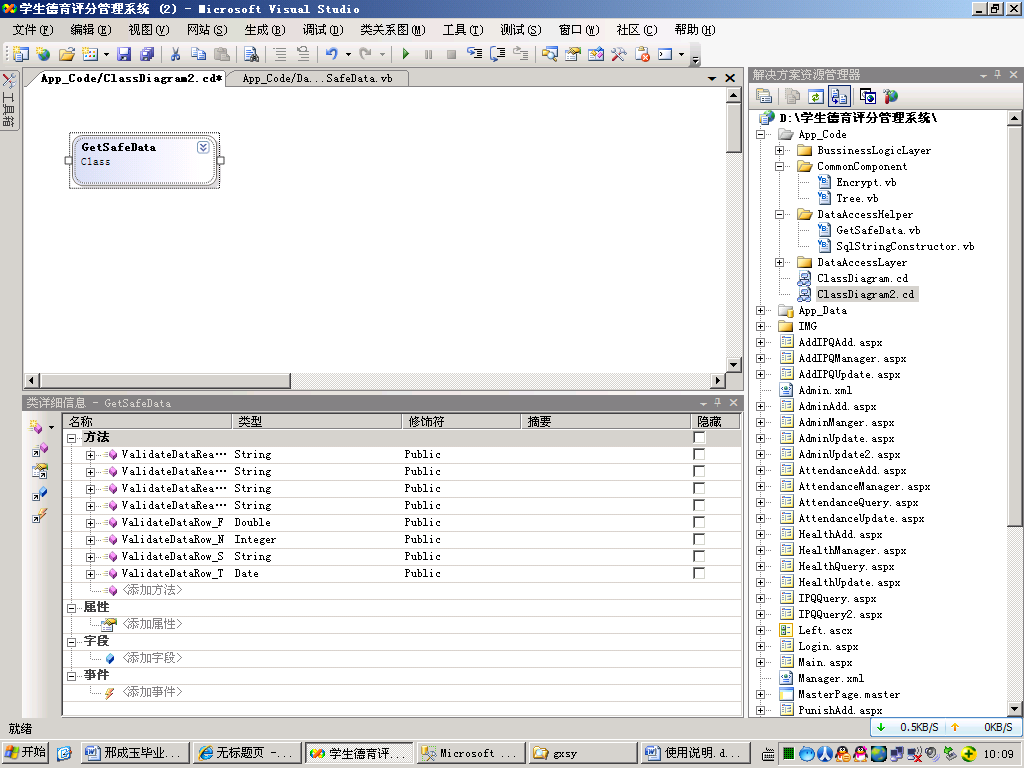
这个类主要的方法和属性如上图，主要有：new（初始化类），Open 和Close（连接和关闭数据库）, ExecuteSQL(执行SQL语句，重载，一个执行一个SQL语句，一个通过哈希表和事务处理一系列的SQL语句，后面删除有外键关联的数据)，GetDataReader，GetDataSet（这些获取静态和动态数据集）,

Insert,Update（插入和更新的函数），属性conn，初始化类的时候从配置文件中获取数据库连接字符串

## 5.2 DataAccessHelper 数据访问接口层

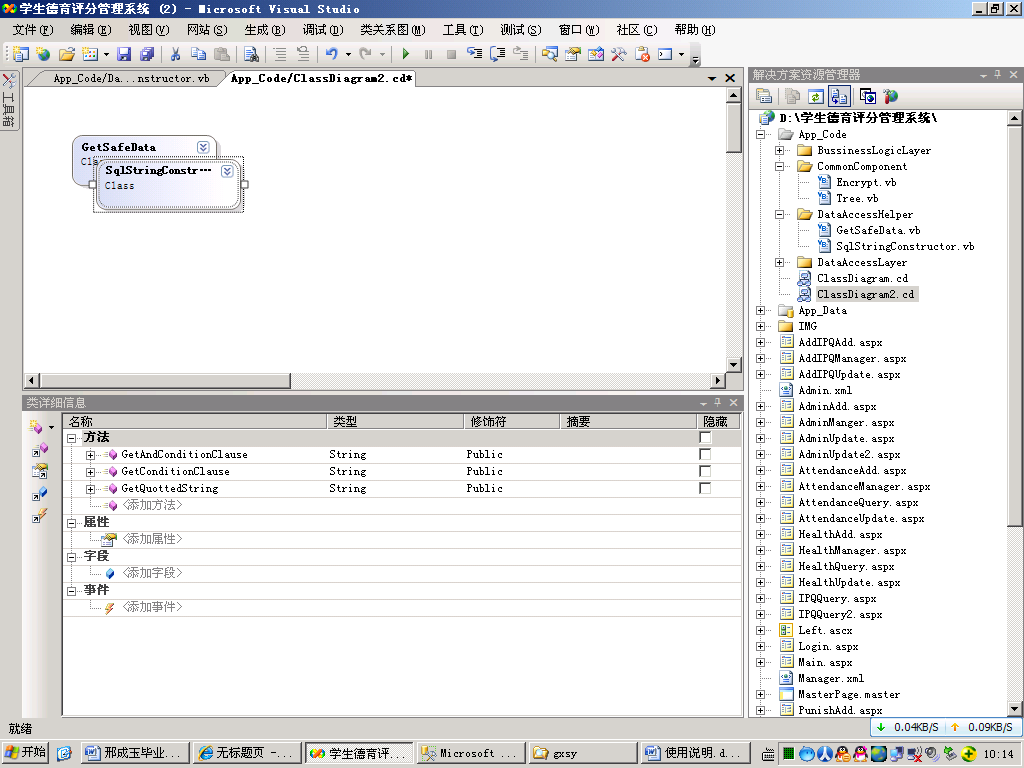
包括获取安全数据类GetSafeData.vb，SQl命令构造器SqlStringConstructor.vb

GetSafeData.vb代码如下



主要保护2种类型的方法，ValidateDataRow\_S和ValidateDataReader\_S,其他的都是修改了下后缀，例如ValidateDataRow\_T,分别代表，s字符串,N整数，F浮点型，T时间类型，从数据库读取或插入数据库时将字符串类型的转换成相应的数据类型，或者将对应的数据类型转化成对应的字符串。

SqlStringConstructor.vb

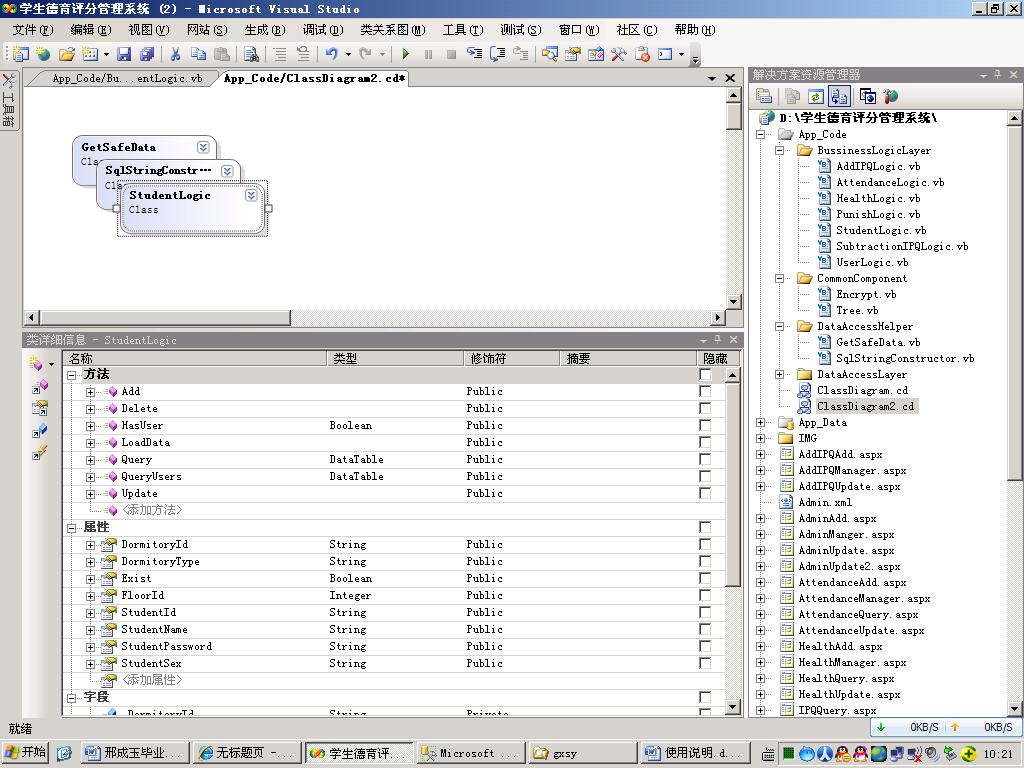


主要包含3个方法GetQuottedString（这个主要是防止别人恶意输入SQL字符串，构造字符串时，使用这个方法）, GetConditionClause(构造不含where的字符串), GetAndConditionClause(参数包括哈希表和where字符串，构造后面带Where的字符串)

## 5.3 BussinessLogicLayer 业务逻辑类

各种实体类 后缀Logic

主要以StudentLogic.vb来解释



属性里主要是学生的一些基本数据，和数据库中的字段是一一对应的。例如数据库脚本中

--学生信息表StudentList

CREATE TABLE [dbo].[StudentList] (

[StudentId] [varchar] (8) NOT NULL Primary Key,

[StudentName] [varchar] (20) NOT NULL ,

[StudentPassword] [varchar] (20) NOT NULL ,

[StudentSex] [varchar] (4) NOT NULL ,

[DormitoryType] [varchar] (8) NOT NULL ,

[FloorId] [int] NOT NULL ,

[DormitoryId] [varchar] (10) NOT NULL

)

GO

方法主要有：LoadData（根据学号查询数据库，获取该学生的所有信息），Add,Update,Delete,Query（对学生信息的增删查改）, HasUser(判断学生是否存在，通过学号查询数据库，如果返回数据表示存在，主要用在添加数据的时候判断学号是否存在)，QueryUsers（通过哈希表构成SQL语句查询数据库），Query（查询所有数据，学生管理界面，页面加载时查询所有学生数据）。其它的业务逻辑方法和这个类似。

## 5.4 功能列表

Manager.xml还有Student.xml,Admin.xml

主要区别是权限不同，功能不同

以Admin.xml的界面举例

<?xml version="1.0" encoding="utf-8" ?>

<lev0 name="后台管理" url="">

<lev1 name="学生信息" url="">

<lev2 name="学生信息查询" url="StudentQuery.aspx"></lev2>

<lev2 name="学生信息管理" url="StudentManager.aspx"></lev2>

</lev1>

<lev1 name="学生处分" url="">

<lev2 name="学生处分浏览" url="PunishQuery.aspx"></lev2>

<lev2 name="学生处分管理" url="PunishManager.aspx"></lev2>

</lev1>

<lev1 name="IPQ" url="">

<lev2 name="IPQ加分查询" url="IPQQuery.aspx"></lev2>

<lev2 name="IPQ加分" url="AddIPQManager.aspx"></lev2>

<lev2 name="IPQ减分" url="SubtractionIPQManager.aspx"></lev2>

</lev1>

<lev1 name="学生考勤" url="">

<lev2 name="每周考勤浏览" url="AttendanceQuery.aspx"></lev2>

<lev2 name="每周考勤管理" url="AttendanceManager.aspx"></lev2>

</lev1>

<lev1 name="宿舍卫生" url="">

<lev2 name="每周卫生检查" url="HealthQuery.aspx"></lev2>

<lev2 name="卫生检查结果管理" url="HealthManager.aspx"></lev2>

</lev1>

<lev1 name="管理员信息" url="">

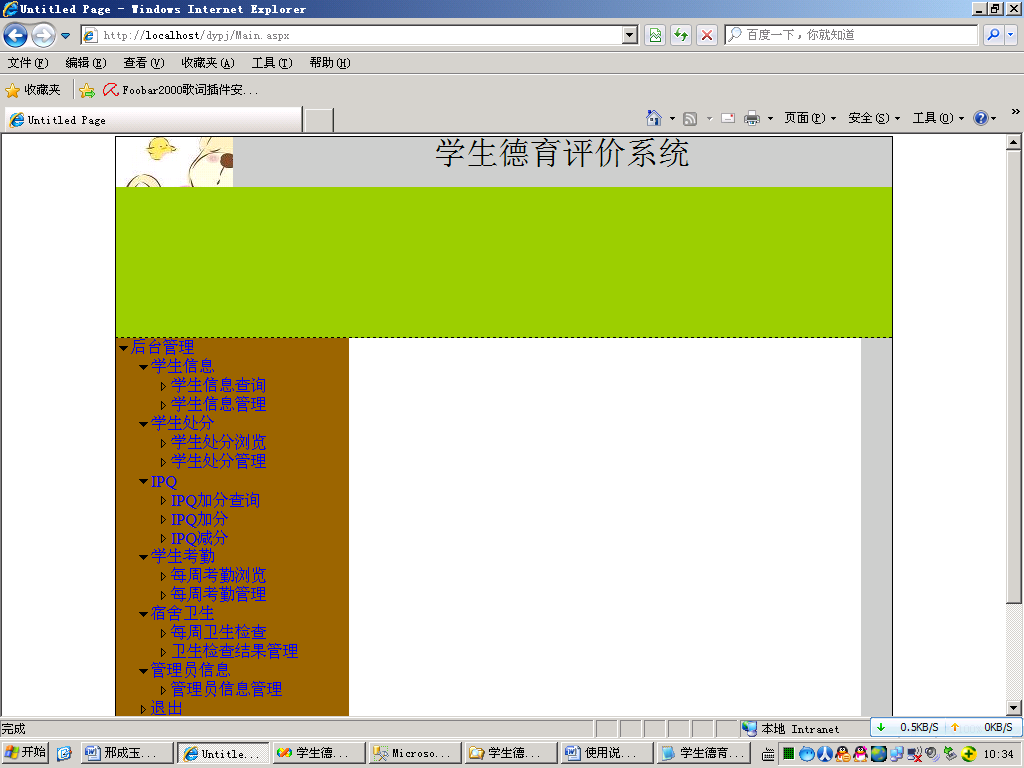
<lev2 name="管理员信息管理" url="AdminManger.aspx"></lev2>

</lev1>

<lev1 name="退出" url="javascript:window.close()"></lev1>

</lev0>

效果如图



## 5.5菜单树用户控件

Left.ascx

<%@ Control Language="VB" AutoEventWireup="false" CodeFile="Left.ascx.vb" Inherits="Left" %>

<asp:TreeView ID="Tree" runat="server" ImageSet="Arrows">

<DataBindings>

<asp:TreeNodeBinding DataMember="lev0" TextField="name" />

<asp:TreeNodeBinding DataMember="lev1" TextField="name" NavigateUrlField="url"/>

<asp:TreeNodeBinding DataMember="lev2" TextField="name" NavigateUrlField ="url" />

</DataBindings>

</asp:TreeView>

界面主要添加了一个名字为Tree的TreeView

Partial Class Left

Inherits System.Web.UI.UserControl

Protected Sub Tree\_Init(ByVal sender As Object, ByVal e As System.EventArgs) Handles Tree.Init

InitData()

End Sub

Private Sub InitData()

Dim privilegeFile As String = ""

If Session("student") Is Nothing = False Then

privilegeFile = "Student.xml"

Else

If Session("manager") Is Nothing = False Then

privilegeFile = "Manager.xml"

Else

If Session("admin") Is Nothing = False Then

privilegeFile = "Admin.xml"

End If

End If

End If

Dim xmlSrc As XmlDataSource = New XmlDataSource

xmlSrc.DataFile = privilegeFile

xmlSrc.DataBind()

Me.Tree.DataSource = xmlSrc

Tree.DataBind()

End Sub

End Class

初始化的时候，根据Session中的数据判断是哪一类用户,将对应的XMl绑定到树

## 5.6模板

MasterPage.master

<%@ Master Language="VB" CodeFile="MasterPage.master.vb" Inherits="MasterPage" %>

<%@ Register TagPrefix="HRManager" TagName="UserControls\_LeftTree" Src="~/Left.ascx" %>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml" >

<head runat="server">

<title>无标题页</title>

<link rel="stylesheet" type="text/css" href="StyleSheet.css" />

</head>

<body>

<form id="form1" runat="server">

<div id="wrap">

<div id="top">

<div id="header">

<div id="logo">

<a href="/" title="返回网站主页"> </a></div>

学生德育评价系统</div>

</div>

<div id="left">

<HRManager:UserControls\_LeftTree ID="leftTree" runat="server" />

</div>

<div id="right">

<asp:ContentPlaceHolder ID="ContentPlaceHolder1" runat="server">

</asp:ContentPlaceHolder>

</div>

<div id="bottom">

&copy; 2011 北方民族大学计算机系软件工程07级1班 版权所有 </div>

</div>

</form>

</body>

</html>

主要修改：注册一个用户控件并使用，使用div搭建框架，具体的样式链接样式表实现

## 5.7样式表

：StyleSheet.css

body

{

padding:0px;

margin-top:2px;

margin-right:0px;

margin-bottom:0px;

margin-left:0px;

}

#wrap

{

height:600px;

width:776px;

padding:0px;

margin-right:auto;

margin-left:auto;

border:1px solid #000000;

background-color:#CCCCCC;

}

#top

{

width:100%;

height:200px;

background-color:#99CC00;

border-bottom-width:1px;

border-bottom-style:dashed;

border-bottom-color:#000000;

}

#bottom

{

width:100%;

height:100px;

text-align:right;

font-size:small;

background-color:#99CC00;

border-bottom-width:1px;

border-bottom-style:dashed;

border-bottom-color:#000000;

}

#left

{

float:left;

width:30%;

height:400px;

background-color:#996600;

margin:0px;

padding:0px;

}

#right

{

width:66%;

height:400px;

background-color:#FFFFFF;

overflow:auto;

margin:0px;

padding:0px;

}

#header

{

height:50px;

width:776px;

margin-top:0px;

margin-right:auto;

margin-bottom:0px;

margin-left:auto;

padding:0px;

background-color:#CCCCCC;

text-align:center;

font-size:xx-large;

}

#logo

{

float:left;

width:117px;

height:100%;

background-image:url("IMG/keai.jpg");

background-repeat:no-repeat;

background-position:left center;

}

#logo a

{

width:117px;

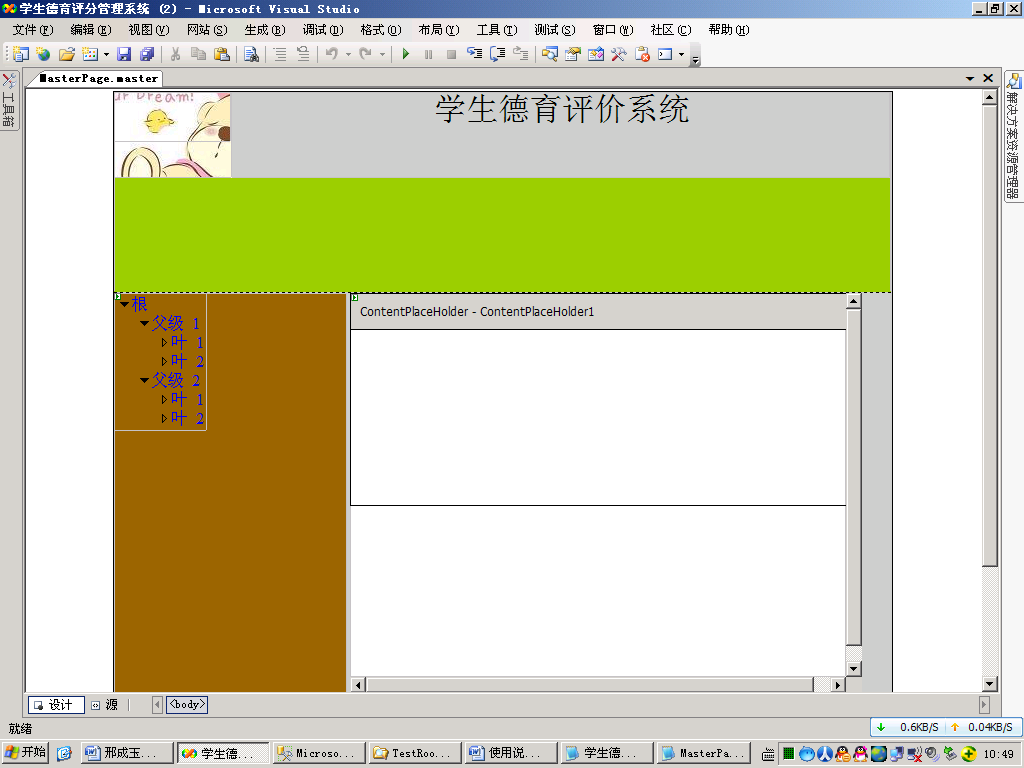
height:100%;

display:block;

}

解释：body页面背景。Top.bottom,left,right框架，header标题，logo，未添加美化

实现效果



## 5.8表示层

StudentManager.aspx 这些是页面显示层的，后缀主要有添加(Add)，修改(update)，查询(Query)，管理（manage）,前缀是一些实体的名字，具体看数据库设计里的英文和中文对照

StudentManager.aspx

<%@ Page Language="VB" MasterPageFile="~/MasterPage.master" AutoEventWireup="false" CodeFile="StudentManager.aspx.vb" Inherits="StudentManager" title="Untitled Page" %>

<asp:Content ID="Content1" ContentPlaceHolderID="ContentPlaceHolder1" Runat="Server">

&gt;&gt;学生信息管理<br />

学号：<asp:TextBox ID="TextBox1" runat="server"></asp:TextBox><asp:Button ID="btnQuery"

runat="server" Text="查询" />

<asp:Button ID="Button1" runat="server" Text="显示所有" /><br />

<asp:GridView ID="GV" runat="server" AllowPaging ="true" AutoGenerateColumns="false" PageSize="5" OnPageIndexChanging="GV\_PageIndexChanging">

<Columns>

<asp:TemplateField>

<ItemTemplate>

<asp:CheckBox ID="chkSelected" runat="server" Checked="false" Visible="true" />

</ItemTemplate></asp:TemplateField>

<asp:BoundField DataField="StudentId" HeaderText="学号" />

<asp:BoundField DataField="StudentName" HeaderText="姓名" />

<asp:BoundField DataField="StudentPassword" HeaderText="密码" />

<asp:BoundField DataField="StudentSex" HeaderText="性别" />

<asp:BoundField DataField="DormitoryType" HeaderText="宿舍类型" />

<asp:BoundField DataField="FloorId" HeaderText="宿舍楼号" />

<asp:BoundField DataField="DormitoryId" HeaderText="宿舍号" />

</Columns>

</asp:GridView>

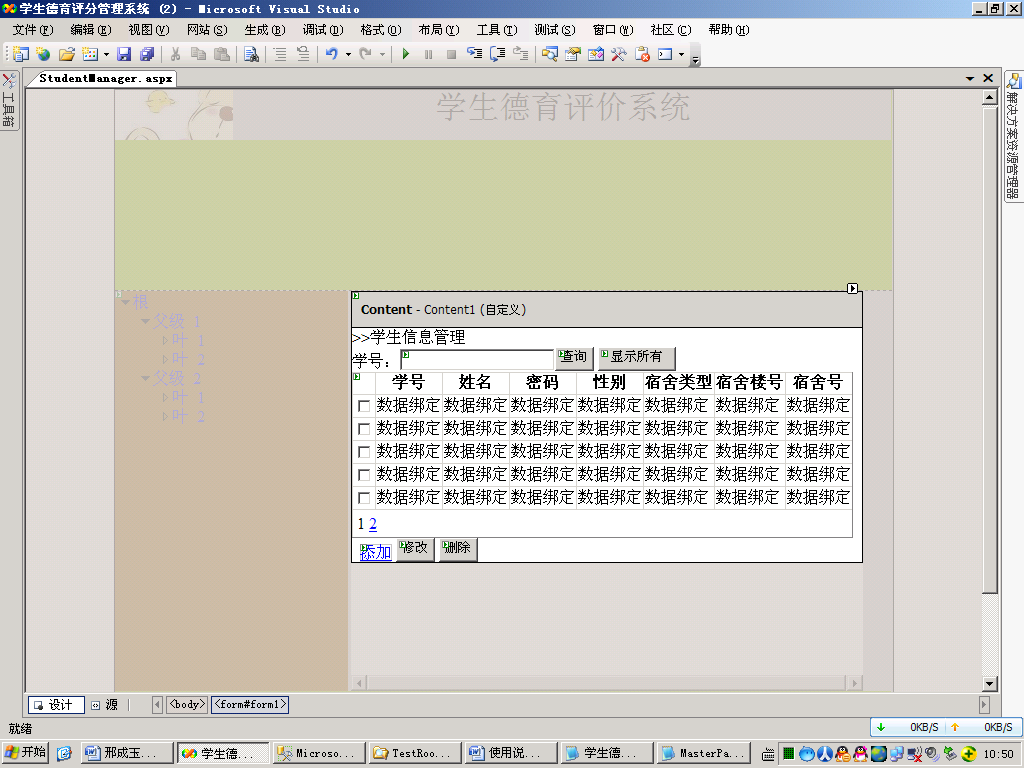
&nbsp;

<asp:HyperLink ID="HyperLink1" runat="server" NavigateUrl="~/StudentAdd.aspx">添加</asp:HyperLink>

<asp:Button ID="btnUpdate" runat="server" Text="修改" />

<asp:Button ID="btnDelete" runat="server" Text="删除" />

</asp:Content>



Page\_Load（判断是否首次加载，然后添加数据），QueryALL（查询所有的学生数据），btnUpdate\_Click，btnDelete\_Click（更新和删除），GetSelected（判断前面的复选框是否选择，并获取第一行的学号），Query（参数学号，查询指定的学生信息）, Button1\_Click(显示所有)，GV\_PageIndexChanging(数据集上单击数字，实现的换页)

学生更新页面StudentUpdate.aspx

<%@ Page Language="VB" MasterPageFile="~/MasterPage.master" AutoEventWireup="false" CodeFile="StudentUpdate.aspx.vb" Inherits="StudentUpdate" title="Untitled Page" %>

<asp:Content ID="Content1" ContentPlaceHolderID="ContentPlaceHolder1" Runat="Server">

&gt;&gt;修改用户<br />

学 &nbsp; &nbsp; &nbsp; 号：<asp:TextBox ID="TextBox1" runat="server" ReadOnly="True"></asp:TextBox><br />

姓 &nbsp; &nbsp; &nbsp; 名：<asp:TextBox ID="TextBox2" runat="server"></asp:TextBox><br />

密 &nbsp; &nbsp; &nbsp; 码：<asp:TextBox ID="TextBox3" runat="server"></asp:TextBox><br />

性 &nbsp; &nbsp; &nbsp; 别：<asp:TextBox ID="TextBox4" runat="server"></asp:TextBox><br />

宿舍类型：<asp:TextBox ID="TextBox5" runat="server"></asp:TextBox><br />

宿舍楼号：<asp:TextBox ID="TextBox6" runat="server"></asp:TextBox><br />

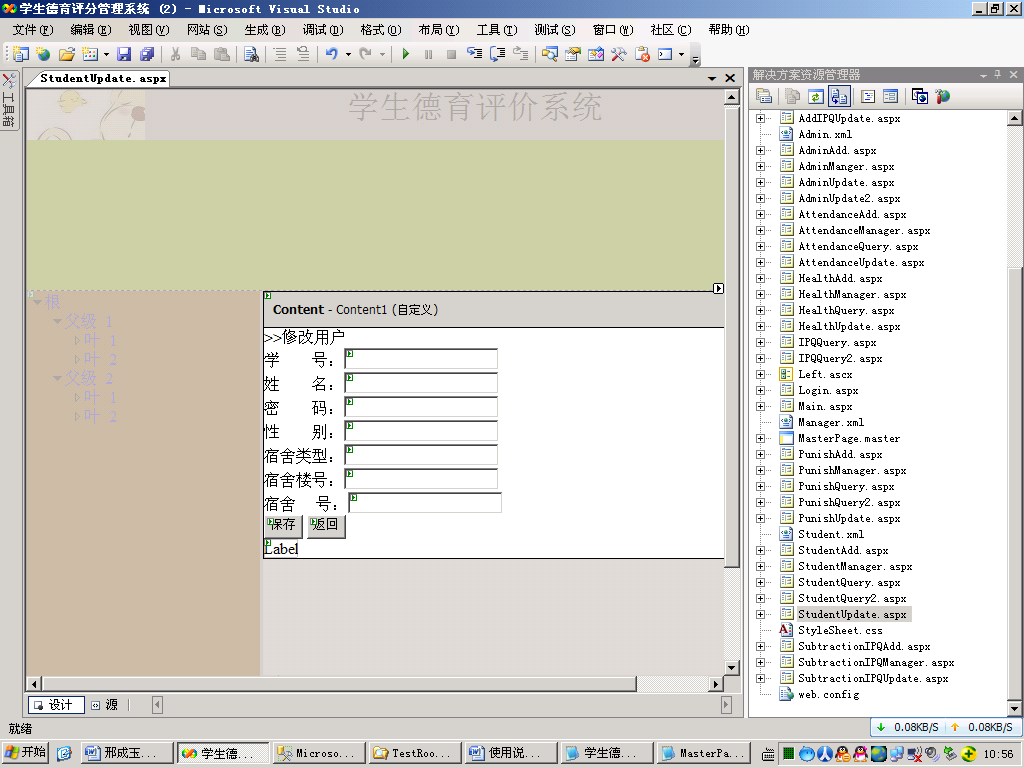
宿舍 &nbsp;&nbsp; 号：<asp:TextBox ID="TextBox7" runat="server"></asp:TextBox><br />

<asp:Button ID="btnSave" runat="server" Text="保存" />

<asp:Button ID="btnReturn" runat="server" Text="返回" /><br />

<asp:Label ID="LabelWarningMessage" runat="server" Text="Label"></asp:Label>

</asp:Content>



Partial Class StudentUpdate

Inherits System.Web.UI.Page

Protected Sub btnSave\_Click(ByVal sender As Object, ByVal e As System.EventArgs) Handles btnSave.Click

Dim ht As Hashtable = New Hashtable

ht.Add("StudentName", SqlStringConstructor.GetQuottedString(Me.TextBox2.Text.Trim()))

ht.Add("StudentPassword", SqlStringConstructor.GetQuottedString(Me.TextBox3.Text.Trim()))

ht.Add("StudentSex", SqlStringConstructor.GetQuottedString(Me.TextBox4.Text.Trim()))

ht.Add("DormitoryType", SqlStringConstructor.GetQuottedString(Me.TextBox5.Text.Trim()))

ht.Add("FloorId", SqlStringConstructor.GetQuottedString(Me.TextBox6.Text.Trim()))

ht.Add("DormitoryId", SqlStringConstructor.GetQuottedString(Me.TextBox7.Text.Trim()))

Dim where As String = "Where StudentId=" + SqlStringConstructor.GetQuottedString(Request.QueryString("login\_name").ToString())

StudentLogic.Update(ht, where)

Response.Redirect("StudentManager.aspx")

End Sub

Protected Sub btnReturn\_Click(ByVal sender As Object, ByVal e As System.EventArgs) Handles btnReturn.Click

Response.Redirect("StudentManager.aspx")

End Sub

Protected Sub Page\_Init(ByVal sender As Object, ByVal e As System.EventArgs) Handles Me.Init

InitData()

End Sub

Private Sub InitData()

Dim loginName As String = Request.QueryString("login\_name").ToString

Dim user As StudentLogic = New StudentLogic

user.LoadData(loginName)

If user.Exist Then

Me.TextBox1.Text = user.StudentId

Me.TextBox2.Text = user.StudentName

Me.TextBox3.Text = user.StudentPassword

Me.TextBox4.Text = user.StudentSex

Me.TextBox5.Text = user.DormitoryType

Me.TextBox6.Text = user.FloorId

Me.TextBox7.Text = user.DormitoryId

End If

End Sub

End Class

InitData（根据从管理页面传递过来的学号查询数据，获取数据，并给文本框添加数据），btnReturn\_Click，btnSave\_Click（返回和保存）

添加学生StudentAdd.aspx

<%@ Page Language="VB" MasterPageFile="~/MasterPage.master" AutoEventWireup="false" CodeFile="StudentAdd.aspx.vb" Inherits="StudentAdd" title="Untitled Page" %>

<asp:Content ID="Content1" ContentPlaceHolderID="ContentPlaceHolder1" Runat="Server">

&gt;&gt;添加用户<br />

学 &nbsp; &nbsp; &nbsp; 号：<asp:TextBox ID="TextBox1" runat="server"></asp:TextBox><br />

姓 &nbsp; &nbsp; &nbsp; 名：<asp:TextBox ID="TextBox2" runat="server"></asp:TextBox><br />

密 &nbsp; &nbsp; &nbsp; 码：<asp:TextBox ID="TextBox3" runat="server"></asp:TextBox><br />

性 &nbsp; &nbsp; &nbsp; 别：<asp:TextBox ID="TextBox4" runat="server"></asp:TextBox><br />

宿舍类型：<asp:TextBox ID="TextBox5" runat="server"></asp:TextBox><br />

宿舍楼号：<asp:TextBox ID="TextBox6" runat="server"></asp:TextBox><br />

宿舍 &nbsp;&nbsp; 号：<asp:TextBox ID="TextBox7" runat="server"></asp:TextBox><br />

<asp:Button ID="btnSave" runat="server" Text="保存" />

<asp:Button ID="btnReturn" runat="server" Text="返回" /><br />

<asp:Label ID="LabelWarningMessage" runat="server" Text="Label"></asp:Label>

</asp:Content>

其实添加和修改没多少区别的，修改页面加载时需要根据数据库查询数据，然后将修改后的数据写入数据库，添加时只要考虑到不重复添加数据就行

Partial Class StudentAdd

Inherits System.Web.UI.Page

Protected Sub btnSave\_Click(ByVal sender As Object, ByVal e As System.EventArgs) Handles btnSave.Click

If StudentLogic.HasUser(Me.TextBox1.Text.Trim()) Then

Me.LabelWarningMessage.Text = "<font color=red>警告:登录名[" + Me.TextBox1.Text.Trim + "]已存在!</font>"

Me.TextBox1.Text = ""

Else

Me.LabelWarningMessage.Text = ""

Dim ht As Hashtable = New Hashtable

ht.Add("StudentId", SqlStringConstructor.GetQuottedString(Me.TextBox1.Text.Trim()))

ht.Add("StudentName", SqlStringConstructor.GetQuottedString(Me.TextBox2.Text.Trim()))

ht.Add("StudentPassword", SqlStringConstructor.GetQuottedString(Me.TextBox3.Text.Trim()))

ht.Add("StudentSex", SqlStringConstructor.GetQuottedString(Me.TextBox4.Text.Trim()))

ht.Add("DormitoryType", SqlStringConstructor.GetQuottedString(Me.TextBox5.Text.Trim()))

ht.Add("FloorId", SqlStringConstructor.GetQuottedString(Me.TextBox6.Text.Trim()))

ht.Add("DormitoryId", SqlStringConstructor.GetQuottedString(Me.TextBox7.Text.Trim()))

StudentLogic.Add(ht)

Response.Redirect("StudentManager.aspx")

End If

End Sub

Protected Sub btnReturn\_Click(ByVal sender As Object, ByVal e As System.EventArgs) Handles btnReturn.Click

Response.Redirect("StudentManager.aspx")

End Sub

End Class

查询界面基本上和管理界面的查询没有区别，除了没有下面的添加，修改删除按钮。这里就不介绍了。

其他的还有

## 5.9登录页面

Login.aspx

<%@ Page Language="VB" AutoEventWireup="false" CodeFile="Login.aspx.vb" Inherits="\_Default" %>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml" >

<head runat="server">

<title>无标题页</title>

<link rel="stylesheet" type="text/css" href="StyleSheet.css" />

</head>

<body style="background-color: #9999ff; text-align: center;">

<form id="form1" runat="server">

<div id="wrap">

<div id="top">

<div id="header">

<div id="logo">

<a href="/" title="返回网站主页"> </a></div>

学生德育评价系统</div>

</div>

<div id="left">

</div>

<div id="right">

用户名：

<asp:TextBox ID="txtUserName" runat="server"></asp:TextBox>

<asp:RequiredFieldValidator ID="RequiredFieldValidator1" runat="server" ControlToValidate="txtUserName"

ErrorMessage="不能为空！"></asp:RequiredFieldValidator><br />

密 码：

<asp:TextBox ID="txtPwd" runat="server" TextMode="Password"></asp:TextBox>

<asp:RequiredFieldValidator ID="RequiredFieldValidator2" runat="server" ControlToValidate="txtPwd"

ErrorMessage="不能为空！"></asp:RequiredFieldValidator> <br />

<asp:RadioButtonList ID="rblClass" runat="server" RepeatDirection="Horizontal">

<asp:ListItem Selected="True" Value="1">学生</asp:ListItem>

<asp:ListItem Value="2">管理员</asp:ListItem>

<asp:ListItem Value="3">辅导员</asp:ListItem>

</asp:RadioButtonList><br />

<asp:Label ID="lblMessage" runat="server"></asp:Label><br />

<asp:Button ID="btnLogin" runat="server" Text="登录" />

<asp:Button ID="btnClose" runat="server" Text="重置" />

</div>

<div id="bottom">

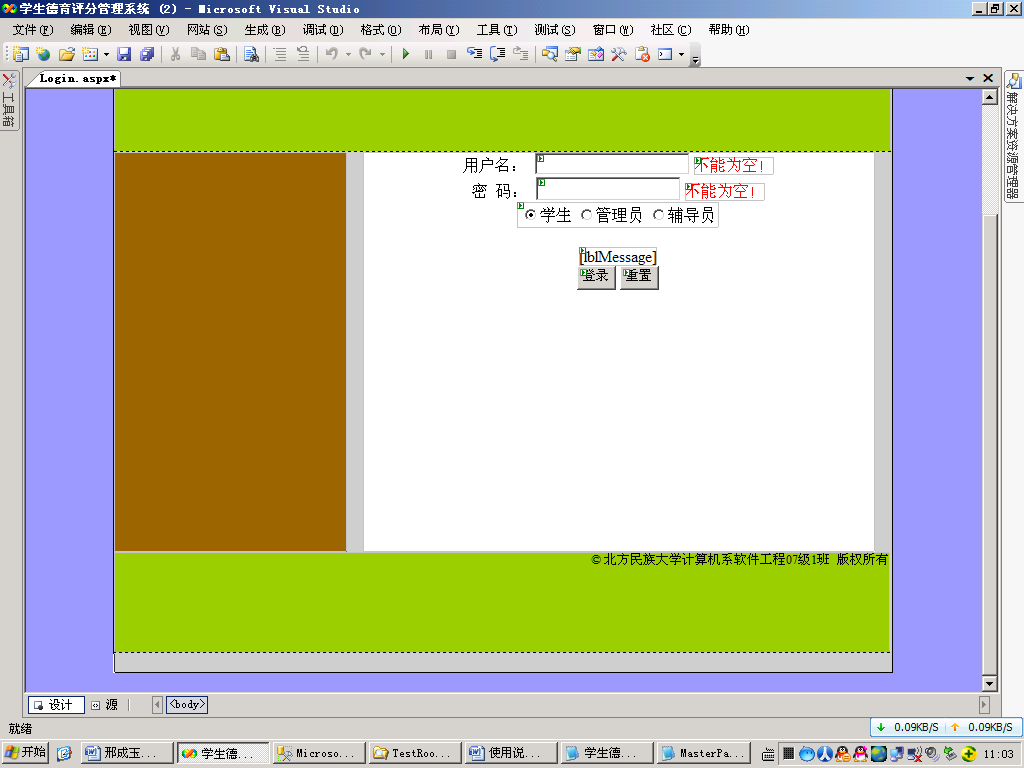
&copy; 北方民族大学计算机系软件工程07级1班 版权所有 </div>

</div>

</form>

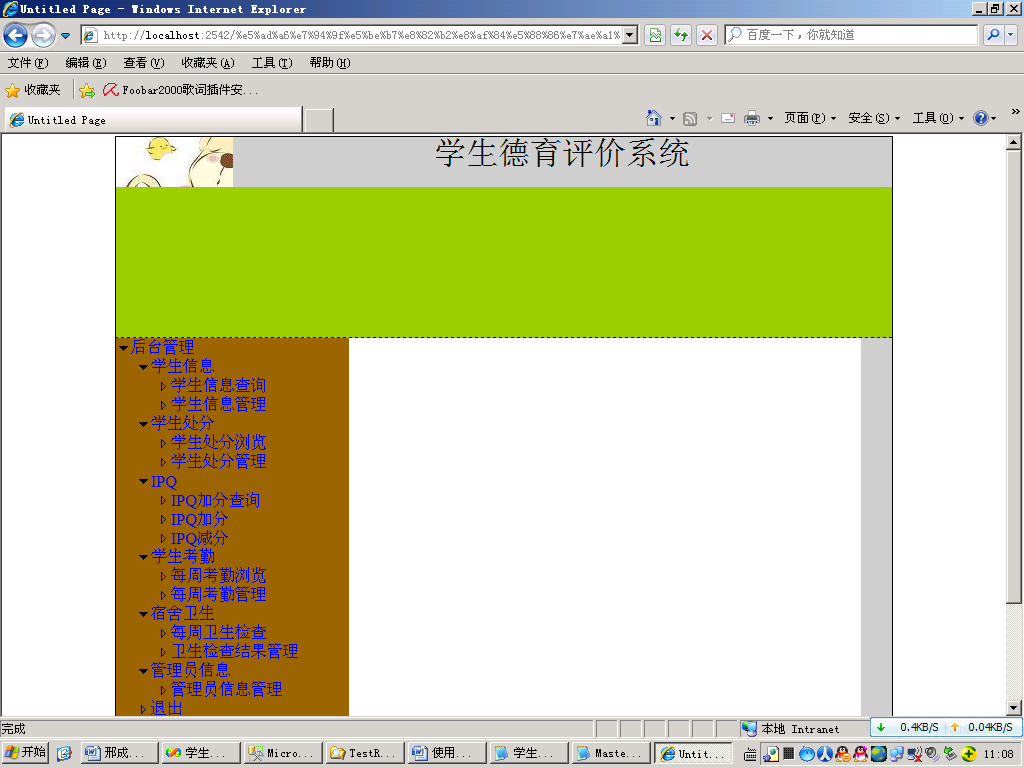
</body>

</html>

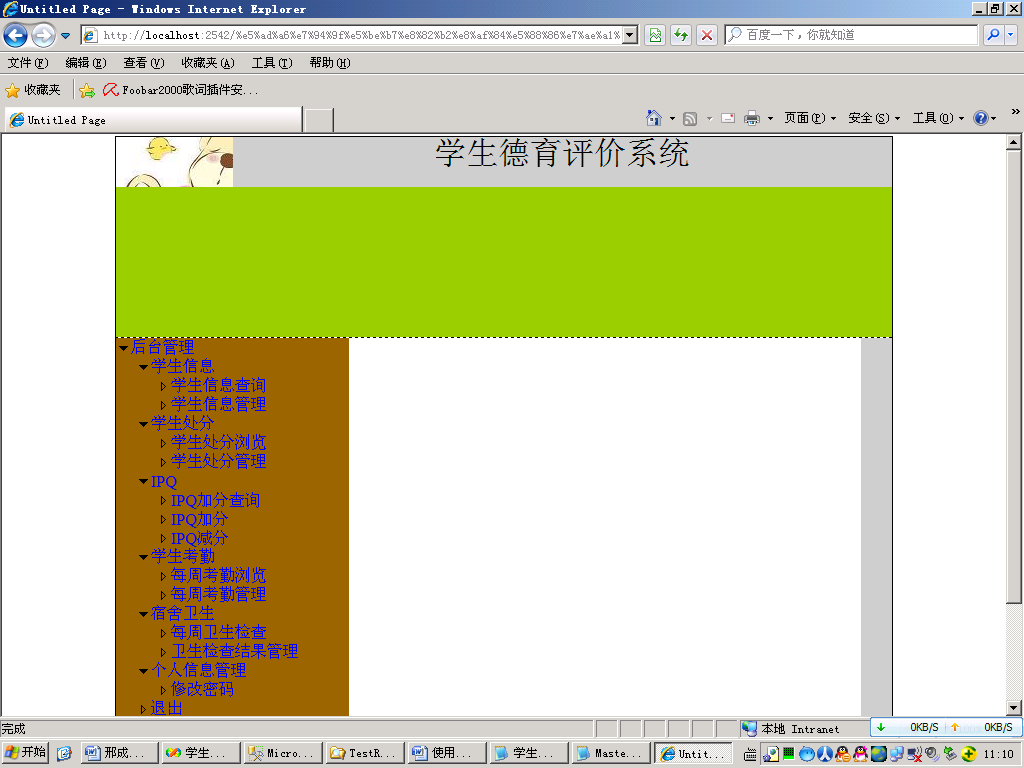


主要有两个方法：btnClose\_Click（清除文本框）, btnLogin\_Click(通过判断用户是否存在，以及用户的权限加载相应的界面，三种界面如下)

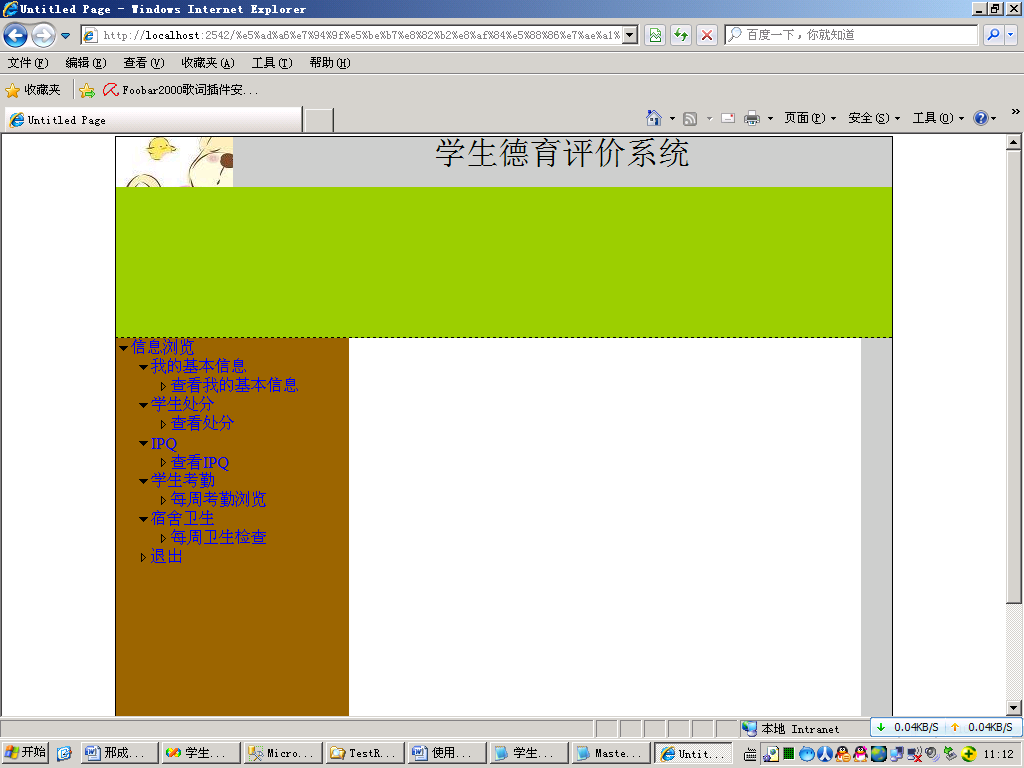
辅导员加载后的页面



管理员加载后的页面（注意，管理员信息下面只有一个修改密码的选项）



学生加载后的界面（只有查询权限，没有管理权限）



系统首页Main.aspx

一个跳转页面，本来这个页面应该根据权限跳转到相应的界面的

# 6 系统的测试

## 6.1 系统测试目的和原则

系统测试是将已经确认的软件、计算机硬件、外设、网络等其他元素结合在一起，进行信息系统的各种组装测试和确认测试，其目的是通过与系统的需求相比较，发现所开发的系统与用户需求不符或矛盾的地方，从而提出更加完善的方案。

## 6.2 系统测试

表6-1系统测试

|  |  |  |
| --- | --- | --- |
| 编号 | 测试内容 | 测试结果 |
| 1 | 管理员登录 | 成功 |
| 2 | 增加学生 | 成功 |
| 3 | 修改学生 | 成功 |
| 4 | 删除学生 | 成功 |
| 5 | 添加IPQ | 成功 |
| 6 | 修改IPQ | 成功 |
| 7 | 删除IPQ | 成功 |
| 8 | 修改密码 | 成功 |
| 9 | 增加每周卫生检查结果 | 成功 |
| 10 | 增加每周考勤 | 成功 |

# 结束语

第一次给自己做毕业设计，中间就遇到了不少麻烦，不过解决麻烦就是最大的乐趣。

这次的毕业设计即将结束，想想这段经历，总是颇多感慨。

关于技术方面：初次接触到vb.net应该是大二的时候，那时只是学会了使用编程语言。数据库是大三之后才接触到。大三开始尝试做简单的管理系统，大四我才开始学做asp.net网页。正是因为有了这些基础，毕业设计没有太大的难度。div+css框架一直没学好，这也许是最大的遗憾。

遇到的问题：用习惯了access，初次使用sqlServer，还真有点不适应，经过查资料，最后终于解决了。

万事都是开头难。经过这次的毕业设计，使我对asp.net又有了进一步的了解。

# 致 谢

路漫漫其修远兮，吾将上下而求索。大学短短的四年即将结束，这期间的点点滴滴依旧留在心头。在这里首先要感谢大学的老师，是老师教给我们知识，可惜自己一直都没学好。还要感谢学长教我学会vb.net。最后感谢同学们，是他们陪我度过这个难忘的大学阶段**。**

# 参考文献

[1] 刘乃丽编 《ASP.net 2.0+SQL Server 2005项目开发》 人民邮电出版社 2007年5月第一版 P61-117 P435-518

[2] 杨阳等 编著 《DIV+CSS网站布局实录》 北京科学出版社 2009 p106-145