**学生选课管理系统的设计与开发**

目录

[1、项目背景 1](#_Toc92394408)

[2、技术选型 1](#_Toc92394409)

[3、需求分析 2](#_Toc92394410)

[3.2管理员角色拥有的功能 2](#_Toc92394411)

[4、数据库设计 3](#_Toc92394412)

[4.1数据库关系图 3](#_Toc92394413)

[4.2完成系统中数据表的结构设计 3](#_Toc92394414)

[5、界面设计 4](#_Toc92394415)

[5.1管理员角色 5](#_Toc92394416)

[5.2学生界面 10](#_Toc92394417)

[6、关键技术的介绍 13](#_Toc92394418)

[7、系统测试 23](#_Toc92394419)

[8、完成情况总结 24](#_Toc92394420)

## 1、项目背景

学生选课管理系统是一个学校不可缺少的部分，传统的人工管理档案的方式存在着很多的缺点，如：效率低、保密性差等，所以开发一套综合教务系统管理软件很有必要，它应该具有传统的手工管理所无法比拟的优点，如：检索迅速、查找方便、可靠性高、存储量大、保密性好、寿命长、成本低等。这些优点能够极大的提高学校的管理效率，有助于推进学校的信息化建设。

## 2、技术选型

C#的窗体程序开发

本系统未采用C#实现MDI——多文档窗口，因为考虑到C#的该技术与java类似，而暑期java实训时，曾用过类似的方法做过停车场管理系统，所以想为这次的系统注入一点新鲜的血液，所以本系统设计的主题采用了了C#的C#TreeView 实现。

Sql server数据库

## 3、需求分析

本系统只有一类用户：管理员。管理员可以通过系统来添加管理员信息、修改管理员信息、添加学生信息、修改学生信息；开设课程、查询课程等。

### 3.2管理员角色拥有的功能

管理员信息管理

添加管理员信息；

管理员可以添加其他的管理员信息。

修改管理员信息；

管理员可以删除其他的管理员信息。

课程管理

开设课程；

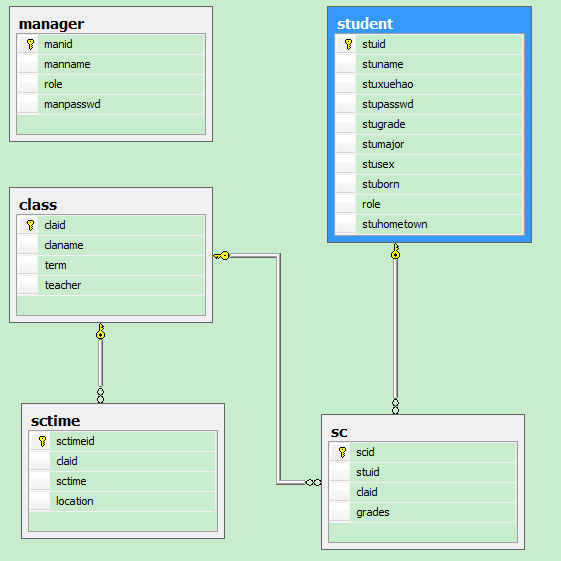
管理员可以录入课程信息，学生就可以浏览到这些课程信息，从而选课。

查询课程；

可以通过学期查询课程，或通过学期和课程结合的方式查询课程。

## 4、数据库设计

### 4.1数据库关系图



### 4.2完成系统中数据表的结构设计

管理员信息表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 用户表：UserInfo | | | | |
| 字段名称 | 数据类型 | 可空 | 默认值 | 说明 |
| id | nvarchar(20) | 不可 |  | 主键，管理员id |
| name | nvarchar(20) | 不可 |  | 管理员姓名 |
| pwd | nvarchar(50) | 不可 |  | 管理员密码 |
| sex | nvarchar(5) | 不可 |  | 管理员性别 |

学生选课信息表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 管理员信息表：CourseInfo | | | | |
| 字段名称 | 数据类型 | 可空 | 默认值 | 说明 |
| id | int | 不可 | 自动增长 | 主键，课程id |
| name | nvarchar(100) | 不可 |  | 课程名称 |
| kcms | nvarchar(500) | 不可 |  | 课程描述 |
| xkrq | DATETIME | 不可 |  | 学生选课日期 |
| xkxs | nvarchar(20) | 不可 |  | 选课学生 |

## 5、界面设计

登录界面：



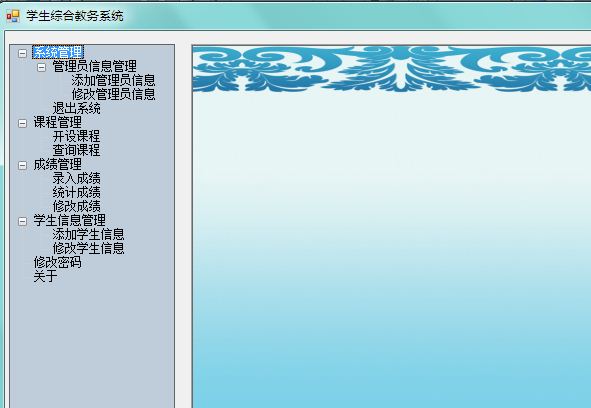
### 5.1管理员角色

登陆界面：

管理员界面：



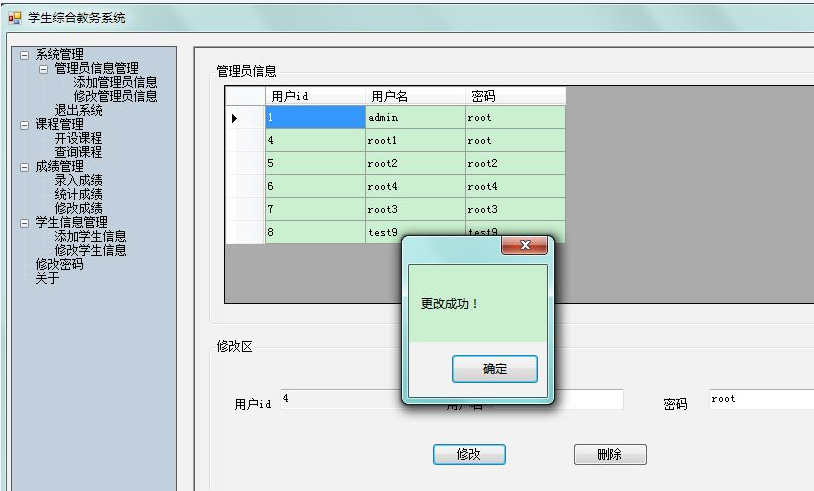
管理员主界面：



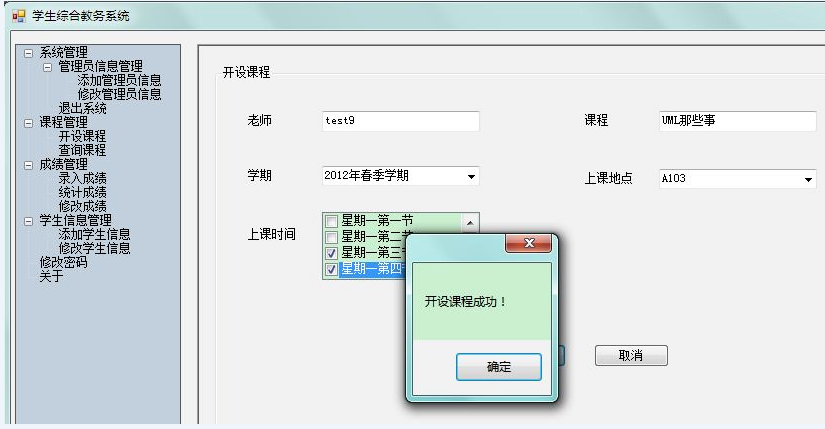
管理员-------添加管理员界面：



管理员——修改管理员界面：



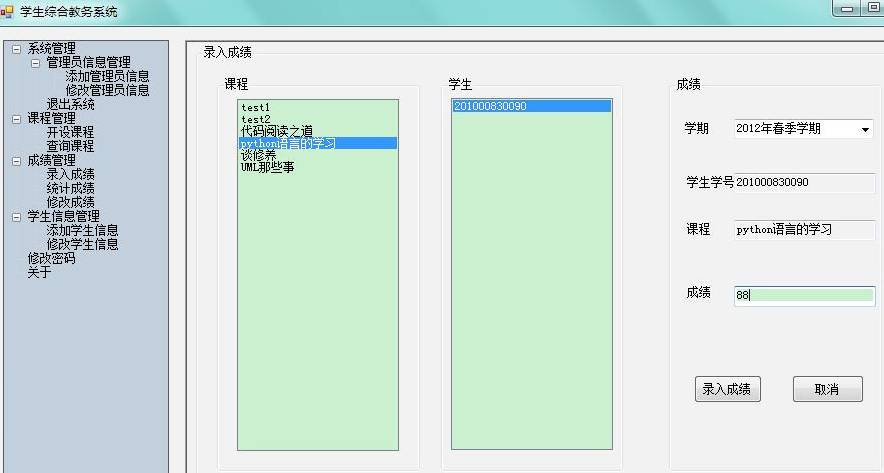
管理员——开设课程界面：



管理员——查询课程界面：



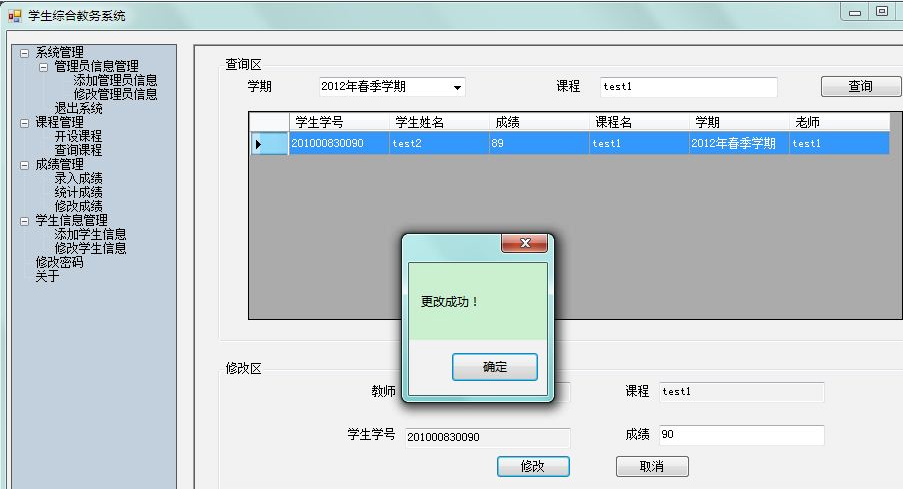
管理员——录入成绩界面：



管理员——统计成绩页面：



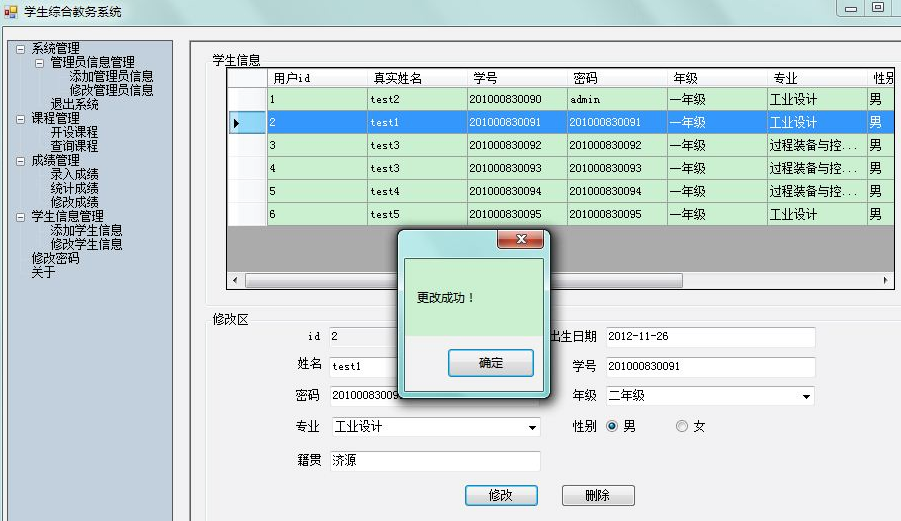
管理员——修改成绩界面：



管理员——添加学生信息界面：



管理员——修改学生信息界面：

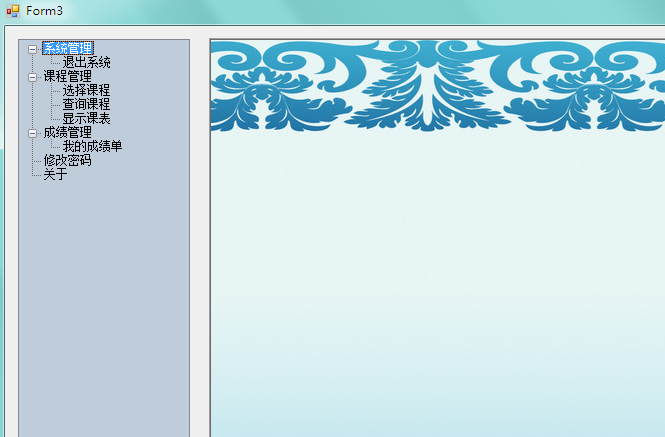


管理员——修改密码界面：



### 5.2学生界面

学生登录主界面：



学生——选择课程界面：



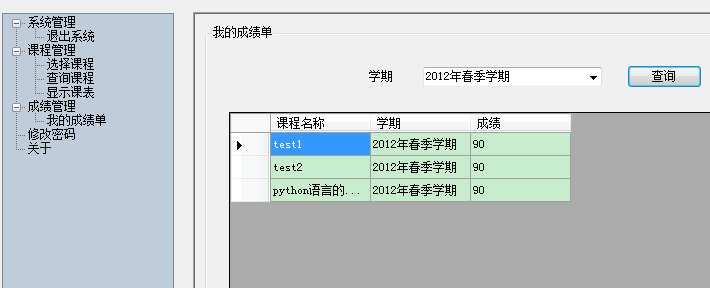
学生——查询课程界面：



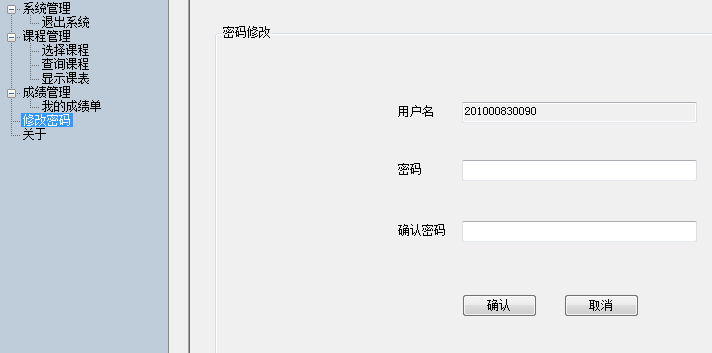
学生——显示课表界面：



学生——我的成绩单：



学生——密码修改界面：



## 6、关键技术的介绍

6.1使用treeview实现点击不同的选项，在右侧面板中显示不同的界面。

具体代码如下：

private void treeView1\_AfterSelect(object sender, TreeViewEventArgs e)

{

switch (this.treeView1.SelectedNode.Text)

{

case"录入成绩":

Form2 f = new Form2();

f.TopLevel = false;

f.FormBorderStyle = FormBorderStyle.None;

f.WindowState = FormWindowState.Maximized;

panel2.Controls.Add(f);

f.Show();

break;

case"我的成绩单":

searchgradeForm f1 = new searchgradeForm();

f1.TopLevel = false;

f1.FormBorderStyle = FormBorderStyle.None;

f1.WindowState = FormWindowState.Maximized;

panel2.Controls.Add(f1);

f1.Show();

break;

case"统计成绩":

countForm f2 = new countForm();

f2.TopLevel = false;

f2.FormBorderStyle = FormBorderStyle.None;

f2.WindowState = FormWindowState.Maximized;

panel2.Controls.Add(f2);

f2.Show();

break;

case"开设课程":

kaisheForm f3 = new kaisheForm();

f3.TopLevel = false;

f3.FormBorderStyle = FormBorderStyle.None;

f3.WindowState = FormWindowState.Maximized;

panel2.Controls.Add(f3);

f3.Show();

break;

case"选择课程":

chooseForm f4 = new chooseForm();

f4.TopLevel = false;

f4.FormBorderStyle = FormBorderStyle.None;

f4.WindowState = FormWindowState.Maximized;

panel2.Controls.Add(f4);

f4.Show();

break;

case"查询课程":

searchclassForm f5 = new searchclassForm();

f5.TopLevel = false;

f5.FormBorderStyle = FormBorderStyle.None;

f5.WindowState = FormWindowState.Maximized;

panel2.Controls.Add(f5);

f5.Show();

break;

case"添加管理员信息":

addmanForm f6 = new addmanForm();

f6.TopLevel = false;

f6.FormBorderStyle = FormBorderStyle.None;

f6.WindowState = FormWindowState.Maximized;

panel2.Controls.Add(f6);

f6.Show();

break;

case"修改管理员信息":

modifymanForm f7 = new modifymanForm();

f7.TopLevel = false;

f7.FormBorderStyle = FormBorderStyle.None;

f7.WindowState = FormWindowState.Maximized;

panel2.Controls.Add(f7);

f7.Show();

break;

case"添加教师信息":

addteacForm f8 = new addteacForm();

f8.TopLevel = false;

f8.FormBorderStyle = FormBorderStyle.None;

f8.WindowState = FormWindowState.Maximized;

panel2.Controls.Add(f8);

f8.Show();

break;

case"修改教师信息":

modifyteacForm f9 = new modifyteacForm();

f9.TopLevel = false;

f9.FormBorderStyle = FormBorderStyle.None;

f9.WindowState = FormWindowState.Maximized;

panel2.Controls.Add(f9);

f9.Show();

break;

case"添加学生信息":

addstuForm f10 = new addstuForm();

f10.TopLevel = false;

f10.FormBorderStyle = FormBorderStyle.None;

f10.WindowState = FormWindowState.Maximized;

panel2.Controls.Add(f10);

f10.Show();

break;

case"修改学生信息":

modifystuForm f11 = new modifystuForm();

f11.TopLevel = false;

f11.FormBorderStyle = FormBorderStyle.None;

f11.WindowState = FormWindowState.Maximized;

panel2.Controls.Add(f11);

f11.Show();

break;

case"关于":

aboutForm f12 = new aboutForm();

f12.TopLevel = false;

f12.FormBorderStyle = FormBorderStyle.None;

f12.WindowState = FormWindowState.Maximized;

panel2.Controls.Add(f12);

f12.Show();

break;

case "退出系统":

Application.Exit();

break;

case "显示课表":

showkebiaoForm f13 = new showkebiaoForm();

f13.TopLevel = false;

f13.FormBorderStyle = FormBorderStyle.None;

f13.WindowState = FormWindowState.Maximized;

panel2.Controls.Add(f13);

f13.Show();

break;

case "修改密码":

modifymimaForm f14 = new modifymimaForm();

f14.TopLevel = false;

f14.FormBorderStyle = FormBorderStyle.None;

f14.WindowState = FormWindowState.Maximized;

panel2.Controls.Add(f14);

f14.Show();

break;

case "修改成绩":

modifygradeFram f15 = new modifygradeFram();

f15.TopLevel = false;

f15.FormBorderStyle = FormBorderStyle.None;

f15.WindowState = FormWindowState.Maximized;

panel2.Controls.Add(f15);

f15.Show();

break;

}

}

6.2在其他的窗口得到在登录窗口输入的信息

实现方法：在loginFram中设置两个方法，在其他类中只需要引用loginFram的这些方法即可。具体实现如下：

public static String getStudent()

{

String stuxuehao = "";

stuxuehao = loginForm.name;

return stuxuehao;

}

public static String getRole()

{

String role1 = "";

role1 = role;

return role1;

}

6.3管理员开设课程时，对于同一时间该教室有课的情况怎么做出异常处理。集体实现如下：

private void button1\_Click(object sender, EventArgs e)

{

string teacher = textBoxteacher.Text;

string classes = textBoxclass.Text;

string term = comboBoxterm.SelectedItem.ToString();

string flags = "1";

SqlConnection conn = new SqlConnection(loginForm.connectionString);

conn.Open();

SqlCommand cmd = new SqlCommand();

cmd.Connection = conn;

//将开课信息插入到开课表里

//MessageBox.Show("开设课程成功！");

string sql = "";

//得到上课的地点

string didian = comboBoxdidian.SelectedItem.ToString();

//checkedListBoxtime

for (int i = 0; i < checkedListBoxtime.Items.Count; i++)

{

if (checkedListBoxtime.GetItemChecked(i))

{

String time = checkedListBoxtime.GetItemText(checkedListBoxtime.Items[i]);

sql = "select \* from sctime where sctime = '" + time + "'and location = '" + didian + "'";

SqlDataAdapter adp = new SqlDataAdapter(sql, conn);

DataSet ds = new DataSet();

adp.Fill(ds);

if (ds.Tables[0].Rows.Count > 0)

{

flags = "2";

MessageBox.Show("该时间该教室已经有课！");

break;

}

else

{

flags = "1";

break;

}

}

}

if (flags == "1")

{

sql = "insert into class (claname,term,teacher) values ('" + classes + "','" + term + "','" + teacher + "')";

cmd.CommandText = sql;

cmd.ExecuteNonQuery();

for (int i = 0; i < checkedListBoxtime.Items.Count; i++)

{

if (checkedListBoxtime.GetItemChecked(i))

{

string time = checkedListBoxtime.GetItemText(checkedListBoxtime.Items[i]);

//将开课表的id得到

sql = "select claid from class where claname = '" + classes + "' and term = '" + term + "' and teacher = '" + teacher + "'";

cmd.CommandText = sql;

String id1 = cmd.ExecuteScalar().ToString();

int id = 0;

int.TryParse(id1, out id);

sql = "insert into sctime values(" + id + ",'" + time + "','" + didian + "')";

cmd.CommandText = sql;

cmd.ExecuteNonQuery();

}

}

MessageBox.Show("开设课程成功！");

}

conn.Close();

}

6.4学生选择课程时，对选择一门课程与已选择的课程出现上课时间冲突的情况的异常处理。集体实现如下：

private void button1\_Click(object sender, EventArgs e)

{

string flags = "1";

//得到stuid

string stuxuehao = textBox1.Text;

SqlConnection conn = new SqlConnection(loginForm.connectionString);

conn.Open();

string sql = "select stuid from student where stuxuehao = '" + stuxuehao + "'";

SqlCommand cmd = new SqlCommand(sql, conn);

String id1 = cmd.ExecuteScalar().ToString();

int.TryParse(id1, out stuid);

//得到课程的id

int claid = 0;

int.TryParse(textBoxid.Text, out claid);

//查询你在该时间是否有课

sql = "select sctime from sctime where claid =" + claid;

SqlDataAdapter adp = new SqlDataAdapter(sql, conn);

DataSet ds = new DataSet();

adp.Fill(ds);

for (int i = 0; i < ds.Tables[0].Rows.Count; i++)

{

DataRow dr = ds.Tables[0].Rows[i];

string time = dr[0].ToString();//第一列

sql = "select \* from sc,sctime,class where class.claid = sc.claid and class.claid = sctime.claid and sctime = '" + time + "' and sc.stuid =" + stuid;

SqlDataAdapter adp1 = new SqlDataAdapter(sql, conn);

DataSet ds1 = new DataSet();

adp1.Fill(ds1);

if (ds1.Tables[0].Rows.Count > 0)

{

flags = "2";

MessageBox.Show("课程上课时间冲突！");

break;

}

}

if (flags == "1")

{

sql = "insert into sc(claid,stuid) values(" + claid + "," + stuid + ")";

cmd.CommandText = sql;

if (cmd.ExecuteNonQuery() > 0)

{

MessageBox.Show("选课成功！");

}

}

if (listBox1.Items.Count > 0)

{//清空所有项

listBox1.Items.Clear();

}

sql = "select class.claname from sc,class where sc.claid = class.claid and stuid=" + stuid;

SqlDataAdapter adp2 = new SqlDataAdapter(sql, conn);

DataSet ds2 = new DataSet();

adp2.Fill(ds2);

foreach (DataRow row in ds2.Tables[0].Rows)

{

listBox1.Items.Add(row[0].ToString());

}

conn.Close();

}

## 7、系统测试

7.1怎么提高访问数据库的速度

改进方法：引入视图、触发器、函数、存储过程，优化表结构，建立索引。

## 8、完成情况总结

通过几周的努力，从前期的需求分析与架构设计到实际的编码阶段，我完成了学生选课及成绩管理系统的开发与设计。已经大致完成了主要的功能，不过还有一些亟待完善的方面，例如对数据的存储采用加密的措施，UI设计并不是很出色等等，这些问题都需要我再以后的时间里去慢慢弥补，还有刚开始，我对treeview并不熟悉，所以先建立了几个原型来测试，领会了treeview的使用方法后才切入正题进行开发。

在这开发的过程中，我收获很大，相比前几次的开发，我真正重视起测试在软件开发中的作用，在软件开发的后期，我专门建立一个bug列表，将出现的bug详细记录在列表中，并在软件调试的后期对照列表认真改进软件，考虑以后使用软件来做相关方面的管理。当然，我也遇到不少代码编写的问题，有时要花较长的时间来解决问题，这也是自己平时编写代码较少的原因。通过这次软件的开发，在编写代码上有了一定的提高，同时也知道了软件开发过程中需要注意的问题。比如，开发逻辑思维要清晰，不然可能会把问题想的复杂化。平时多和项目成员交流，这样你才能知道自己的想法有那些出入，利于软件的开发。

通过这次实验，使我对软件开发有了更深一步的了解。在软件开发中应注意的问题和考虑的问题都能较好的做到。