ТУ-София

Курсов проект

по

Програмиране в разпределена среда

Изработил: Татяна Галинова Ганчева

ФКСУ, КСИ, 42гр, фак. № 121214019

Съдържание:

Увод-въведение в проблема, който се решава

Целта на това уеб приложение е да съхранява в база информация за група от хара, в този случай- за студенти. Приложението изважда информация от база данни, систематизира я и я подрежда, така че да е удобна за ползване от крайния потребитрел.

Всяко едно учреждение като например униерситет, колеж или училище трябва да съхранява данни за своите възпитаници и да може да има бърз и лесен достъп до тази база данни.

С уеб приложението тези учреждения могат да намират информация за своите студенти или ученици лесно и да търсят в морето от записани лица.

Анализ на съществуващи разработки

Като едно новосъздадено уеб приложение с уеб сървис, и това също търпи още много промение и подобрения с цел да стане по-функционално и да предлага повече възможности.

Като за начало бих започнала от най-лесното – добавяне на още полета за информация за даден студент. Според желанията на крайния потребител, може да се добави още информация с цел по-пълно представяне на дадения студент в базата данни.

Също така може информацията да се раздели на няколко подраздела или страници с линкове към тях с цел по-добра структура на информацията и уеб приложението.

В момента не се изисква определена валидация от крайния потребител, за да добавя или маха студент от базата данни. Но добра идея за развитие на уеб сървиса и страницата е да се сложи такава. И да има само определена група от хора, които да могат да променят базата данни. Всички останали да влизат като гости и да могат само да разглеждат информацията вътре.

Проектиране

Основната дейност на тази уеб страница е показване и актуализиране на информация. Информацията се съхранява в база данни, която за момента се намира локално на този компютър. Базата данни има следната структура:

Column Name	Data Type	Allow Nulls
ID	int	☐
Name	varchar(50)	☑
Age	int	☑
Course	int	☑
Class	varchar(50)	☑
AverageMark	float	☑
		☐

Тя е създадена с Microsoft SQL Service Management Studio.

Уеб сървиса използва протокола за обмен на структурирана информация SOAP. При стартиране на приложението се отваря уеб страницата в браузъра и се показва поле, в което можем да търсим даден студент или да изберем студент по име от списък отдолу и да видим повече информация за него.

Ако решим да търсим студент можем да изберем дали да го търсим по име или по факултет(Class).

Най-отдолу има три бутона- бутон search, който при натискането му, взима информацията за критерий на търсене от drop down менюто и самата ключова дума от текстовото поле. И с тях търси в базата данни. Ако сме задали име за критерий на търсене, то ще се претърсва само колоната Name от таблицата в базата данни.

До бутона за търсене има и бутон за изтриване- Delete. С него можем да изтрием информация за даден студент от базата данни. Ако искаме да изтрием студент, трябва да го изберем от таблицата с имена на страницата, по този начин го маркираме и след това да натиснем бутона за изтриване.

И накрая има и бутон View Details. При избиране на име на студент от списъка с имена на началната страница, то се маркира. Когато искаме да видим повече детайли за този студент, като например години, кой курс е, каква специалност ит.н., натискаме бутона View Details. Той ни препраща към друга страница. Там виждаме информацията за този студент.

Можем да актуализираме информацията или да добавим нов студент , въвеждайки информацията в полетата на страницата.

Реализация

Курсовият проект се състои от два проекта реализирани във Visual Studio.

Първият е StudentInfoBase, а вторият е Client Project.

В първият проект се намират основните логики на моята уеб страница-методите за добавяне, изтриване, актуализиране и показване на информация. Тези методи са в уеб сървиса StudentInfoSOAP.asmx.cs.

Метод за добавяне на студент:

```
        [WebMethod]
        public int AddNewStudent(string name, int age, int course, string Class, double
AvrgMark)
        {
            using(conn=new SqlConnection(connections))
            {
                conn.Open();
                var comand = new SqlCommand("Insert into StudentInfos(Name, Age, Course,
class, avrgMark) values ('" + name + "','" + age + "','" + course + "','" + Class + "','"
+ AvrgMark + "')", conn);
                int rows = comand.ExecuteNonQuery();
                conn.Close();
                return rows;

            }
        }
```

Метод за показване на всички студенти:

```
[WebMethod]
        public DataTable ShowAllStudents()
        {
            using(conn=new SqlConnection(connections))
            {
                conn.Open();
                var commd = new SqlCommand("Select * from StudentInfos", conn);
                SqlDataReader rdr = commd.ExecuteReader();
                DataTable table = new DataTable("Students");
                table.Load(rdr);
                conn.Close();
                return table;

            }
        }
```

Метод за изтриване на студент:

```
[WebMethod]
        public int RemoveStudent(int StudentID)
        {
            using(conn=new SqlConnection(connections))
```

```
            {
                conn.Open();
                var commd = new SqlCommand("Delete from StudentInfos where id=" +
StudentID, conn);
                int row = commd.ExecuteNonQuery();
                conn.Close();
                return row;
            }
        }
```

Метод за актуализиране на информацията при промяна на даден запис:

```
[WebMethod]
        public int refresh(int StudentID, string name, int age, int course, string Class,
double AvrgMark)
        {
            using (conn = new SqlConnection(connections))
            {
                conn.Open();
                var commd=new SqlCommand("Update StudentInfos set Name'" + name + "',
Age='" + age + "',Course = '" + course + "',Class='" + Class + "'," +
                    "AverageMark='" + AvrgMark + "'", conn);
                int row = commd.ExecuteNonQuery();
                conn.Close();
                return row;
            }
        }
```

Също така в този проект има е една html-страница, в която отново са само декларирани основните полета за всеки метод от уеб сървиса.

Пример за метода за добавяне на студент:

```
<form action="StudentInfoSOAP.asmx/AddNewStudent" target="_blank" method="post">
        <table>
            <tr>
                <td>
                    Name:
                </td>
                <td>
                    <input class="frmInput" type="text" size="30" name="name" />
                </td>
            </tr>

            <tr>
                <td>
                    Age:
                </td>
                <td>
                    <input class="frmInput" type="text" size="30" name="age" />
                </td>
            </tr>

            <tr>
                <td>
```

```
                Course:
            </td>
            <td>
                <input class="frmInput" type="text" size="30" name="course" />
            </td>
        </tr>

        <tr>
            <td>
                Class:
            </td>
            <td>
                <input class="frmInput" type="text" size="30" name="Class" />
            </td>
        </tr>

        <tr>
            <td>
                Average Mark:
            </td>
            <td>
                <input class="frmInput" type="text" size="30" name="AvrgMark" />
            </td>
        </tr>

        <tr>
            <td></td>
            <td align="right">
                <input type="submit" value="Submit" class="button" />
            </td>
        </tr>

    </table>

</form>
```

Един от най-важните компоненти в проекта също е тук- връзката към базата с данни:

```
string connections = "Data Source=DESKTOP-RV4S75D; Initial Catalog=StudentInfo";
      SqlConnection conn;
```

Този проект е качен на сървър, в случая IIS Express,с определено URL.

Така след като се намира на сървър на нашия компютър, вторият проект- Client Project, може да достъпва него и неговите функционалности чрез референция.

Добавена е Service Reference към адреса на нашия уеб сървис и вторият проект вече може да се възползва от неговите методи и логика.

В проекта клиент имаме няколко папки. Първата е Model с класа Student. Класът представлява модела на приложението с всичк.и полета:

```
public class Student
    {
        private int _id;
        private string _name;
        private int _age;
        private int _course;
        private string _class;
        private double _averageMark;

        public int ID
        {
            get
            {
            return _id;
            }
            set
            {
                _id = value;
            }
        }

        public string Name
        {
            get
            {
                return _name;
            }
            set
            {
                _name = value;
            }
        }
```

.....

И един празен конструктор: `public Student() { }`

Втората е Controllers. Там се намират класовете AddNew.cs, Refresh.cs, StudentInfo.cs.

В класа AddNew се намира логиката за добавяне на студент, като се използва референция към уеб сървиса:

```
ServiceReference2.StudentInfoSOAPSoapClient servc = new
ServiceReference2.StudentInfoSOAPSoapClient();
```

И логиката на класа:

```
private Student _student;
        public Student Student
        {
            get
            {
                return _student;
            }
            set
```

```
            {
                _student = value;
            }
        }

        public void addNewStudent()
        {
            ServiceReference2.StudentInfoSOAPSoapClient servc = new
ServiceReference2.StudentInfoSOAPSoapClient();
            string name = Student.Name;
            int age = Student.Age;
            int course = Student.Course;
            string classes = Student.Class;
            double avrgmark = Student.AverageMark;
            StudentInfo info = new StudentInfo();
            List<Student> students = info.ShowAllStudentsInData();
            string ok = "";
            foreach(Student s in students)
            {
                if (name == s.Name)
                {
                    ok = "ok";
                }
                if (ok.Equals(""))
                {
                    servc.AddNewStudent(name, age, course, classes, avrgmark);
                }
            }

        }
```

Класът Refresh.cs използвам за поместване на логиката на показването на информацията за даден студент и възможността за нейното актуализиране след това.

```
public class Refresh
    {
        private string _name;
        public string Name
        {
            get
            {
                return _name;
            }
            set
            {
                _name = value;
            }
        }
        private Student _student;
        public Student student
        {
            get
            {
                return _student;
            }
            set
```

```
            {
                _student = value;
            }
        }

        public void refresh()
        {
            ServiceReference2.StudentInfoSOAPSoapClient service = new
ServiceReference2.StudentInfoSOAPSoapClient();
            int id = student.ID;
            string name = student.Name;
            int age = student.Age;
            int course = student.Course;
            string classes = student.Class;
            double avrgMarks = student.AverageMark;
            service.refresh(id, name, age, course, classes, avrgMarks);

        }

        public Student getStudent()
        {
            StudentInfo stdInf = new StudentInfo();
            List<Student> students = stdInf.ShowAllStudentsInData();
            Student returned = new Student();
            string name = Name;
            foreach(Student s in students)
            {
                if (name.Equals(s.Name))
                {
                    returned = s;
                    student = s;
                }
            }
            return returned;
        }

    }
```

И последният клас от тази папка е StudentInfo.cs. Може би беше редно да започна с него, защото там се крие основната част от методите, които след това се използват чрез обекти в другите класове. Първо декларирам колекции от студенти и от имената им и още две стойности за търсена дума и избран студент от списък:

```
private string _searchW;
        private string _selected;
        private List<Student> _students;
        private List<string> _nme;

        public List<Student> Students
        {
            get
            {
                return _students;
            }
```

```
            set
            {
                _students = value;
            }
        }

        public List<string> Nme
        {
            get
            {
                return _nme;
            }
            set
            {
                _nme = value;
            }
        }

        public string SearchWord
        {
            get
            {
                return _searchW;
            }
            set
            {
                _searchW = value;
            }
        }

        public string SelectedOne
        {
            get
            {
                return _selected;
            }
            set
            {
                _selected = value;
            }
        }
```

<u>След това е методът за показване на всички студент в базата данни:</u>

```
public List<Student> ShowAllStudentsInData()
        {
            ServiceReference2.StudentInfoSOAPSoapClient NewService = new
ServiceReference2.StudentInfoSOAPSoapClient();
            var enumerbl = NewService.ShowAllStudents().AsEnumerable();
            List<Student> allSt = new List<Student>();
            allSt =
                (from item in enumerbl
                 select new Student
                 {
                     ID = item.Field<int>("Id"),
                     Name = item.Field<string>("Name"),
                     Age = item.Field<int>("Age"),
                     Course = item.Field<int>("Course"),
```

```
                    Class = item.Field<string>("Class"),
                    AverageMark = item.Field<double>("AverageMark")
                }).ToList();
            return allSt;

        }
```

Логиката за търсене на студент по име или по факултет(Class):

```
public List<Student> SearchStudent(string searchOption)
        {
            string Sword = null;
            Sword = SearchWord;
            string name = null;
            int age = 0;
            int course = 0;
            string classes = null;
            double avrgMrk = 0;
            List<Student> fndtStudent = new List<Student>();
            if (searchOption.Equals("Name"))
            {
                name = SearchWord;

            }
            else if (searchOption.Equals("Class"))
            {
                classes = SearchWord;
            }
            List<Student> allStudents = ShowAllStudentsInData();

            foreach (Student searchedStudnt in allStudents)
            {
                if (SearchWord.Equals(""))
                {
                    fndtStudent = allStudents;
                }

                else if (name != null && name.Equals(searchedStudnt.Name))
                {
                    fndtStudent.Add(searchedStudnt);
                }
                else if (classes!=null&&classes.Equals(searchedStudnt.Class))
                {
                    fndtStudent.Add(searchedStudnt);
                }

            }
            Students = fndtStudent;
            return Students;

        }
```

Методът за изкарване на всички имена, ползваме го за началната страница:

```
public List<string> ShowAllNames(string srch)
        {
            List<Student> students = SearchStudent(srch);
            List<Student> show = new List<Student>();
```

```
            List<string> names = new List<string>();
            if (students.Equals(null))
            {
                show = ShowAllStudentsInData();
            }
            else
            {
                show = Students;
            }
            string name = null;
            foreach(Student st in show)
            {
                name = st.Name;
                names.Add(name);
            }
            Nme = names;
            return Nme;
        }
```

И разбира се и <u>методът за премахване на студент</u>:

```
        public void removeStudent()
        {
            ServiceReference2.StudentInfoSOAPSoapClient service = new
ServiceReference2.StudentInfoSOAPSoapClient();
            int ID = 0;
            string selected = SelectedOne;
            List<Student> students = ShowAllStudentsInData();
            foreach(Student std in students)
            {
                if (selected.Equals(std.Name))
                {
                    ID = std.ID;
                    service.RemoveStudent(ID);
                }
            }
        }
```

В проекта ClientProject имаме и още две уеб страници:

AllStudents.aspx и Review.aspx.

При стартиране ни се показва страницата AllStudents.aspx.

Предаването на информация и преминаването от страница в страница става чрез бутоните. Логика им се намира в code-behind-а на страницата:

<u>За бутона за търсене:</u>

```
protected void ShowAll()
        {
            string choose = DropDown.SelectedValue;
            string searchedFor = searching.Text;
            StudentInfo stdInf = new StudentInfo();
            stdInf.SearchWord = searchedFor;
```

```
            List<string> list = stdInf.ShowAllNames(choose);
            names.DataSource = null;
            names.DataSource = list;
            names.DataBind();
        }

        protected void btnSrc_Click(object sender, EventArgs e)
        {
            ShowAll();
        }
```

За бутона за изтриване:

```
protected void btnDelete_Click(object sender, EventArgs e)
        {
            if (names.SelectedIndex > 0)
            {
                ListViewItem item = names.Items[names.SelectedIndex];
                Label lbl = (Label)item.FindControl("lblSel");
                string finalResult = lbl.Text;
                StudentInfo stInf = new StudentInfo();
                stInf.SelectedOne = finalResult;
                stInf.removeStudent();

            }
            else
            {
                lblErr.Text = "You must select an Item!";
            }
        }
```

За избиране на студент от списъка:

```
protected void names_SelectedIndexChanged(object sender, EventArgs e)
        {
            if (names.SelectedIndex >= 1)
            {
                //view state//
            }
            else
            {
                ViewState["SelectedKey"] = null;            }
        }
```

За преглеждане на повече информация за студента:

```
protected void btnOvrvw_Click(object sender, EventArgs e)
        {
            if (names.SelectedIndex > 0)
            {
                ListViewItem item = names.Items[names.SelectedIndex];
                Label lbl = (Label)item.FindControl("lblSel");
                string finalResult = lbl.Text;
                if (finalResult != null)
                {
                    Session["sel"] = finalResult;
                    Response.Redirect("Review.aspx");
```

```
            }
        }
        else
        {
            lblErr.Text = "You must select an Item!";
        }
    }
```

При избиране на бутона View Details с цел преглед на по-подробна информация за студента, ние сме прехвърлени към друга страница- Review.

В нея при зареждането на страницата се зарежда информацията за студента, който сме избрали:

```
Refresh up = new Refresh();
Student student = new Student();

    protected void Page_Load(object sender, EventArgs e)
    {
        string field = (string)(Session["sel"]);
        up.Name = field;
        student = up.getStudent();
        display(field);
    }

public void display(string field)
    {
        up.Name = field;
        student = up.getStudent();
        if (!IsPostBack)
        {
            txtName.Text = student.Name;
            txtAge.Text = student.Age.ToString();
            txtCourse.Text = student.Course.ToString();
            txtClass.Text = student.Class.ToString();
            txtAverageMark.Text = student.AverageMark.ToString();

        }
    }
```

Има опция за промяна на някои от данните. Промените се нанасят директно в полето и когато сме готови, натискаме Refresh.:

```
protected void btnRefresh_Click(object sender, EventArgs e)
    {
        Student NewStudent = up.student;
        student.ID = NewStudent.ID;
        student.Name = txtName.Text;
        student.Age = Convert.ToInt32(txtAge.Text);
        student.Course = Convert.ToInt32(txtCourse.Text);
        student.Class = txtClass.Text;
        student.AverageMark = Convert.ToDouble(txtAverageMark.Text);
        up.refresh();
    }
```

И накрая имаме опцията и за добавяне на нов студент. Отново информацията се нанася директно с полетата и се натиска Add New Student- бутона:

```
protected void btnAddNew_Click(object sender, EventArgs e)
        {
            AddNew add = new AddNew();
            student.Name = txtName.Text;
            student.Age = Convert.ToInt32(txtAge.Text);
            student.Course = Convert.ToInt32(txtCourse.Text);
            student.Class = txtClass.Text;
            student.AverageMark = Convert.ToDouble(txtAverageMark.Text);
            add.Student = student;
            add.addNewStudent();
        }
```

Заключение

В заключение мога да кажа, че уеб страницата изпълнява основните цели, които й бях поставила при проектирането й. Достъпът до инаформацията е лесен и бърз, интерфейсът не е сложен. Има още много функционалности, които могат да се добавят и също така интерфейсът да се модернизира.

Другото, което би подлежало на корекция от моя страна е базата данни. В момента тя е локална и е достъпна само то този компютър. По-доброто решение би било да се качи на сървър.

Литература

1. http://81.161.243.12/bgmoodle/pluginfile.php/15656/mod_resource/content/1/Upr4%20PRS%20SOAP.pdf
2. https://support.microsoft.com/en-us/help/942062/-http-error-403-14---forbidden-error-when-you-open-an-iis-7-0-webpage
3. http://searchmicroservices.techtarget.com/definition/SOAP-Simple-Object-Access-Protocol

Приложение

StudentInfoBase

StudentInfoSOAP.asmx.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.Services;
using System.Data.SqlClient;
using System.Data;

namespace StudentInfoBase
{
    /// <summary>
    /// Summary description for StudentInfoSOAP
    /// </summary>
    [WebService(Namespace = "http://tempuri.org/")]
    [WebServiceBinding(ConformsTo = WsiProfiles.BasicProfile1_1)]
    [System.ComponentModel.ToolboxItem(false)]
    // To allow this Web Service to be called from script, using ASP.NET AJAX, uncomment
the following line.
    // [System.Web.Script.Services.ScriptService]
    public class StudentInfoSOAP : System.Web.Services.WebService
    {
        string connections = "Data Source=DESKTOP-RV4S75D; Initial Catalog=StudentInfo";
        SqlConnection conn;

        [WebMethod]
        public int AddNewStudent(string name, int age, int course, string Class, double
AvrgMark)
        {
            using(conn=new SqlConnection(connections))
            {
                conn.Open();
                var comand = new SqlCommand("Insert into StudentInfos(Name, Age, Course,
class, avrgMark) values ('" + name + "','" + age + "','" + course + "','" + Class + "','"
+ AvrgMark + "')", conn);
                int rows = comand.ExecuteNonQuery();
                conn.Close();
                return rows;

            }
        }

        [WebMethod]
        public DataTable ShowAllStudents()
        {
            using(conn=new SqlConnection(connections))
            {
                conn.Open();
                var commd = new SqlCommand("Select * from StudentInfos", conn);
                SqlDataReader rdr = commd.ExecuteReader();
                DataTable table = new DataTable("Students");
```

```
                table.Load(rdr);
                conn.Close();
                return table;

            }
        }

        [WebMethod]
        public int RemoveStudent(int StudentID)
        {
            using(conn=new SqlConnection(connections))
            {
                conn.Open();
                var commd = new SqlCommand("Delete from StudentInfos where id=" +
StudentID, conn);
                int row = commd.ExecuteNonQuery();
                conn.Close();
                return row;
            }
        }

        [WebMethod]
        public int refresh(int StudentID, string name, int age, int course, string Class,
double AvrgMark)
        {
            using (conn = new SqlConnection(connections))
            {
                conn.Open();
                var commd=new SqlCommand("Update StudentInfos set Name'" + name + "',
Age='" + age + "',Course = '" + course + "',Class='" + Class + "'," +
                    "AverageMark='" + AvrgMark + "'", conn);
                int row = commd.ExecuteNonQuery();
                conn.Close();
                return row;
            }
        }
    }
}
```

HtmlPage1.html

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8" />
    <title>ServiceWeb</title>
</head>
<body>

    <form action="StudentInfoSOAP.asmx/AddNewStudent" target="_blank" method="post">
        <table>
            <tr>
                <td>
                    Name:
                </td>
                <td>
                    <input class="frmInput" type="text" size="30" name="name" />
```

```
            </td>
        </tr>

        <tr>
            <td>
                Age:
            </td>
            <td>
                <input class="frmInput" type="text" size="30" name="age" />
            </td>
        </tr>

        <tr>
            <td>
                Course:
            </td>
            <td>
                <input class="frmInput" type="text" size="30" name="course" />
            </td>
        </tr>

        <tr>
            <td>
                Class:
            </td>
            <td>
                <input class="frmInput" type="text" size="30" name="Class" />
            </td>
        </tr>

        <tr>
            <td>
                Average Mark:
            </td>
            <td>
                <input class="frmInput" type="text" size="30" name="AvrgMark" />
            </td>
        </tr>

        <tr>
            <td></td>
            <td align="right">
                <input type="submit" value="Submit" class="button" />
            </td>
        </tr>

    </table>

</form>
<br />

<form action="StudentInfoSOAP.asmx/ShowAllStudents" target="_blank" method="post">
    <input type="submit" value="Submit" class="button" />
</form>
<br />
<form action="StudentInfoSOAP.asmx/RemoveStudent" target="_blank" method="post">
    <table>
        <tr>
```

```
            <td>
                ID:
            </td>
            <td>
                <input class="frmInput" type="text" size="30" name="StudentID" />
            </td>
        </tr>
        <tr>
            <td></td>
            <td align="right">
                <input type="submit" value="Submit" class="button" />
            </td>
        </tr>
    </table>
</form>

<form action="StudentInfoSOAP.asmx/refresh" target="_blank" method="post">
    <table>
        <tr>
            <td>
                ID:
            </td>
            <td>
                <input class="frmInput" type="text" size="30" name="StudentID" />
            </td>
        </tr>
        <tr>
            <td>
                Name:
            </td>
            <td>
                <input class="frmInput" type="text" size="30" name="name" />
            </td>
        </tr>
        <tr>
            <td>
                Age:
            </td>
            <td>
                <input class="frmInput" type="text" size="30" name="age" />
            </td>
        </tr>

        <tr>
            <td>
                Course:
            </td>
            <td>
                <input class="frmInput" type="text" size="30" name="course" />
            </td>
        </tr>

        <tr>
            <td>
                Class:
            </td>
            <td>
                <input class="frmInput" type="text" size="30" name="Class" />
```

```
                </td>
            </tr>

            <tr>
                <td>
                    Average Mark:
                </td>
                <td>
                    <input class="frmInput" type="text" size="30" name="AvrgMark" />
                </td>
            </tr>
            <tr>
                <td></td>
                <td align="right">
                    <input type="submit" value="Submit" class="button" />
                </td>
            </tr>
        </table>
    </form>

</body>
       </html>
```

ClientProject

AddNew.cs

```
using ClientProject.Model;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;

namespace ClientProject.Controllers
{
    public class AddNew
    {
        private Student _student;
        public Student Student
        {
            get
            {
                return _student;
            }
            set
            {
                _student = value;
            }
        }

        public void addNewStudent()
        {
```

```
            ServiceReference2.StudentInfoSOAPSoapClient servc = new
ServiceReference2.StudentInfoSOAPSoapClient();
            string name = Student.Name;
            int age = Student.Age;
            int course = Student.Course;
            string classes = Student.Class;
            double avrgmark = Student.AverageMark;
            StudentInfo info = new StudentInfo();
            List<Student> students = info.ShowAllStudentsInData();
            string ok = "";
            foreach(Student s in students)
            {
                if (name == s.Name)
                {
                    ok = "ok";
                }
                if (ok.Equals(""))
                {
                    servc.AddNewStudent(name, age, course, classes, avrgmark);
                }
            }

        }
    }
       }
```

Refresh.cs

```
using ClientProject.Model;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;

namespace ClientProject.Controllers
{
    public class Refresh
    {
        private string _name;
        public string Name
        {
            get
            {
                return _name;
            }
            set
            {
                _name = value;
            }
        }
        private Student _student;
        public Student student
        {
            get
            {
```

```
                return _student;
            }
            set
            {
                _student = value;
            }
        }

        public void refresh()
        {
            ServiceReference2.StudentInfoSOAPSoapClient service = new
ServiceReference2.StudentInfoSOAPSoapClient();
            int id = student.ID;
            string name = student.Name;
            int age = student.Age;
            int course = student.Course;
            string classes = student.Class;
            double avrgMarks = student.AverageMark;
            service.refresh(id, name, age, course, classes, avrgMarks);

        }

        public Student getStudent()
        {
            StudentInfo stdInf = new StudentInfo();
            List<Student> students = stdInf.ShowAllStudentsInData();
            Student returned = new Student();
            string name = Name;
            foreach(Student s in students)
            {
                if (name.Equals(s.Name))
                {
                    returned = s;
                    student = s;
                }
            }
            return returned;
        }

    }
       }
```

StudentInfo.cs

```
using ClientProject.Model;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Net;
using System.Net.Http;
using System.Web.Http;
using System.Data;

namespace ClientProject.Controllers
{
    public class StudentInfo : ApiController
```

```
    {
        private string _searchW;
        private string _selected;
        private List<Student> _students;
        private List<string> _nme;

        public List<Student> Students
        {
            get
            {
                return _students;
            }
            set
            {
                _students = value;
            }
        }

        public List<string> Nme
        {
            get
            {
                return _nme;
            }
            set
            {
                _nme = value;
            }
        }

        public string SearchWord
        {
            get
            {
                return _searchW;
            }
            set
            {
                _searchW = value;
            }
        }

        public string SelectedOne
        {
            get
            {
                return _selected;
            }
            set
            {
                _selected = value;
            }
        }

        public List<Student> ShowAllStudentsInData()
        {
            ServiceReference2.StudentInfoSOAPSoapClient NewService = new
ServiceReference2.StudentInfoSOAPSoapClient();
```

```
        var enumerbl = NewService.ShowAllStudents().AsEnumerable();
        List<Student> allSt = new List<Student>();
        allSt =
            (from item in enumerbl
             select new Student
             {
                 ID = item.Field<int>("Id"),
                 Name = item.Field<string>("Name"),
                 Age = item.Field<int>("Age"),
                 Course = item.Field<int>("Course"),
                 Class = item.Field<string>("Class"),
                 AverageMark = item.Field<double>("AverageMark")
             }).ToList();
        return allSt;

    }

    public List<Student> SearchStudent(string searchOption)
    {
        string Sword = null;
        Sword = SearchWord;
        string name = null;
        int age = 0;
        int course = 0;
        string classes = null;
        double avrgMrk = 0;
        List<Student> fndtStudent = new List<Student>();
        if (searchOption.Equals("Name"))
        {
            name = SearchWord;

        }
        else if (searchOption.Equals("Class"))
        {
            classes = SearchWord;
        }
        List<Student> allStudents = ShowAllStudentsInData();

        foreach (Student searchedStudnt in allStudents)
        {
            if (SearchWord.Equals(""))
            {
                fndtStudent = allStudents;
            }

            else if (name != null && name.Equals(searchedStudnt.Name))
            {
                fndtStudent.Add(searchedStudnt);
            }
            else if (classes!=null&&classes.Equals(searchedStudnt.Class))
            {
                fndtStudent.Add(searchedStudnt);
            }

        }
        Students = fndtStudent;
        return Students;
```

```
        }

        public List<string> ShowAllNames(string srch)
        {
            List<Student> students = SearchStudent(srch);
            List<Student> show = new List<Student>();
            List<string> names = new List<string>();
            if (students.Equals(null))
            {
                show = ShowAllStudentsInData();
            }
            else
            {
                show = Students;
            }
            string name = null;
            foreach(Student st in show)
            {
                name = st.Name;
                names.Add(name);
            }
            Nme = names;
            return Nme;
        }

        public void removeStudent()
        {
            ServiceReference2.StudentInfoSOAPSoapClient service = new
ServiceReference2.StudentInfoSOAPSoapClient();
            int ID = 0;
            string selected = SelectedOne;
            List<Student> students = ShowAllStudentsInData();
            foreach(Student std in students)
            {
                if (selected.Equals(std.Name))
                {
                    ID = std.ID;
                    service.RemoveStudent(ID);
                }
            }
        }

    }
       }
```

Student.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;

namespace ClientProject.Model
{
    public class Student
    {
        private int _id;
```

```
private string _name;
private int _age;
private int _course;
private string _class;
private double _averageMark;

public int ID
{
    get
    {
    return _id;
    }
    set
    {
        _id = value;
    }
}

public string Name
{
    get
    {
        return _name;
    }
    set
    {
        _name = value;
    }
}

public int Age
{
    get
    {
        return _age;
    }
    set
    {
        _age = value;
    }
}

public int Course
{
    get
    {
        return _course;
    }
    set
    {
        _course = value;
    }
}

public string Class
{
    get
    {
```

```
                return _class;
            }
            set
            {
                _class = value;
            }
        }

        public double AverageMark
        {
            get
            {
                return _averageMark;
            }
            Set


            {
                _averageMark = value;
            }
        }

        public Student() { }
    }


   }
```

AllStudents.aspx

```
<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="AllStudents.aspx.cs"
Inherits="ClientProject.AllStudents" %>

<!DOCTYPE html>

<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
    <title></title>
</head>
<body>
    <form id="form1" runat="server">
        <br />
        <asp:Label ID="lbl" runat="server" Width="400" Font-Bold Font-Size="Large"
Text="Students" style="margin-left:264px" />
        <br />
        <br />
        <asp:Label ID="lbl2" runat="server" Text="Search for Student:" Height="16px"
style="margin-left:0px; margin-top: 0px" Width="115px" Font-Bold></asp:Label>
        <asp:TextBox ID="searching" runat="server" Width="129px" Height="16px"
style="margin-left: 12px"></asp:TextBox>
        <asp:DropDownList ID="DropDown" runat="server">
            <asp:ListItem Enabled="true" Text="Search by: " Value="1"></asp:ListItem>
            <asp:ListItem Text="Name" Value="name"></asp:ListItem>
            <asp:ListItem Text="Classes" Value="classes"></asp:ListItem>
```

```
        </asp:DropDownList>
        <br />
        <br />
        <asp:ListView ID="names" runat="server"
OnSelectedIndexChanging="names_SelectedIndexChanging"
OnSelectedIndexChanged="names_SelectedIndexChanged">
        <LayoutTemplate>
           <table runat="server" id="tblCategories"
                 cellspacing="0" cellpadding="1" width="440px" border="1">
            <tr id="itemPlaceholder" runat="server"></tr>
          </table>
        </LayoutTemplate>
        <ItemTemplate>
            <tr runat="server">
                    <td>
                   <asp:LinkButton ID="btnSel" runat="server" Text="Select"
CommandName="Select"/>
                    </td>
                    <td style="width:300px">
                        <asp:Label ID="lblSel" Text="<%#Container.DataItem%>"
runat="server"/>
                    </td>
                </tr>
        </ItemTemplate>
         <SelectedItemTemplate>
                <tr runat="server" style="background-color:#90EE90">

                   <td> </td>
                    <td style="width:300px">
                        <asp:Label ID="Label1" Text="<%#Container.DataItem%>"
runat="server"/>
                    </td>
                </tr>
            </SelectedItemTemplate>
        </asp:ListView>
        <br />
         <asp:Label ID="lblErr" runat="server" ForeColor="Red" Font-Bold></asp:Label>
        <br />
        <br />


        <asp:Button ID="btnSrch" runat="server" Text="Search" Width="90px"
OnClick="btnSrc_Click" Font-Bold/>


        <asp:Button ID="btnDelete" runat="server" style="margin-left: 29px" Text="Delete"
Width="100px" OnClick="btnDelete_Click" Font-Bold />


        <asp:Button ID="btnOvrvw" runat="server" style="margin-left: 34px" Text="View
Details" Width="100px" OnClick="btnOvrvw_Click" Font-Bold />

    </form>
</body>
</html>
```

AllStudents.aspx.cs

```
using ClientProject.Controllers;
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;

namespace ClientProject
{
    public partial class AllStudents : System.Web.UI.Page
    {
        protected void Page_Load(object sender, EventArgs e)
        {
            lblErr.Text = "";
            ShowAll();
        }

        protected void ShowAll()
        {
            string choose = DropDown.SelectedValue;
            string searchedFor = searching.Text;
            StudentInfo stdInf = new StudentInfo();
            stdInf.SearchWord = searchedFor;
            List<string> list = stdInf.ShowAllNames(choose);
            names.DataSource = null;
            names.DataSource = list;
            names.DataBind();
        }

        protected void btnSrc_Click(object sender, EventArgs e)
        {
            ShowAll();
        }

        protected void btnDelete_Click(object sender, EventArgs e)
        {
            if (names.SelectedIndex > 0)
            {
                ListViewItem item = names.Items[names.SelectedIndex];
                Label lbl = (Label)item.FindControl("lblSel");
                string finalResult = lbl.Text;
                StudentInfo stInf = new StudentInfo();
                stInf.SelectedOne = finalResult;
                stInf.removeStudent();

            }
            else
            {
                lblErr.Text = "You must select an Item!";
            }
        }

        protected void names_SelectedIndexChanged(object sender, EventArgs e)
        {
```

```
            if (names.SelectedIndex >= 1)
            {
                //view state//
            }
            else
            {
                ViewState["SelectedKey"] = null;            }
        }

        protected void btnOvrvw_Click(object sender, EventArgs e)
        {
            if (names.SelectedIndex > 0)
            {
                ListViewItem item = names.Items[names.SelectedIndex];
                Label lbl = (Label)item.FindControl("lblSel");
                string finalResult = lbl.Text;
                if (finalResult != null)
                {
                    Session["sel"] = finalResult;
                    Response.Redirect("Review.aspx");

                }
            }
            else
            {
                lblErr.Text = "You must select an Item!";
            }
        }
    }
       }
```

Review.aspx

```
<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="Review.aspx.cs"
Inherits="ClientProject.WebPage.Review" %>

<!DOCTYPE html>

<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
    <title></title>
     <style type="text/css">
         #form1 {
            width: 776px;
            height: 501px;
        }
    </style>
</head>
<body>
    <form id="form1" runat="server">
        <br />
        <br />
         <asp:Label ID="lbl" runat="server" Width="400 " Font-Bold Font-Italic Font-
Size="X-Large" Text="Student " style="margin-left: 264px"/>
        <br />
        <br />
        <asp:Label Text="Name: " width="90px" runat="server"></asp:Label>
```

```
        <asp:TextBox ID="txtName" runat="server" Width="200px"></asp:TextBox>
        <br />
        <asp:Label Text="Age: " runat="server" width="90px"></asp:Label>
        <asp:TextBox ID="txtAge" runat="server" Width="201px"></asp:TextBox>
        <br />
        <asp:Label Text="Course: " runat="server" Width="90px"></asp:Label>
        <asp:TextBox ID="txtCourse" runat="server" Width="200px"></asp:TextBox>
        <br />
        <asp:Label Text="Class: " runat="server" Width="90px"></asp:Label>
        <asp:TextBox ID="txtClass" runat="server" Width="200px"></asp:TextBox>
        <br />
        <asp:Label Text="Average Mark: " runat="server" Width="90px"></asp:Label>
        <asp:TextBox ID="txtAverageMark" runat="server" Width="200px"></asp:TextBox>
        <br />
        <br />
        <br />
        <asp:Button ID="btnRefresh" runat="server" Text="Refresh" Width="86px"
OnClick="btnRefresh_Click" />
        <asp:Button ID="btnAddNew" runat="server" style="margin-left: 48px" Text="Add new
Student " Width="115px" OnClick="btnAddNew_Click" />



    </form>
</body>
</html>
```

Review.aspx.cs

```
using ClientProject.Controllers;
using ClientProject.Model;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;

namespace ClientProject.WebPage
{
    public partial class Review : System.Web.UI.Page
    {


        Refresh up = new Refresh();
        Student student = new Student();
        protected void Page_Load(object sender, EventArgs e)
        {
            string field = (string)(Session["sel"]);
            up.Name = field;
            student = up.getStudent();
            display(field);
        }

        public void display(string field)
        {
            up.Name = field;
```

```
            student = up.getStudent();
            if (!IsPostBack)
            {
                txtName.Text = student.Name;
                txtAge.Text = student.Age.ToString();
                txtCourse.Text = student.Course.ToString();
                txtClass.Text = student.Class.ToString();
                txtAverageMark.Text = student.AverageMark.ToString();

            }
        }

        protected void btnRefresh_Click(object sender, EventArgs e)
        {
            Student NewStudent = up.student;
            student.ID = NewStudent.ID;
            student.Name = txtName.Text;
            student.Age = Convert.ToInt32(txtAge.Text);
            student.Course = Convert.ToInt32(txtCourse.Text);
            student.Class = txtClass.Text;
            student.AverageMark = Convert.ToDouble(txtAverageMark.Text);
            up.refresh();
        }

        protected void btnAddNew_Click(object sender, EventArgs e)
        {
            AddNew add = new AddNew();
            student.Name = txtName.Text;
            student.Age = Convert.ToInt32(txtAge.Text);
            student.Course = Convert.ToInt32(txtCourse.Text);
            student.Class = txtClass.Text;
            student.AverageMark = Convert.ToDouble(txtAverageMark.Text);
            add.Student = student;
            add.addNewStudent();
        }
    }
   }
```