Practice Project

ASP.NET WEB API Application

Source Code:

BAL Library:

marks.cs:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace BAL
{
    public class marks
    {
        public int student_id { get; set; }
        public string student_name { get; set; }
        public int subject_marks { get; set; }
    }
}
```

DAL Library:

Student_DAL.cs:

```
using BAL;
using System;
using System.Collections.Generic;
using System.Data.SqlClient;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace DAL
{
    public class Student_DAL
    {
        public bool Insert(marks school)
        {
            SqlConnection cn = new SqlConnection("Data Source=DESKTOP-
5GL4B5D\\SQLEXPRESS1;Initial Catalog=School;Integrated Security=True");

            SqlCommand cmdInsert = new SqlCommand("insert into
marks(student_id,student_name,subject_marks)
values(@student_id,@student_name,@subject_marks)", cn);
            cmdInsert.Parameters.AddWithValue("@student_id", school.student_id);
            cmdInsert.Parameters.AddWithValue("@student_name", school.student_name);
            cmdInsert.Parameters.AddWithValue("@subject_marks",
school.subject_marks);
```

```
            cn.Open();
            int i = cmdInsert.ExecuteNonQuery();

            bool status = false;

            if (i == 1)
            {
                status = true;
            }

            cn.Close();//finally
            cn.Dispose();//finally
            return status;



        }
        public bool Update(marks school)
        {

                        SqlConnection cn = new SqlConnection("Data Source=DESKTOP-
5GL4B5D\\SQLEXPRESS1;Initial Catalog=School;Integrated Security=True");

            SqlCommand cmdUpdate = new SqlCommand("[dbo].[Updatemarks]", cn);

            cmdUpdate.CommandType = System.Data.CommandType.StoredProcedure;
            cmdUpdate.Parameters.AddWithValue("@p_studid", school.student_id);
            cmdUpdate.Parameters.AddWithValue("@p_studname", school.student_name);
            cmdUpdate.Parameters.AddWithValue("@p_submarks", school.subject_marks);
            cn.Open();
            int s = cmdUpdate.ExecuteNonQuery();
            bool statusd = false;
            if (s == 1)
            {
                statusd = true;
            }
            cn.Close();//finally
            cn.Dispose();//finally
            return statusd;

        }

        public marks Find(int id)
        {
            SqlConnection cn = new SqlConnection("Data Source=DESKTOP-
5GL4B5D\\SQLEXPRESS1;Initial Catalog=School;Integrated Security=True");


            SqlCommand cmdSelect = new SqlCommand("[dbo].sp_Findmarks", cn);

            cmdSelect.CommandType = System.Data.CommandType.StoredProcedure;
            cmdSelect.Parameters.AddWithValue("@p_studid", id);

            SqlParameter p1 = new SqlParameter();
            p1.ParameterName = "@p_mark_studname";
```

```
            p1.SqlDbType = System.Data.SqlDbType.NVarChar;
            p1.Size = 30;
            p1.Direction = System.Data.ParameterDirection.Output;
            cmdSelect.Parameters.Add(p1);


            SqlParameter p2 = new SqlParameter();
            p2.ParameterName = "@p_marks_submarks";
            p2.SqlDbType = System.Data.SqlDbType.Int;
            p2.Size = 20;
            p2.Direction = System.Data.ParameterDirection.Output;
            cmdSelect.Parameters.Add(p2);




            cn.Open();
            cmdSelect.ExecuteNonQuery();

            marks found = new marks();

            found.student_name = p1.Value.ToString();
            found.subject_marks = Convert.ToInt32(p2.Value);




            cn.Close();
            cn.Dispose();


            return found;



        }
        public List<marks> List()
        {


            SqlConnection cn = new SqlConnection("Data Source=DESKTOP-
5GL4B5D\\SQLEXPRESS1;Initial Catalog=School;Integrated Security=True");

            SqlCommand cmdlist = new SqlCommand("select
student_id,student_name,subject_marks from marks", cn);

            cn.Open();
            SqlDataReader dr = cmdlist.ExecuteReader();
            List<marks> emplist = new List<marks>();
            if (dr.HasRows)
            {
                while (dr.Read())
                {
                    marks bal = new marks();
                    bal.student_id = Convert.ToInt32(dr["student_id"]);
                    bal.student_name = dr["student_name"].ToString();
```

```
                    bal.subject_marks = Convert.ToInt32(dr["subject_marks"]);


                    emplist.Add(bal);
                }
            }
            cn.Close();
            cn.Dispose();
            return emplist;

        }
        public bool Delete(int stuid)
        {
            SqlConnection cn = new SqlConnection("Data Source=DESKTOP-
5GL4B5D\\SQLEXPRESS1;Initial Catalog=School;Integrated Security=True");

            SqlCommand cmdDelete = new SqlCommand("[dbo].sp_Deletemarks", cn);
            cmdDelete.CommandType = System.Data.CommandType.StoredProcedure;
            cmdDelete.Parameters.AddWithValue("@p_id", stuid);
            cn.Open();
            int i = cmdDelete.ExecuteNonQuery();
            bool status = false;
            if (i == 1)
            {
                status = true;
            }
            cn.Close();//finally
            cn.Dispose();//finally
            return status;

        }




    }
}
```

Student_Helper:

```
using BAL;
using DAL;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace Helper
{
    public class Student_Helper
    {
        Student_DAL dal = null;
        public Student_Helper()
        {
            dal = new Student_DAL();
        }
```

```
        public bool AddE(marks school)
        {
            return dal.Insert(school);

        }

        public bool Edit(marks school)
        {
            return dal.Update(school);
        }

        public marks search(int id)
        {
            return dal.Find(id);
        }
        public List<marks> BList()
        {
            return dal.List();
        }
        public bool remove(int id)
        {
            return dal.Delete(id);
        }
    }
}
```

SubController.cs:

```
using API.Models;
using BAL;
using Helper;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Net;
using System.Net.Http;
using System.Web.Http;

namespace API.Controllers
{
    public class SubController : ApiController
    {
        Student_Helper obj = null;
        public SubController()
        {
            obj = new Student_Helper();
        }

        // [Route("GetAllMarks")]
        [HttpGet]
        public List<sub_mark> GetMarkList()
        {


            List<marks> empbal = new List<marks>();
```

```
            empbal = obj.BList();
            List<sub_mark> emps = new List<sub_mark>();
            foreach (var item in empbal)
            {
                //Employees emp = new Employees();
                emps.Add(new sub_mark { student_id = item.student_id, student_name =
item.student_name, subject_marks = item.subject_marks });
            }
            return emps;

        }




        // GET api/<controller>/5
        // [Route("~/FindE/{id}")]
        //  [Route("FindById/{id:int:min(1)}")]

       [Route("FindById/{id:int?}")]
        public sub_mark GetMarkByID(int id = 1)
        {
            marks empbal = new marks();
            empbal = obj.search(id);
            sub_mark emp = new sub_mark();
            //emp.Id = empbal.Id;
            emp.student_id = id;
            emp.student_name = empbal.student_name;
            emp.subject_marks = empbal.subject_marks;

            return emp;

        }

        // POST api/<controller>
        public HttpResponseMessage PostMarks([FromBody] sub_mark empdata)
        {
            marks empbal = new marks();
            empbal.student_id = empdata.student_id;
            empbal.student_name = empdata.student_name;
            empbal.subject_marks = empdata.subject_marks;


            bool ans = obj.AddE(empbal);
            if (ans)
            {
                return Request.CreateResponse(HttpStatusCode.OK);
            }
            else
            {
                return Request.CreateResponse(HttpStatusCode.NotAcceptable);
            }

        }

        // PUT api/<controller>/5
```

```
        public HttpResponseMessage PutMarks([FromBody] sub_mark empdata)
        {

            marks empbal = new marks();
            empbal.student_id = empdata.student_id;
            empbal.student_name = empdata.student_name;
            empbal.subject_marks = empdata.subject_marks;



            bool ans = obj.Edit(empbal);
            if (ans)
            {
                return Request.CreateResponse(HttpStatusCode.OK);
            }
            else
            {
                return Request.CreateResponse(HttpStatusCode.NotAcceptable);
            }
        }

        // DELETE api/<controller>/5
        public HttpResponseMessage DeleteProduct(int id)
        {
            bool ans = obj.remove(id);
            if (ans)
            {
                return Request.CreateResponse(HttpStatusCode.OK);
            }
            else
            {
                return Request.CreateResponse(HttpStatusCode.NotAcceptable);
            }

        }


    }
}
```

API:

Sub_mark.cs:

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

namespace API.Models
{
    public class sub_mark
    {
        public int student_id { get; set; }
        public string student_name { get; set; }
        public int subject_marks { get; set; }
```

```
    }
}
```

Clientdemo:

Markcontroller:

```
using Clientdemo.Models;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Net.Http;
using System.Web;
using System.Web.Mvc;

namespace Clientdemo.Controllers
{
    public class markController : Controller
    {
        // GET: mark
        public ActionResult Index()
        {
            List<mark> emplist = new List<mark>();
            using (var client = new HttpClient())
            {
                client.BaseAddress = new Uri("https://localhost:44300/api/");

                var responseTask = client.GetAsync("Sub");
                responseTask.Wait();
                var result = responseTask.Result;
                if (result.IsSuccessStatusCode)
                {
                    var readData = result.Content.ReadAsAsync<mark[]>();
                    readData.Wait();
                    var empdata = readData.Result;
                    foreach (var item in empdata)
                    {
                        emplist.Add(new mark
                        {
                            student_id = item.student_id,
                            student_name = item.student_name,
                            subject_marks = item.subject_marks
                        });

                    }
                }
            }
            return View(emplist);

        }
        public ActionResult Create()
        {

            return View();
        }
```

```
        [HttpPost]

        public ActionResult Create(mark empmodel)
        {


            using (var client = new HttpClient())
            {
                client.BaseAddress = new Uri("https://localhost:44300/api/Sub");

                var emp = new mark
                {
                    student_id = empmodel.student_id,
                    student_name = empmodel.student_name,
                    subject_marks = empmodel.subject_marks
                };

                var postTask = client.PostAsJsonAsync<mark>(client.BaseAddress,
emp);
                postTask.Wait();
                var result = postTask.Result;
                if (result.IsSuccessStatusCode)
                {
                    var readtaskResult = result.Content.ReadAsAsync<mark>();

                    readtaskResult.Wait();
                    var dataInserted = readtaskResult.Result;
                }


            }

            return RedirectToAction("Index");
        }
    }
}
```

Mark.cs:

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

namespace Clientdemo.Models
{
    public class mark
    {
        public int student_id { get; set; }
        public string student_name { get; set; }
        public int subject_marks { get; set; }

    }
}
```

Views:

Index.cshtml:

```
@model IEnumerable<Clientdemo.Models.mark>

@{
    ViewBag.Title = "Index";
}

<h2>Index</h2>

<p>
    @Html.ActionLink("Create New", "Create")
</p>
<table class="table">
    <tr>
        <th>
            @Html.DisplayNameFor(model => model.student_id)
        </th>
        <th>
            @Html.DisplayNameFor(model => model.student_name)
        </th>
        <th>
            @Html.DisplayNameFor(model => model.subject_marks)
        </th>
        <th></th>
    </tr>

@foreach (var item in Model) {
    <tr>
        <td>
            @Html.DisplayFor(modelItem => item.student_id)
        </td>
        <td>
            @Html.DisplayFor(modelItem => item.student_name)
        </td>
        <td>
            @Html.DisplayFor(modelItem => item.subject_marks)
        </td>
        <td>
            @Html.ActionLink("Edit", "Edit", new {  id=item.student_id }) |
            @Html.ActionLink("Details", "Details", new {  id=item.student_id }) |
            @Html.ActionLink("Delete", "Delete", new { id=item.student_id })
        </td>
    </tr>
}

</table>
```

Create.cshtml:

```
@model Clientdemo.Models.mark

@{
    ViewBag.Title = "Create";
```

```
}

<h2>Create</h2>


@using (Html.BeginForm())
{
    @Html.AntiForgeryToken()

    <div class="form-horizontal">
        <h4>mark</h4>
        <hr />
        @Html.ValidationSummary(true, "", new { @class = "text-danger" })
        <div class="form-group">
            @Html.LabelFor(model => model.student_id, htmlAttributes: new { @class =
"control-label col-md-2" })
            <div class="col-md-10">
                @Html.EditorFor(model => model.student_id, new { htmlAttributes =
new { @class = "form-control" } })
                @Html.ValidationMessageFor(model => model.student_id, "", new {
@class = "text-danger" })
            </div>
        </div>

        <div class="form-group">
            @Html.LabelFor(model => model.student_name, htmlAttributes: new { @class
= "control-label col-md-2" })
            <div class="col-md-10">
                @Html.EditorFor(model => model.student_name, new { htmlAttributes =
new { @class = "form-control" } })
                @Html.ValidationMessageFor(model => model.student_name, "", new {
@class = "text-danger" })
            </div>
        </div>

        <div class="form-group">
            @Html.LabelFor(model => model.subject_marks, htmlAttributes: new {
@class = "control-label col-md-2" })
            <div class="col-md-10">
                @Html.EditorFor(model => model.subject_marks, new { htmlAttributes =
new { @class = "form-control" } })
                @Html.ValidationMessageFor(model => model.subject_marks, "", new {
@class = "text-danger" })
            </div>
        </div>

        <div class="form-group">
            <div class="col-md-offset-2 col-md-10">
                <input type="submit" value="Create" class="btn btn-default" />
            </div>
        </div>
    </div>
}

<div>
    @Html.ActionLink("Back to List", "Index")
</div>
```

```
@section Scripts {
    @Scripts.Render("~/bundles/jqueryval")
}
```