**Blog Tracker Application**

**Capstone Project**

**Source code:**

**BAL:**

**BLClass1.cs:**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace BLL

{

public class BLClass1

{

public string EmailId { get; set; }

public string Name { get; set; }

public DateTime DateOfJoining { get; set; }

public int PassCode { get; set; }

}

public class BLClass2

{

public int BlogId { get; set; }

public string Title { get; set; }

public string subject { get; set; }

public DateTime DateOfCreation { get; set; }

public string BlogUrl { get; set; }

public string EmpEmailId { get; set; }

}

}

**DAL:**

using System;

using System.Collections.Generic;

using System.ComponentModel.DataAnnotations;

using System.Data.Entity;

using System.Linq;

using System.Runtime.Remoting.Contexts;

using System.Text;

using System.Threading.Tasks;

namespace DAL

{

public class AdminInfo

{

[Key]

[Required()]

public string EmailId { get; set; }

[Required()]

[MaxLength(20, ErrorMessage = "Maximum 20 characters only")]

public string pass { get; set; }

}

public class EmpInfo

{

[Key]

[Required()]

public string EmailId { get;set; }

[MaxLength(20, ErrorMessage = "Not allowed above 20 chars")]

[MinLength(2, ErrorMessage = "Not allowed below 2 chars")]

public string Name { get; set; }

public DateTime DateOfJoining { get; set; }

public int PassCode { get; set; }

//1 Book---M Enrollments(members)

public virtual ICollection<BlogInfo> I { get; set; }

// [DataType(DataType.DateTime,ErrorMessage ="not valid date")]

}

public class BlogInfo

{

[Key]

[Required]

public int BlogId { get; set; }

public string Title { get; set; }

public string subject { get; set; }

public DateTime DateOfCreation { get; set; }

[Required()]

public string BlogUrl { get; set; }

[EmailAddress]

public string EmpEmailId { get; set; }

//1 Book---M Enrollments(members)

public virtual ICollection<EmpInfo> e { get; set; }

// [DataType(DataType.DateTime,ErrorMessage ="not valid date")]

}

public class EmpDBInitializer : DropCreateDatabaseIfModelChanges<MyContext1>

{

protected override void Seed(MyContext1 context)

{

IList<AdminInfo> defaultStandards = new List<AdminInfo>();

defaultStandards.Add(new AdminInfo() { EmailId="suki123@gmail.com",pass="suki123" });

context.AdminTable.AddRange(defaultStandards);

base.Seed(context);

}

}

public class MyContext1 : DbContext

{

public MyContext1() : base("MyContext1")

{

//createdatabase if not exists

//drop create always

//drop create if model changes

Database.SetInitializer<MyContext1>(new DropCreateDatabaseIfModelChanges<MyContext1>());

Database.SetInitializer(new EmpDBInitializer());

}

public virtual DbSet<EmpInfo> EmpTable { get; set; }

public virtual DbSet<AdminInfo> AdminTable { get; set; }

public virtual DbSet<BlogInfo> BlogTable { get; set; }

}

}

**EmpOperation.cs:**

using BLL;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace DAL

{

public class EmpOperation

{

public List<BLClass1> GetAllEmployeeDetails()

{

MyContext1 context = new MyContext1();

List<EmpInfo> clist = context.EmpTable.ToList();

List<BLClass1> cblist = new List<BLClass1>();

foreach (var item in clist)

{

cblist.Add(new BLClass1 { EmailId=item.EmailId,Name=item.Name,DateOfJoining=item.DateOfJoining,PassCode=item.PassCode });

}

return cblist;

}

public bool Insert(BLClass1 bal)

{

try

{

MyContext1 context = new MyContext1();

EmpInfo b = new EmpInfo();

b.EmailId = bal.EmailId;

b.Name = bal.Name;

b.DateOfJoining = bal.DateOfJoining;

b.PassCode = bal.PassCode;

context.EmpTable.Add(b);

context.SaveChanges();

return true;

}

catch (Exception ex)

{

return false;

}

}

public bool DeleteEmployeeDetails(int id)

{

try

{

MyContext1 context = new MyContext1();

List<EmpInfo> s = context.EmpTable.ToList();

EmpInfo r = s.Find(pr => pr.PassCode == id);

context.EmpTable.Remove(r);

context.SaveChanges();

return true;

}

catch (Exception ex)

{

return false;

}

}

public BLClass1 search(int id)

{

MyContext1 context = new MyContext1();

List<EmpInfo> customers = context.EmpTable.ToList();

EmpInfo obj = customers.Find(cust => cust.PassCode== id);

// List<BLClass1> cblist = new List<BLClass1>();

BLClass1 b = new BLClass1();

b.EmailId = obj.EmailId;

b.Name = obj.Name;

b.DateOfJoining = obj.DateOfJoining;

b.PassCode = obj.PassCode;

return b;

//context.SaveChanges();

}

public bool UpdateEmployeeDetails(BLClass1 bal)

{

try

{

MyContext1 context = new MyContext1();

List<EmpInfo> customers = context.EmpTable.ToList();

EmpInfo obj = customers.Find(cust => cust.PassCode == bal.PassCode);

obj.Name = bal.Name;

obj.DateOfJoining = bal.DateOfJoining;

obj.PassCode = bal.PassCode;

// context.Updatebookdetails();

context.SaveChanges();

return true;

}

catch (Exception ex)

{

return false;

}

}

}

}

**BlogOperation.cs:**

using BLL;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace DAL

{

public class BlogOperation

{

public List<BLClass2> GetAllDetails()

{

MyContext1 context = new MyContext1();

List<BlogInfo> clist = context.BlogTable.ToList();

List<BLClass2> cblist = new List<BLClass2>();

foreach (var item in clist)

{

cblist.Add(new BLClass2 { BlogId=item.BlogId,Title=item.Title,subject=item.subject,BlogUrl=item.BlogUrl,DateOfCreation=item.DateOfCreation,EmpEmailId=item.EmpEmailId });

}

return cblist;

}

public bool Insert(BLClass2 bal)

{

try

{

MyContext1 context = new MyContext1();

BlogInfo b = new BlogInfo();

b.BlogId = bal.BlogId;

b.Title = bal.Title;

b.subject = bal.subject;

b.BlogUrl = bal.BlogUrl;

b.DateOfCreation = bal.DateOfCreation;

b.EmpEmailId = bal.EmpEmailId;

context.BlogTable.Add(b);

context.SaveChanges();

return true;

}

catch (Exception ex)

{

return false;

}

}

public bool DeleteDetails(int id)

{

try

{

MyContext1 context = new MyContext1();

List<BlogInfo> s = context.BlogTable.ToList();

BlogInfo r = s.Find(pr => pr.BlogId == id);

context.BlogTable.Remove(r);

context.SaveChanges();

return true;

}

catch (Exception ex)

{

return false;

}

}

public BLClass2 search(int id)

{

MyContext1 context = new MyContext1();

List<BlogInfo> customers = context.BlogTable.ToList();

BlogInfo obj = customers.Find(cust => cust.BlogId == id);

// List<BLClass1> cblist = new List<BLClass1>();

BLClass2 b = new BLClass2();

b.BlogId = obj.BlogId;

b.Title = obj.Title;

b.subject = obj.subject;

b.BlogUrl = obj.BlogUrl;

b.DateOfCreation=obj.DateOfCreation;

b.EmpEmailId=obj.EmpEmailId;

return b;

//context.SaveChanges();

}

public bool UpdateDetails(BLClass2 bal)

{

try

{

MyContext1 context = new MyContext1();

List<BlogInfo> customers = context.BlogTable.ToList();

BlogInfo obj = customers.Find(cust => cust.BlogId == bal.BlogId);

obj.Title = bal.Title;

obj.subject = bal.subject;

obj.BlogUrl = bal.BlogUrl;

obj.DateOfCreation=bal.DateOfCreation;

obj.EmpEmailId = bal.EmpEmailId;

// context.Updatebookdetails();

context.SaveChanges();

return true;

}

catch (Exception ex)

{

return false;

}

}

}

}

**Unit Testing:**

**Admin\_Test.cs:**

using DAL;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Test

{

public class Admin\_test

{

MyContext1 db = new MyContext1();

public bool check()

{

bool ans = false;

var found = db.AdminTable.ToList();

if (found[0].EmailId == "suki123@gmail.com" && found[0].pass == "suki123")

{

ans = true;

}

return ans;

}

}

}

**Blogapp\_Test.cs:**

using DAL;

using NUnit.Framework;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Test

{

[TestFixture]

public class blogapp\_test

{

[TestCase]

public void Employee\_mail\_Test()

{

MyContext1 db = new MyContext1();

var found = db.AdminTable.ToList();

Assert.AreEqual("suki123@gmail.com", found[0].EmailId);

Console.WriteLine("Validation success");

}

[TestCase]

public void Admin\_password\_Test()

{

MyContext1 db = new MyContext1();

var found = db.AdminTable.ToList();

Assert.AreEqual("suki123", found[0].pass);

Console.WriteLine("Validation success");

}

Admin\_test t = new Admin\_test();

[TestCase]

public void AdminTest()

{

t.check();

}

}

}

**BlogController.cs:**

using BLL;

using blogs.Models;

using DAL;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Net;

using System.Net.Http;

using System.Web.Http;

namespace blogs.Controllers

{

public class BlogController : ApiController

{

// GET api/<controller>

BlogOperation obj = null;

public BlogController()

{

obj = new BlogOperation();

}

// [Route("GetAllMarks")]

[HttpGet]

public List<bl> GetBlogList()

{

//bl --model

//BLclass1

List<BLClass2> empbal = new List<BLClass2>();

empbal = obj.GetAllDetails();

List<bl> emps = new List<bl>();

foreach (var item in empbal)

{

//Employees emp = new Employees();

emps.Add(new bl { BlogId = item.BlogId, Title = item.Title, subject = item.subject, BlogUrl = item.BlogUrl, DateOfCreation = item.DateOfCreation ,EmpEmailId=item.EmpEmailId});

}

return emps;

}

public HttpResponseMessage PostDetails([FromBody] bl empdata)

{

BLClass2 empbal = new BLClass2();

empbal.BlogId = empdata.BlogId;

empbal.Title = empdata.Title;

empbal.subject = empdata.subject;

empbal.BlogUrl = empdata.BlogUrl;

empbal.DateOfCreation= empdata.DateOfCreation;

empbal.EmpEmailId = empdata.EmpEmailId;

bool ans = obj.Insert(empbal);

if (ans)

{

return Request.CreateResponse(HttpStatusCode.OK);

}

else

{

return Request.CreateResponse(HttpStatusCode.NotAcceptable);

}

}

[Route("FindById/{id:int?}")]

public bl GetBlogByID(int id)

{

BLClass2 empbal = new BLClass2();

empbal = obj.search(id);

bl emp = new bl();

//emp.Id = empbal.Id;

emp.BlogId= id;

emp.Title = empbal.Title;

emp.subject = empbal.subject;

emp.BlogUrl = empbal.BlogUrl;

emp.DateOfCreation= empbal.DateOfCreation;

emp.EmpEmailId=empbal.EmpEmailId;

return emp;

}

public HttpResponseMessage PutDetails([FromBody] bl empdata)

{

BLClass2 empbal = new BLClass2();

empbal.BlogId = empdata.BlogId;

empbal.Title = empdata.Title;

empbal.subject = empdata.subject;

empbal.BlogUrl = empdata.BlogUrl;

empbal.DateOfCreation=empdata.DateOfCreation;

empbal.EmpEmailId=empdata.EmpEmailId;

bool ans = obj.UpdateDetails(empbal);

if (ans)

{

return Request.CreateResponse(HttpStatusCode.OK);

}

else

{

return Request.CreateResponse(HttpStatusCode.NotAcceptable);

}

}

// DELETE api/<controller>/5

public HttpResponseMessage DeleteBlogDetails(int id)

{

bool ans = obj.DeleteDetails(id);

if (ans)

{

return Request.CreateResponse(HttpStatusCode.OK);

}

else

{

return Request.CreateResponse(HttpStatusCode.NotAcceptable);

}

}

}

}

EmpsController.cs:

using BLL;

using blogs.Models;

using DAL;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Net;

using System.Net.Http;

using System.Web.Http;

namespace blogs.Controllers

{

public class EmpsController : ApiController

{

// GET api/<controller>

EmpOperation obj = null;

public EmpsController()

{

obj = new EmpOperation();

}

// [Route("GetAllMarks")]

[HttpGet]

public List<Emp> GetEmployeeList()

{

//sub\_mark --model

//BLclass1

List<BLClass1> empbal = new List<BLClass1>();

empbal = obj.GetAllEmployeeDetails();

List<Emp> emps = new List<Emp>();

foreach (var item in empbal)

{

//Employees emp = new Employees();

emps.Add(new Emp { EmailId = item.EmailId, Name = item.Name, DateOfJoining = item.DateOfJoining, PassCode = item.PassCode });

}

return emps;

}

public HttpResponseMessage PostEmpDetails([FromBody] Emp empdata)

{

BLClass1 empbal = new BLClass1();

empbal.EmailId = empdata.EmailId;

empbal.Name = empdata.Name;

empbal.DateOfJoining = empdata.DateOfJoining;

empbal.PassCode = empdata.PassCode;

bool ans = obj.Insert(empbal);

if (ans)

{

return Request.CreateResponse(HttpStatusCode.OK);

}

else

{

return Request.CreateResponse(HttpStatusCode.NotAcceptable);

}

}

[Route("FindById/{id:int?}")]

public Emp GetEmployeeByID(int id)

{

BLClass1 empbal = new BLClass1();

empbal = obj.search(id);

Emp emp = new Emp();

//emp.Id = empbal.Id;

emp.EmailId = empbal.EmailId;

emp.Name = empbal.Name;

emp.DateOfJoining= empbal.DateOfJoining;

emp.PassCode= id;

return emp;

}

public HttpResponseMessage PutEmpDetails([FromBody] Emp empdata)

{

BLClass1 empbal = new BLClass1();

empbal.EmailId = empdata.EmailId;

empbal.Name = empdata.Name;

empbal.DateOfJoining = empdata.DateOfJoining;

empbal.PassCode = empdata.PassCode;

bool ans = obj.UpdateEmployeeDetails(empbal);

if (ans)

{

return Request.CreateResponse(HttpStatusCode.OK);

}

else

{

return Request.CreateResponse(HttpStatusCode.NotAcceptable);

}

}

// DELETE api/<controller>/5

public HttpResponseMessage DeleteEmpDetails(int id)

{

bool ans = obj.DeleteEmployeeDetails(id);

if (ans)

{

return Request.CreateResponse(HttpStatusCode.OK);

}

else

{

return Request.CreateResponse(HttpStatusCode.NotAcceptable);

}

}

}

}

Home:

using DAL;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Net.Http;

using System.Net;

using System.Web;

using System.Web.Http;

using System.Web.Mvc;

using BLL;

using blogs.Models;

namespace blogs.Controllers

{

public class HomeController : Controller

{

public ActionResult Index()

{

ViewBag.Title = "Home Page";

return View();

}

}

}

**Model:**

**Emp.cs:**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

namespace blogs.Models

{

public class Emp

{

public string EmailId { get; set; }

public string Name { get; set; }

public DateTime DateOfJoining { get; set; }

public int PassCode { get; set; }

}

public class bl

{

public int BlogId { get; set; }

public string Title { get; set; }

public string subject { get; set; }

public DateTime DateOfCreation { get; set; }

public string BlogUrl { get; set; }

public string EmpEmailId { get; set; }

}

}

bloguiController.cs:

using DAL;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Net.Http;

using System.Runtime.Remoting.Messaging;

using System.Web;

using System.Web.Mvc;

using UI.Models;

namespace UI.Controllers

{

public class bloguiController : Controller

{

// GET: blogui

MyContext1 db = new MyContext1();

BlogOperation ob = new BlogOperation();

public ActionResult Login()

{

return View();

}

[HttpPost]

public ActionResult Login(EmpModel log)

{

var user = db.EmpTable.Where(x => x.EmailId == log.EmailId && x.PassCode == log.PassCode).Count();

if (user > 0)

{

return RedirectToAction("Index");

}

else

{

return View();

}

}

public ActionResult Index()

{

List<Blogmodel> emplist = new List<Blogmodel>();

using (var client = new HttpClient())

{

client.BaseAddress = new Uri("https://localhost:44320/api/");

var responseTask = client.GetAsync("Blog");

responseTask.Wait();

var result = responseTask.Result;

if (result.IsSuccessStatusCode)

{

var readData = result.Content.ReadAsAsync<Blogmodel[]>();

readData.Wait();

var empdata = readData.Result;

foreach (var item in empdata)

{

emplist.Add(new Blogmodel

{

BlogId = item.BlogId,

Title = item.Title,

subject = item.subject,

DateOfCreation = item.DateOfCreation,

BlogUrl = item.BlogUrl,

EmpEmailId = item.EmpEmailId

});

}

}

}

return View(emplist);

}

public ActionResult Create()

{

return View();

}

[HttpPost]

public ActionResult Create(Blogmodel empmodel)

{

using (var client = new HttpClient())

{

client.BaseAddress = new Uri("https://localhost:44320//api/Blog");

var emp = new Blogmodel

{

BlogId = empmodel.BlogId,

Title = empmodel.Title,

subject = empmodel.subject,

DateOfCreation = empmodel.DateOfCreation,

BlogUrl = empmodel.BlogUrl,

EmpEmailId = empmodel.EmpEmailId

};

var postTask = client.PostAsJsonAsync<Blogmodel>(client.BaseAddress, emp);

postTask.Wait();

var result = postTask.Result;

if (result.IsSuccessStatusCode)

{

var readtaskResult = result.Content.ReadAsAsync<Blogmodel>();

readtaskResult.Wait();

var dataInserted = readtaskResult.Result;

}

}

return RedirectToAction("Index");

}

public ActionResult EditBlog(int id)

{

var emp = ob.search(id);

Blogmodel model = new Blogmodel();

model.BlogId = id;

model.Title = emp.Title;

model.subject = emp.subject;

model.DateOfCreation = emp.DateOfCreation;

model.BlogUrl = emp.BlogUrl;

model.EmpEmailId = emp.EmpEmailId;

return View(model);

}

// POST: Emp/Edit/5

[HttpPost]

public ActionResult EditBlog(int id, FormCollection collection)

{

try

{

// TODO: Add update logic here

var emp = ob.search(id);

emp.BlogId = Convert.ToInt32(Request["BlogId"]);

emp.Title = Request["Title"].ToString();

emp.subject = Request["subject"].ToString();

emp.DateOfCreation = Convert.ToDateTime(Request["DateOfCreation"]);

emp.BlogUrl = Request["BlogUrl"].ToString();

emp.EmpEmailId = Request["EmpEmailId"].ToString();

bool ans = ob.UpdateDetails(emp);

if (ans)

{

return RedirectToAction("Index");

}

else

{

return View();

}

}

catch

{

return View();

}

}

public ActionResult DeleteBlog(int id)

{

var emp = ob.search(id);

Blogmodel model = new Blogmodel();

model.BlogId = id;

model.Title = emp.Title;

model.subject = emp.subject;

model.DateOfCreation = emp.DateOfCreation;

model.BlogUrl = emp.BlogUrl;

model.EmpEmailId = emp.EmpEmailId;

return View(model);

}

// POST: Emp/Delete/5

[HttpPost]

public ActionResult DeleteBlog(int id, FormCollection collection)

{

try

{

// TODO: Add delete logic here

var dataFound = ob.search(id);

if (dataFound != null)

{

bool ans = ob.DeleteDetails(id);

if (ans)

{

return RedirectToAction("Index");

}

else

{

return View();

}

}

return RedirectToAction("Index");

}

catch

{

return View();

}

}

public ActionResult BlogDetails(int id)

{

var data = ob.search(id);

Blogmodel emp = new Blogmodel();

emp.BlogId = id;

emp.Title = data.Title;

emp.subject = data.subject;

emp.DateOfCreation = data.DateOfCreation;

emp.BlogUrl = data.BlogUrl;

emp.EmpEmailId = data.EmpEmailId;

return View(emp);

}

}

}

EmpUIController.cs:

using DAL;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Net.Http;

using System.Runtime.Remoting.Messaging;

using System.Web;

using System.Web.Mvc;

using UI.Models;

namespace UI.Controllers

{

public class EmpUIController : Controller

{

// GET: EmpUI

MyContext1 db = new MyContext1();

EmpOperation e = new EmpOperation();

public ActionResult Login()

{

return View();

}

[HttpPost]

public ActionResult Login(AdminModel log)

{

var user = db.AdminTable.Where(x => x.EmailId == log.EmailId && x.pass == log.Pass).Count();

if (user > 0)

{

return RedirectToAction("Index");

}

else

{

return View();

}

}

public ActionResult Index()

{

List<EmpModel> emplist = new List<EmpModel>();

using (var client = new HttpClient())

{

client.BaseAddress = new Uri("https://localhost:44320/api/");

var responseTask = client.GetAsync("Emps");

responseTask.Wait();

var result = responseTask.Result;

if (result.IsSuccessStatusCode)

{

var readData = result.Content.ReadAsAsync<EmpModel[]>();

readData.Wait();

var empdata = readData.Result;

foreach (var item in empdata)

{

emplist.Add(new EmpModel

{

EmailId = item.EmailId,

Name = item.Name,

DateOfJoining = item.DateOfJoining,

PassCode = item.PassCode

});

}

}

}

return View(emplist);

}

public ActionResult Create()

{

return View();

}

[HttpPost]

public ActionResult Create(EmpModel empmodel)

{

using (var client = new HttpClient())

{

client.BaseAddress = new Uri("https://localhost:44320/api/Emps");

var emp = new EmpModel

{

EmailId = empmodel.EmailId,

Name = empmodel.Name,

DateOfJoining = empmodel.DateOfJoining,

PassCode = empmodel.PassCode

};

var postTask = client.PostAsJsonAsync<EmpModel>(client.BaseAddress, emp);

postTask.Wait();

var result = postTask.Result;

if (result.IsSuccessStatusCode)

{

var readtaskResult = result.Content.ReadAsAsync<EmpModel>();

readtaskResult.Wait();

var dataInserted = readtaskResult.Result;

}

}

return RedirectToAction("Index");

}

public ActionResult EditEmp(int id)

{

var emp = e.search(id);

EmpModel model = new EmpModel();

model.PassCode = id;

model.EmailId = emp.EmailId;

model.Name = emp.Name;

model.DateOfJoining = emp.DateOfJoining;

return View(model);

}

// POST: Emp/Edit/5

[HttpPost]

public ActionResult EditEmp(int id, FormCollection collection)

{

try

{

// TODO: Add update logic here

var emp = e.search(id);

emp.PassCode = Convert.ToInt32(Request["PassCode"]);

emp.EmailId = Request["EmailId"].ToString();

emp.Name = Request["Name"].ToString();

emp.DateOfJoining = Convert.ToDateTime(Request["DateOfJoining"]);

bool ans = e.UpdateEmployeeDetails(emp);

if (ans)

{

return RedirectToAction("Index");

}

else

{

return View();

}

}

catch

{

return View();

}

}

public ActionResult Delete(int id)

{

var emp = e.search(id);

EmpModel model = new EmpModel();

model.EmailId = emp.EmailId;

model.Name = emp.Name;

model.DateOfJoining = emp.DateOfJoining;

model.PassCode = emp.PassCode;

return View(model);

}

// POST: Emp/Delete/5

[HttpPost]

public ActionResult Delete(int id, FormCollection collection)

{

try

{

// TODO: Add delete logic here

var dataFound = e.search(id);

if (dataFound != null)

{

bool ans = e.DeleteEmployeeDetails(id);

if (ans)

{

return RedirectToAction("Index");

}

else

{

return View();

}

}

return RedirectToAction("Index");

}

catch

{

return View();

}

}

public ActionResult Details(int id)

{

var data = e.search(id);

EmpModel model = new EmpModel();

model.EmailId = data.EmailId;

model.Name = data.Name;

model.DateOfJoining = data.DateOfJoining;

model.PassCode = data.PassCode;

return View(model);

}

}

}

Home:

using System;

using System.Collections.Generic;

using System.Linq;

using System.Net.Http;

using System.Web;

using System.Web.Mvc;

using UI.Models;

namespace UI.Controllers

{

public class HomeController : Controller

{

public ActionResult Index()

{

List<Blogmodel> emplist = new List<Blogmodel>();

using (var client = new HttpClient())

{

client.BaseAddress = new Uri("https://localhost:44320/api/");

var responseTask = client.GetAsync("Blog");

responseTask.Wait();

var result = responseTask.Result;

if (result.IsSuccessStatusCode)

{

var readData = result.Content.ReadAsAsync<Blogmodel[]>();

readData.Wait();

var empdata = readData.Result;

foreach (var item in empdata)

{

emplist.Add(new Blogmodel

{

BlogId = item.BlogId,

Title = item.Title,

subject = item.subject,

DateOfCreation = item.DateOfCreation,

BlogUrl = item.BlogUrl,

EmpEmailId = item.EmpEmailId

});

}

}

}

return View(emplist);

}

}

}

AdminModel.cs:

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

namespace UI.Models

{

public class AdminModel

{

public string EmailId { get; set; }

public string Pass { get; set; }

}

}

EmpModel.cs:

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

namespace UI.Models

{

public class EmpModel

{

public string EmailId { get; set; }

public string Name { get; set; }

public DateTime DateOfJoining { get; set; }

public int PassCode { get; set; }

}

}

BlogModel.cs:

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

namespace UI.Models

{

public class Blogmodel

{

public int BlogId { get; set; }

public string Title { get; set; }

public string subject { get; set; }

public DateTime DateOfCreation { get; set; }

public string BlogUrl { get; set; }

public string EmpEmailId { get; set; }

}

}

Views:

Blogui:

Index.cshtml:

@model IEnumerable<UI.Models.Blogmodel>

@{

ViewBag.Title = "Index";

}

<h2><center>List of Blogs</center></h2>

<p>

@Html.ActionLink("Add New Blog", "Create")

</p>

<table class="table">

<tr>

<th>

@Html.DisplayNameFor(model => model.BlogId)

</th>

<th>

@Html.DisplayNameFor(model => model.Title)

</th>

<th>

@Html.DisplayNameFor(model => model.subject)

</th>

<th>

@Html.DisplayNameFor(model => model.DateOfCreation)

</th>

<th>

@Html.DisplayNameFor(model => model.BlogUrl)

</th>

<th></th>

</tr>

@foreach (var item in Model) {

<tr>

<td>

@Html.DisplayFor(modelItem => item.BlogId)

</td>

<td>

@Html.DisplayFor(modelItem => item.Title)

</td>

<td>

@Html.DisplayFor(modelItem => item.subject)

</td>

<td>

@Html.DisplayFor(modelItem => item.DateOfCreation)

</td>

<td><mark bgcolor="green">

@Html.DisplayFor(modelItem => item.BlogUrl)

</mark>

</td>

<td>

@Html.ActionLink("EditBlog", "EditBlog", new { id = item.BlogId }) |

@Html.ActionLink("DeleteBlog", "DeleteBlog", new { id = item.BlogId }) |

@Html.ActionLink("BlogDetails", "BlogDetails", new { id = item.BlogId })

</td>

</tr>

}

</table>

Create.cshtml:

@model UI.Models.Blogmodel

@{

ViewBag.Title = "Create";

}

<h2>Add Blog Details</h2>

@using (Html.BeginForm())

{

@Html.AntiForgeryToken()

<div class="form-horizontal">

<h4>Blog</h4>

<hr />

@Html.ValidationSummary(true, "", new { @class = "text-danger" })

@\*<div class="form-group">

@Html.LabelFor(model => model.BlogId, htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

@Html.EditorFor(model => model.BlogId, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.BlogId, "", new { @class = "text-danger" })

</div>

</div>\*@

<div class="form-group">

@Html.LabelFor(model => model.Title, htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

@Html.EditorFor(model => model.Title, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.Title, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

@Html.LabelFor(model => model.subject, htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

@Html.EditorFor(model => model.subject, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.subject, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

@Html.LabelFor(model => model.DateOfCreation, htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

@Html.EditorFor(model => model.DateOfCreation, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.DateOfCreation, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

@Html.LabelFor(model => model.BlogUrl, htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

@Html.EditorFor(model => model.BlogUrl, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.BlogUrl, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

@Html.LabelFor(model => model.EmpEmailId, htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

@Html.EditorFor(model => model.EmpEmailId, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.EmpEmailId, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

<div class="col-md-offset-2 col-md-10">

<input type="submit" value="Add Blog" class="btn btn-success" />

</div>

</div>

</div>

}

<div>

@Html.ActionLink("Back to List", "Index")

</div>

@section Scripts {

@Scripts.Render("~/bundles/jqueryval")

}

EditBlog.cshtml:

@model UI.Models.Blogmodel

@{

ViewBag.Title = "EditBlog";

}

<h2>EditBlog</h2>

@using (Html.BeginForm())

{

@Html.AntiForgeryToken()

<div class="form-horizontal">

<h4>Blogmodel</h4>

<hr />

@Html.ValidationSummary(true, "", new { @class = "text-danger" })

<div class="form-group">

@Html.LabelFor(model => model.BlogId, htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

@Html.EditorFor(model => model.BlogId, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.BlogId, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

@Html.LabelFor(model => model.Title, htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

@Html.EditorFor(model => model.Title, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.Title, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

@Html.LabelFor(model => model.subject, htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

@Html.EditorFor(model => model.subject, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.subject, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

@Html.LabelFor(model => model.DateOfCreation, htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

@Html.EditorFor(model => model.DateOfCreation, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.DateOfCreation, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

@Html.LabelFor(model => model.BlogUrl, htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

@Html.EditorFor(model => model.BlogUrl, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.BlogUrl, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

@Html.LabelFor(model => model.EmpEmailId, htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

@Html.EditorFor(model => model.EmpEmailId, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.EmpEmailId, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

<div class="col-md-offset-2 col-md-10">

<input type="submit" value="Save" class="btn btn-default" />

</div>

</div>

</div>

}

<div>

@Html.ActionLink("Back to List", "Index")

</div>

@section Scripts {

@Scripts.Render("~/bundles/jqueryval")

}

DetailsBlog.cshtml:

@model UI.Models.Blogmodel

@{

ViewBag.Title = "BlogDetails";

}

<h2>BlogDetails</h2>

<div>

<h4>Blogmodel</h4>

<hr />

<dl class="dl-horizontal">

<dt>

@Html.DisplayNameFor(model => model.BlogId)

</dt>

<dd>

@Html.DisplayFor(model => model.BlogId)

</dd>

<dt>

@Html.DisplayNameFor(model => model.Title)

</dt>

<dd>

@Html.DisplayFor(model => model.Title)

</dd>

<dt>

@Html.DisplayNameFor(model => model.subject)

</dt>

<dd>

@Html.DisplayFor(model => model.subject)

</dd>

<dt>

@Html.DisplayNameFor(model => model.DateOfCreation)

</dt>

<dd>

@Html.DisplayFor(model => model.DateOfCreation)

</dd>

<dt>

@Html.DisplayNameFor(model => model.BlogUrl)

</dt>

<dd>

@Html.DisplayFor(model => model.BlogUrl)

</dd>

<dt>

@Html.DisplayNameFor(model => model.EmpEmailId)

</dt>

<dd>

@Html.DisplayFor(model => model.EmpEmailId)

</dd>

</dl>

</div>

<p>

@Html.ActionLink("Edit", "EditBlog", new { id = Model.BlogId }) |

@Html.ActionLink("Back to List", "Index")

</p>

DeleteBlog.cshtml:

@model UI.Models.Blogmodel

@{

ViewBag.Title = "DeleteBlog";

}

<h2>DeleteBlog</h2>

<h3>Are you sure you want to delete this?</h3>

<div>

<h4>Blogmodel</h4>

<hr />

<dl class="dl-horizontal">

<dt>

@Html.DisplayNameFor(model => model.BlogId)

</dt>

<dd>

@Html.DisplayFor(model => model.BlogId)

</dd>

<dt>

@Html.DisplayNameFor(model => model.Title)

</dt>

<dd>

@Html.DisplayFor(model => model.Title)

</dd>

<dt>

@Html.DisplayNameFor(model => model.subject)

</dt>

<dd>

@Html.DisplayFor(model => model.subject)

</dd>

<dt>

@Html.DisplayNameFor(model => model.DateOfCreation)

</dt>

<dd>

@Html.DisplayFor(model => model.DateOfCreation)

</dd>

<dt>

@Html.DisplayNameFor(model => model.BlogUrl)

</dt>

<dd>

@Html.DisplayFor(model => model.BlogUrl)

</dd>

<dt>

@Html.DisplayNameFor(model => model.EmpEmailId)

</dt>

<dd>

@Html.DisplayFor(model => model.EmpEmailId)

</dd>

</dl>

@using (Html.BeginForm()) {

@Html.AntiForgeryToken()

<div class="form-actions no-color">

<input type="submit" value="DeleteBlog" class="btn btn-default" /> |

@Html.ActionLink("Back to List", "Index")

</div>

}

</div>

Login.cshtml:

@model UI.Models.EmpModel

@{

ViewBag.Title = "Login";

}

<h2>Employee Login page</h2>

@using (Html.BeginForm())

{

@Html.AntiForgeryToken()

<div class="form-horizontal">

<hr />

@Html.ValidationSummary(true, "", new { @class = "text-danger" })

<div class="form-group">

@Html.LabelFor(model => model.EmailId, htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

@Html.EditorFor(model => model.EmailId, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.EmailId, "", new { @class = "text-danger" })

</div>

</div>

@\* <div class="form-group">

@Html.LabelFor(model => model.Name, htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

@Html.EditorFor(model => model.Name, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.Name, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

@Html.LabelFor(model => model.DateOfJoining, htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

@Html.EditorFor(model => model.DateOfJoining, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.DateOfJoining, "", new { @class = "text-danger" })

</div>

</div>\*@

<div class="form-group">

@Html.LabelFor(model => model.PassCode, htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

@Html.EditorFor(model => model.PassCode, new { htmlAttributes = new { @class = "form-control", type = "password" } })

@Html.ValidationMessageFor(model => model.PassCode, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

<div class="col-md-offset-2 col-md-10">

<input type="submit" value="Login" class="btn btn-success" />

</div>

</div>

</div>

}

<div>

@Html.ActionLink("Back to List", "Index")

</div>

@section Scripts {

@Scripts.Render("~/bundles/jqueryval")

}

EmpUI:

Create.cshtml:

@model UI.Models.EmpModel

@{

ViewBag.Title = "Create";

}

<h2>Add Employee Details</h2>

@using (Html.BeginForm())

{

@Html.AntiForgeryToken()

<div class="form-horizontal">

<h4>EmpModel</h4>

<hr />

@Html.ValidationSummary(true, "", new { @class = "text-danger" })

<div class="form-group">

@Html.LabelFor(model => model.EmailId, htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

@Html.EditorFor(model => model.EmailId, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.EmailId, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

@Html.LabelFor(model => model.Name, htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

@Html.EditorFor(model => model.Name, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.Name, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

@Html.LabelFor(model => model.DateOfJoining, htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

@Html.EditorFor(model => model.DateOfJoining, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.DateOfJoining, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

@Html.LabelFor(model => model.PassCode, htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

@Html.EditorFor(model => model.PassCode, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.PassCode, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

<div class="col-md-offset-2 col-md-10">

<input type="submit" value="Add and save" class="btn btn-primary" />

</div>

</div>

</div>

}

<div>

@Html.ActionLink("Back to List", "Index")

</div>

@section Scripts {

@Scripts.Render("~/bundles/jqueryval")

}

Index.cshtml:

@model IEnumerable<UI.Models.EmpModel>

@{

ViewBag.Title = "Index";

}

<h2>Employee List</h2>

<p>

@Html.ActionLink("Add New Employee", "Create")

</p>

<table class="table">

<tr>

<th>

@Html.DisplayNameFor(model => model.EmailId)

</th>

<th>

@Html.DisplayNameFor(model => model.Name)

</th>

<th>

@Html.DisplayNameFor(model => model.DateOfJoining)

</th>

<th>

@Html.DisplayNameFor(model => model.PassCode)

</th>

<th></th>

</tr>

@foreach (var item in Model) {

<tr>

<td>

@Html.DisplayFor(modelItem => item.EmailId)

</td>

<td>

@Html.DisplayFor(modelItem => item.Name)

</td>

<td>

@Html.DisplayFor(modelItem => item.DateOfJoining)

</td>

<td>

@Html.DisplayFor(modelItem => item.PassCode)

</td>

<td>

@Html.ActionLink("EditEmp", "EditEmp", new { id=item.PassCode}) |

@Html.ActionLink("Details", "Details", new { id = item.PassCode }) |

@Html.ActionLink("Delete", "Delete", new { id = item.PassCode })

</td>

</tr>

}

</table>

Delete.cshtml:

@model UI.Models.EmpModel

@{

ViewBag.Title = "Delete";

}

<h2>Delete</h2>

<h3>Are you sure you want to delete this?</h3>

<div>

<h4>EmpModel</h4>

<hr />

<dl class="dl-horizontal">

<dt>

@Html.DisplayNameFor(model => model.EmailId)

</dt>

<dd>

@Html.DisplayFor(model => model.EmailId)

</dd>

<dt>

@Html.DisplayNameFor(model => model.Name)

</dt>

<dd>

@Html.DisplayFor(model => model.Name)

</dd>

<dt>

@Html.DisplayNameFor(model => model.DateOfJoining)

</dt>

<dd>

@Html.DisplayFor(model => model.DateOfJoining)

</dd>

<dt>

@Html.DisplayNameFor(model => model.PassCode)

</dt>

<dd>

@Html.DisplayFor(model => model.PassCode)

</dd>

</dl>

@using (Html.BeginForm()) {

@Html.AntiForgeryToken()

<div class="form-actions no-color">

<input type="submit" value="Delete" class="btn btn-default" /> |

@Html.ActionLink("Back to List", "Index")

</div>

}

</div>

Details.cshtml:

@model UI.Models.EmpModel

@{

ViewBag.Title = "Details";

}

<h2>Details</h2>

<div>

<h4>EmpModel</h4>

<hr />

<dl class="dl-horizontal">

<dt>

@Html.DisplayNameFor(model => model.EmailId)

</dt>

<dd>

@Html.DisplayFor(model => model.EmailId)

</dd>

<dt>

@Html.DisplayNameFor(model => model.Name)

</dt>

<dd>

@Html.DisplayFor(model => model.Name)

</dd>

<dt>

@Html.DisplayNameFor(model => model.DateOfJoining)

</dt>

<dd>

@Html.DisplayFor(model => model.DateOfJoining)

</dd>

<dt>

@Html.DisplayNameFor(model => model.PassCode)

</dt>

<dd>

@Html.DisplayFor(model => model.PassCode)

</dd>

</dl>

</div>

<p>

@Html.ActionLink("EditEmp", "EditEmp", new { id = Model.PassCode }) |

@Html.ActionLink("Back to List", "Index")

</p>

EditEmp.cshtml:

@model UI.Models.EmpModel

@{

ViewBag.Title = "EditEmp";

}

<h2>EditEmp</h2>

@using (Html.BeginForm())

{

@Html.AntiForgeryToken()

<div class="form-horizontal">

<h4>EmpModel</h4>

<hr />

@Html.ValidationSummary(true, "", new { @class = "text-danger" })

<div class="form-group">

@Html.LabelFor(model => model.EmailId, htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

@Html.EditorFor(model => model.EmailId, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.EmailId, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

@Html.LabelFor(model => model.Name, htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

@Html.EditorFor(model => model.Name, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.Name, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

@Html.LabelFor(model => model.DateOfJoining, htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

@Html.EditorFor(model => model.DateOfJoining, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.DateOfJoining, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

@Html.LabelFor(model => model.PassCode, htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

@Html.EditorFor(model => model.PassCode, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.PassCode, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

<div class="col-md-offset-2 col-md-10">

<input type="submit" value="Save" class="btn btn-default" />

</div>

</div>

</div>

}

<div>

@Html.ActionLink("Back to List", "Index")

</div>

@section Scripts {

@Scripts.Render("~/bundles/jqueryval")

}

Login.cshtml:

@model UI.Models.AdminModel

@{

ViewBag.Title = "Login";

}

<h2>Admin Login Page</h2>

@using (Html.BeginForm())

{

@Html.AntiForgeryToken()

<div class="form-horizontal">

<hr />

@Html.ValidationSummary(true, "", new { @class = "text-danger" })

<div class="form-group">

@Html.LabelFor(model => model.EmailId, htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

@Html.EditorFor(model => model.EmailId, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.EmailId, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

@Html.LabelFor(model => model.Pass, htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

@Html.EditorFor(model => model.Pass, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.Pass, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

<div class="col-md-offset-2 col-md-10">

<input type="submit" value="Login" class="btn btn-primary" />

</div>

</div>

</div>

}

<div>

@Html.ActionLink("Back to List", "Index")

</div>

@section Scripts {

@Scripts.Render("~/bundles/jqueryval")

}

Home:

Index.cshtml:

@model IEnumerable<UI.Models.Blogmodel>

@{

ViewBag.Title = "Index";

}

<h2><center>List of Blogs</center></h2>

<table class="table">

<tr>

<th>

@Html.DisplayNameFor(model => model.Title)

</th>

<th>

@Html.DisplayNameFor(model => model.subject)

</th>

<th>

@Html.DisplayNameFor(model => model.DateOfCreation)

</th>

<th>

@Html.DisplayNameFor(model => model.BlogUrl)

</th>

</tr>

@foreach (var item in Model) {

<tr>

<td>

@Html.DisplayFor(modelItem => item.Title)

</td>

<td>

@Html.DisplayFor(modelItem => item.subject)

</td>

<td>

@Html.DisplayFor(modelItem => item.DateOfCreation)

</td>

<td>

@Html.DisplayFor(modelItem => item.BlogUrl)

</td>

</tr>

}

</table>