**DESCRIPTION - Using Code first Approach Entity Framework**

Basics of C#, MS-SQL Server, ASP.Net MVC, Web API, and NUnit

**A. Data Access Layer (DAL):** This layer will be a type of Class Library. This application will create a database using Entity Framework - Code First Approach on **Azure** and implement Database functionalities such as saving blog details, to get all blogs details, get the blog by ID, update blog details, to delete blog details.

Consider the following Entity Model classes to create a database for the same:

1. Add entity class **AdminInfo** and public properties:

|  |  |
| --- | --- |
| **Property Name** | **Type** |
| EmailId | String |
| Password | String |

1. Add entity class **EmpInfo** and public properties as below:

|  |  |
| --- | --- |
| **Property Name** | **Type** |
| EmailId | String |
| Name | String |
| DateOfJoining | Datetime |
| PassCode | Int |

1. Add entity class **BlogInfo** and public properties as below:

|  |  |
| --- | --- |
| **Property Name** | **Type** |
| BlogId | Int |
| Title | String |
| Subject | String |
| DateOfCreation | Datetime |
| BlogUrl | String |
| EmpEmailId | String |

1. After creating the above entity classes, create the context class by inheriting **DbContext** class
2. 5. Add one more class by inheriting **DropCreateDatabaseIfModelChanges<contextclass>** and default data into AdminInfo table by overriding the SEED method

6. Write database functionalities for EmpInfo entity using **Data Repository Pattern**: saving employee details, to get all employee details, and to validate employees while logging in

7. Write database functionalities for BlogInfo entity using **Data Repository Pattern**: saving blog details, to update blog, to delete a blog, to get all blog details, and to get the blog by ID

**B. App Service Layer:** This layer will be a type of ASP.Net Web Application (Web API).  
This application contains RESTful services to consume functionalities from the DAL class.

1. Add a Web API controller and write action methods to issue a GET and POST and PUT and DELETE request to perform all the functionalities written for EmpInfo and BlogInfo entities inside DAL layer

2. Use attribute-based routing while implementing these functionalities

3. Enable Swagger support for documentation to test this layer

4. Enable CORS (Cross Origin Resource Sharing) policies for all controllers

**C. App UI Layer:** This will be an ASP.Net MVC as a front-end application to implement the following functionalities:

1. Display all blog details on the home page written by all employees

2. Authenticate admin using admin Email ID and password

3. Show page for adding employee details for the admin

4. Authenticate employee using employee EmaiI ID and passcode

5. Show a page for adding blog details to employees

**Note:** While writing views, use bootstrap to design responsive web UI for all above functionalities

**D. DALTest Layer:** This layer is a type of Class Library to test DAL layer functionality using the NUnit and Moq framework.

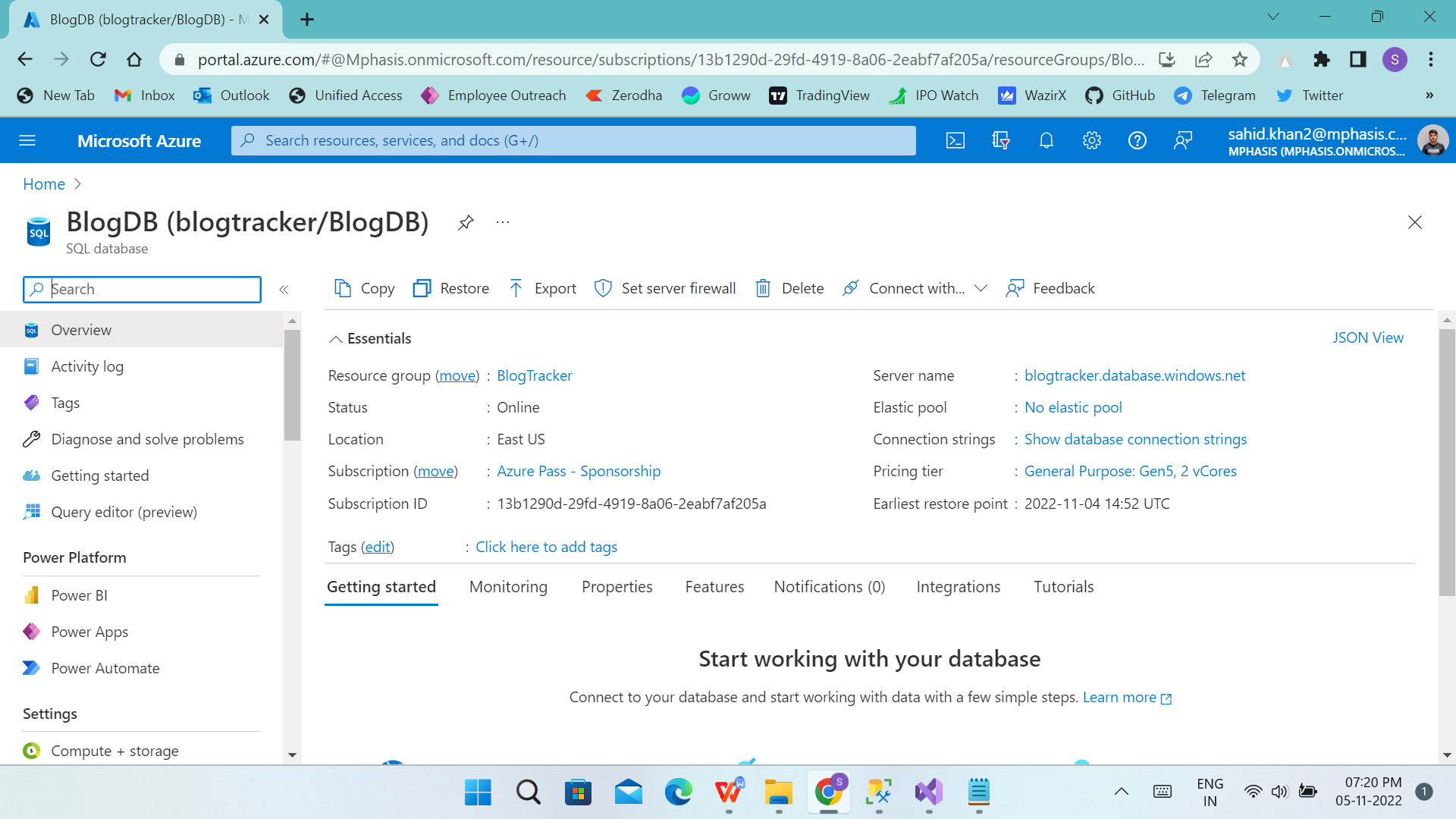
**E. Docker Containerization:** Create Docker Compose using the Docker container orchestration feature through visual studio for AppLayer and Service Layer projects.

****Case Study:****

Simplona Tech Solutions needs a full stack ****Blog Tracker Application (BTA)**** to maintain blog details written by their employees as part of the continuous learning process. The employee must write one blog per week and add blog-related information like the blog title, subject, date of creation, and blog URL in the Blog Tracker Application. You need to create a Full Stack Application that allows employees to add their blog-related details and show all blogs written by employees on the home page.

**Azure SQL DB:**

Created SQL DB and Server in Azure portal.

****

**Visual Studio: Choose asp.net web app(with MVC) - with docker & WebAPI.**

1. **After creating project add 2 Library- DAL and dal.tests.**

**In DAL Lib: Install Entity framework and give connection strings in app.config**

**In BlogDAL.cs**

using System;

using System.Collections.Generic;

using System.ComponentModel.DataAnnotations;

using System.Data.Entity;

namespace BlogDAL

{

public class AdminInfo

{

[Required()]

[Key]

public string EmailID { get; set; }

public string Password { get; set; }

}

public class EmpInfo

{

[Required()]

[Key]

public int Passcode { get; set; }

public string EmailID { get; set; }

public string Name { get; set; }

public DateTime DateOfJoining { get; set; }

}

public class BlogInfo

{

[Required()]

[Key]

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; }

}

public class MyContext : DbContext

{

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

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

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

}

public class AdminInfoDbInitializer : DropCreateDatabaseIfModelChanges<MyContext>

{

protected override void Seed(MyContext context)

{

var depts = new List<AdminInfo> {

new AdminInfo { EmailID = "sahid@gmail.com", Password = "12345"},

new AdminInfo { EmailID = "admin@gmail.com", Password = "12345" },

new AdminInfo { EmailID = "ramy@gmail.com", Password = "12345" },

};

depts.ForEach(s => context.AdminInfos.Add(s));

context.SaveChanges();

}

}

}

**In ValidateAdmin.cs**

using System.Linq;

namespace BlogDAL

{

public class ValidateAdmin

{

MyContext context = null;

public ValidateAdmin()

{

context = new MyContext();

}

public bool Validate(string email, string pass)

{

bool ans = false;

var find = context.AdminInfos.ToList();

foreach (var item in find)

{

if(item.EmailID == email && item.Password == pass)

{

ans = true;

}

}

return ans;

}

}

}

**In EmpInfoDAL.cs**

using System;

using System.Collections.Generic;

using System.Linq;

namespace BlogDAL

{

public class EmpInfoDAL

{

MyContext context = null;

public EmpInfoDAL()

{

context = new MyContext();

}

public bool AddEmp(EmpInfo p)

{

try

{

context.EmpInfos.Add(p);

context.SaveChanges();

return true;

}

catch (Exception ex)

{

return false;

}

}

public bool DeleteEmployeeDetails(int id)

{

try

{

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

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

context.EmpInfos.Remove(r);

context.SaveChanges();

return true;

}

catch (Exception ex)

{

return false;

}

}

public EmpInfo GetEmployeeDetailsByPasscode(int id)

{

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

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

return r;

}

public List<EmpInfo> GetAllEmployeeDetails()

{

return context.EmpInfos.ToList();

}

public bool UpdateEmployeeDetails(int id, EmpInfo p)

{

try

{

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

EmpInfo k = s.Find(pr => pr.Passcode == id);

k.EmailID = p.EmailID;

k.Name = p.Name;

k.DateOfJoining = p.DateOfJoining;

k.Passcode = p.Passcode;

context.SaveChanges();

return true;

}

catch (Exception ex)

{

return false;

}

}

public bool ValidateEmp(string email, int pass)

{

bool ans = false;

var find = context.EmpInfos.ToList();

foreach (var item in find)

{

if (item.EmailID == email && item.Passcode == pass)

{

ans = true;

}

}

return ans;

}

}

}

**In BlogInfoDAL.cs**

using System;

using System.Collections.Generic;

using System.Linq;

namespace BlogDAL

{

public class BlogInfoDAL

{

MyContext context = null;

public BlogInfoDAL()

{

context = new MyContext();

}

public bool AddBlog(BlogInfo p)

{

try

{

context.BlogInfos.Add(p);

context.SaveChanges();

return true;

}

catch (Exception ex)

{

return false;

}

}

public bool DeleteBlogDetails(int id)

{

try

{

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

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

context.BlogInfos.Remove(r);

context.SaveChanges();

return true;

}

catch (Exception ex)

{

return false;

}

}

public BlogInfo GetAllBlogDetailsByBlogID(int id)

{

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

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

return r;

}

public List<BlogInfo> GetAllBlogDetails()

{

return context.BlogInfos.ToList();

}

public bool UpdateBlogDetails(int id, BlogInfo p)

{

try

{

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

BlogInfo k = s.Find(pr => pr.BlogID == id);

k.BlogID = p.BlogID;

k.Title = p.Title;

k.Subject = p.Subject;

k.DateOfCreation = p.DateOfCreation;

k.BlogURL = p.BlogURL;

k.EmpEmailID = p.EmpEmailID;

context.SaveChanges();

return true;

}

catch (Exception ex)

{

return false;

}

}

}

}

**In App.config- Below Entity Framework**

<connectionStrings>

<add connectionString="Server=tcp:blogtracker.database.windows.net,1433;Initial Catalog=BlogDB;Persist Security Info=False;User ID=sahidxb;Password=Hello@123456;MultipleActiveResultSets=False;Encrypt=True;TrustServerCertificate=False;Connection Timeout=30;"

name="MyContext" providerName="System.data.SqlClient">

</add>

</connectionStrings>

**In Main Project - MVC: add ref of DAL and dal.tests to it.**

**In AdminModel.cs**

using System.ComponentModel.DataAnnotations;

namespace BlogTrackerApp.Models

{

public class AdminModel

{

[Key]

[Required()]

public string EmailID { get; set; }

[DataType(DataType.Password)]

public string Password { get; set; }

}

}

**In EmpModel.cs**

namespace BlogTrackerApp.Models

{

public class EmpModel

{

[Required()]

[Key]

[DataType(DataType.Password)]

public int Passcode { get; set; }

public string EmailID { get; set; }

public string Name { get; set; }

[DisplayFormat(DataFormatString = "{0:yyyy-MM-dd}", ApplyFormatInEditMode = true)]

[DataType(DataType.Date)]

public DateTime DateOfJoining { get; set; }

}

}

**In BlogModel.cs**

using System;

using System.ComponentModel.DataAnnotations;

namespace BlogTrackerApp.Models

{

public class BlogModel

{

[Required()]

[Key]

public int BlogID { get; set; }

public string Title { get; set; }

public string Subject { get; set; }

[DisplayFormat(DataFormatString = "{0:yyyy-MM-dd}", ApplyFormatInEditMode = true)]

[DataType(DataType.Date)]

public DateTime DateOfCreation { get; set; }

public string BlogURL { get; set; }

public string EmpEmailID { get; set; }

}

}

**In AdminController.cs**

using System.Web.Mvc;

using BlogDAL;

namespace BlogTrackerApp.Controllers

{

public class AdminController : Controller

{

ValidateAdmin ad = null;

public AdminController()

{

ad = new ValidateAdmin();

}

// GET: Admin

public ActionResult AdminLogin()

{

return View();

}

[HttpPost]

public ActionResult AdminLogin(FormCollection collection)

{

bool valid = ad.Validate(Request["EmailID"].ToString(), Request["Password"].ToString());

if (valid)

{

//return RedirectToAction("actionName", "controllerName", null);

return RedirectToAction("EmpPage", "Emp", null);

}

else

{

ViewBag.Message = "Email or Password is wrong...";

return RedirectToAction("AdminLogin");

}

}

}

}

**In EmpController.cs**

using System;

using System.Collections.Generic;

using System.Web.Mvc;

using BlogDAL;

using BlogTrackerApp.Models;

namespace BlogTrackerApp.Controllers

{

public class EmpController : Controller

{

EmpInfoDAL helper = null;

public EmpController()

{

helper = new EmpInfoDAL();

}

// GET: Emp

public ActionResult EmpPage()

{

var emplist = helper.GetAllEmployeeDetails();

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

foreach(var item in emplist)

{

m.Add(new EmpModel { Passcode = item.Passcode, EmailID = item.EmailID, Name = item.Name , DateOfJoining = item.DateOfJoining});

}

return View(m);

}

// GET: Emp/Details/5

public ActionResult EmpDetails(int id)

{

var data = helper.GetEmployeeDetailsByPasscode(id);

EmpModel emp = new EmpModel();

emp.Passcode = id;

emp.EmailID = data.EmailID;

emp.Name= data.Name;

emp.DateOfJoining = data.DateOfJoining;

return View(emp);

}

public ActionResult AddEmployee()

{

return View();

}

// GET: Emp/Create

[HttpPost]

public ActionResult AddEmployee(FormCollection collection)

{

try

{

// TODO: Add insert logic here

EmpInfo bal = new EmpInfo();

bal.Passcode = Convert.ToInt32(Request["Passcode"]);

bal.EmailID = Request["EmailID"].ToString();

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

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

bool ans = helper.AddEmp(bal);

if (ans)

{

return RedirectToAction("EmpPage");

}

else

{

return View();

}

}

catch (Exception ex)

{

ViewBag.exMsg = ex.Message;

return View();

}

}

// GET: Emp/Edit/5

public ActionResult EditEmp(int id)

{

var emp = helper.GetEmployeeDetailsByPasscode(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 = helper.GetEmployeeDetailsByPasscode(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 = helper.UpdateEmployeeDetails(id,emp);

if (ans)

{

return RedirectToAction("EmpPage");

}

else

{

return View();

}

}

catch

{

return View();

}

}

// GET: Emp/Delete/5

public ActionResult DeleteEmp(int id)

{

var emp = helper.GetEmployeeDetailsByPasscode(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/Delete/5

[HttpPost]

public ActionResult DeleteEmp(int id, FormCollection collection)

{

try

{

// TODO: Add delete logic here

var dataFound = helper.GetEmployeeDetailsByPasscode(id);

if (dataFound != null)

{

bool ans = helper.DeleteEmployeeDetails(id);

if (ans)

{

return RedirectToAction("EmpPage");

}

else

{

return View();

}

}

return RedirectToAction("EmpPage");

}

catch

{

return View();

}

}

public ActionResult EmpLogin()

{

return View();

}

[HttpPost]

public ActionResult EmpLogin(FormCollection collection)

{

bool valid = helper.ValidateEmp(Request["EmailID"].ToString(), Convert.ToInt32(Request["Passcode"]));

if (valid)

{

//return RedirectToAction("actionName", "controllerName", null);

return RedirectToAction("BlogPage", "Blog", null);

}

else

{

ViewBag.Message = "Email or Passcode is wrong...";

return RedirectToAction("EmpLogin");

}

}

}

}

**In BlogController.cs**

using System;

using System.Collections.Generic;

using System.Web.Mvc;

using BlogDAL;

using BlogTrackerApp.Models;

namespace BlogTrackerApp.Controllers

{

public class BlogController : Controller

{

BlogInfoDAL helper = null;

public BlogController()

{

helper = new BlogInfoDAL();

}

// GET: Emp

public ActionResult BlogPage()

{

var list = helper.GetAllBlogDetails();

List<BlogModel> m = new List<BlogModel>();

foreach (var item in list)

{

m.Add(new BlogModel { BlogID = item.BlogID, Title = item.Title, Subject = item.Subject, DateOfCreation = item.DateOfCreation, BlogURL = item.BlogURL, EmpEmailID = item.EmpEmailID });

}

return View(m);

}

public ActionResult BlogHomePage()

{

var list = helper.GetAllBlogDetails();

List<BlogModel> m = new List<BlogModel>();

foreach (var item in list)

{

m.Add(new BlogModel { Title = item.Title, Subject = item.Subject, DateOfCreation = item.DateOfCreation, BlogURL = item.BlogURL, EmpEmailID = item.EmpEmailID });

}

return View(m);

}

// GET: Emp/Details/5

public ActionResult BlogDetails(int id)

{

var data = helper.GetAllBlogDetailsByBlogID(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);

}

public ActionResult AddaBlog()

{

return View();

}

// GET: Emp/Create

[HttpPost]

public ActionResult AddaBlog(FormCollection collection)

{

try

{

// TODO: Add insert logic here

BlogInfo bal = new BlogInfo();

bal.BlogID = Convert.ToInt32(Request["BlogID"]);

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

bal.Subject = Request["Subject"].ToString();

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

bal.BlogURL = Request["BlogURL"].ToString();

bal.EmpEmailID = Request["EmpEmailID"].ToString();

bool ans = helper.AddBlog(bal);

if (ans)

{

return RedirectToAction("BlogPage");

}

else

{

return View();

}

}

catch (Exception ex)

{

ViewBag.exMsg = ex.Message;

return View();

}

}

// GET: Emp/Edit/5

public ActionResult EditBlog(int id)

{

var emp = helper.GetAllBlogDetailsByBlogID(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 = helper.GetAllBlogDetailsByBlogID(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 = helper.UpdateBlogDetails(id, emp);

if (ans)

{

return RedirectToAction("BlogPage");

}

else

{

return View();

}

}

catch

{

return View();

}

}

// GET: Emp/Delete/5

public ActionResult DeleteBlog(int id)

{

var emp = helper.GetAllBlogDetailsByBlogID(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 = helper.GetAllBlogDetailsByBlogID(id);

if (dataFound != null)

{

bool ans = helper.DeleteBlogDetails(id);

if (ans)

{

return RedirectToAction("BlogPage");

}

else

{

return View();

}

}

return RedirectToAction("BlogPage");

}

catch

{

return View();

}

}

}

}

**In EmpAPIController.cs**

using System;

using System.Collections.Generic;

using System.Net;

using System.Net.Http;

using System.Web.Http;

using BlogDAL;

using BlogTrackerApp.Models;

namespace BlogTrackerApp.Controllers

{

public class EmpAPIController : ApiController

{

// GET api/<controller>

EmpInfoDAL ms = null;

public EmpAPIController()

{

ms = new EmpInfoDAL();

}

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

[Route("GetAllEmployeeDetails")]

public IEnumerable<EmpModel> Get()

{

List<EmpInfo> c = ms.GetAllEmployeeDetails();

foreach (var item in c)

{

EmpModel v = new EmpModel();

v.EmailID = item.EmailID;

v.Name = item.Name;

v.DateOfJoining = item.DateOfJoining;

v.Passcode = item.Passcode;

s.Add(v);

}

return s;

}

// GET api/<controller>/5

[Route("GetEmployeeDetailsByPasscode/{id}")]

public EmpModel Get(int id)

{

EmpModel r = new EmpModel();

EmpInfo p = new EmpInfo();

p = ms.GetEmployeeDetailsByPasscode(id);

r.EmailID = p.EmailID.ToString();

r.Name = p.Name.ToString();

r.DateOfJoining = Convert.ToDateTime(p.DateOfJoining);

r.Passcode = Convert.ToInt32(p.Passcode);

return r;

}

// POST api/<controller>

[Route("AddEmployeeDetails")]

public HttpResponseMessage Post([FromBody] EmpModel value)

{

EmpInfo r = new EmpInfo();

r.EmailID = value.EmailID;

r.Name = value.Name;

r.DateOfJoining = value.DateOfJoining;

r.Passcode = value.Passcode;

bool k = ms.AddEmp(r);

if (k)

{

return Request.CreateResponse(HttpStatusCode.OK);

}

else

{

return Request.CreateResponse(HttpStatusCode.NotAcceptable);

}

}

// PUT api/<controller>/5

[Route("UpdateEmployeeDetails/{id}")]

public HttpResponseMessage Put(int id, [FromBody] EmpModel value)

{

EmpInfo r = new EmpInfo();

r.EmailID = value.EmailID;

r.Name = value.Name;

r.DateOfJoining = value.DateOfJoining;

r.Passcode = value.Passcode;

bool k = ms.UpdateEmployeeDetails(id, r);

if (k)

{

return Request.CreateResponse(HttpStatusCode.OK);

}

else

{

return Request.CreateResponse(HttpStatusCode.NotAcceptable);

}

}

// DELETE api/<controller>/5

[Route("DeleteEmployeeDetails/{id}")]

public HttpResponseMessage Delete(int id)

{

bool k = ms.DeleteEmployeeDetails(id);

if (k)

{

return Request.CreateResponse(HttpStatusCode.OK);

}

else

{

return Request.CreateResponse(HttpStatusCode.NotAcceptable);

}

}

}

}

**In BlogAPIController.cs**

using BlogDAL;

using BlogTrackerApp.Models;

using System;

using System.Collections.Generic;

using System.Net;

using System.Net.Http;

using System.Web.Http;

namespace BlogTrackerApp.Controllers

{

public class BlogAPIController : ApiController

{

// GET api/<controller>

BlogInfoDAL ms = null;

public BlogAPIController()

{

ms = new BlogInfoDAL();

}

List<BlogModel> s = new List<BlogModel>();

[Route("GetAllBlogDetails")]

public IEnumerable<BlogModel> Get()

{

List<BlogInfo> c = ms.GetAllBlogDetails();

foreach (var item in c)

{

BlogModel v = new BlogModel();

v.BlogID = item.BlogID;

v.Title = item.Title;

v.Subject = item.Subject;

v.DateOfCreation = item.DateOfCreation;

v.BlogURL = item.BlogURL;

v.EmpEmailID = item.EmpEmailID;

s.Add(v);

}

return s;

}

// GET api/<controller>/5

[Route("GetAllBlogDetailsByBlogID/{id}")]

public BlogModel Get(int id)

{

BlogModel r = new BlogModel();

BlogInfo p = new BlogInfo();

p = ms.GetAllBlogDetailsByBlogID(id);

r.BlogID = Convert.ToInt32(p.BlogID);

r.Title = p.Title.ToString();

r.Subject = p.Subject.ToString();

r.DateOfCreation= Convert.ToDateTime(p.DateOfCreation);

r.BlogURL = p.BlogURL.ToString();

r.EmpEmailID = p.EmpEmailID.ToString();

return r;

}

// POST api/<controller>

[Route("AddBlogDetails")]

public HttpResponseMessage Post([FromBody] BlogModel value)

{

BlogInfo r = new BlogInfo();

r.BlogID = value.BlogID;

r.Title = value.Title;

r.Subject = value.Subject;

r.DateOfCreation = value.DateOfCreation;

r.BlogURL = value.BlogURL;

r.EmpEmailID = value.EmpEmailID;

bool k = ms.AddBlog(r);

if (k)

{

return Request.CreateResponse(HttpStatusCode.OK);

}

else

{

return Request.CreateResponse(HttpStatusCode.NotAcceptable);

}

}

// PUT api/<controller>/5

[Route("UpdateBlogDetails/{id}")]

public HttpResponseMessage Put(int id, [FromBody] BlogModel value)

{

BlogInfo r = new BlogInfo();

r.BlogID = value.BlogID;

r.Title = value.Title;

r.Subject = value.Subject;

r.DateOfCreation = value.DateOfCreation;

r.BlogURL = value.BlogURL;

r.EmpEmailID = value.EmpEmailID;

bool k = ms.UpdateBlogDetails(id, r);

if (k)

{

return Request.CreateResponse(HttpStatusCode.OK);

}

else

{

return Request.CreateResponse(HttpStatusCode.NotAcceptable);

}

}

// DELETE api/<controller>/5

[Route("DeleteBlogDetails/{id}")]

public HttpResponseMessage Delete(int id)

{

bool k = ms.DeleteBlogDetails(id);

if (k)

{

return Request.CreateResponse(HttpStatusCode.OK);

}

else

{

return Request.CreateResponse(HttpStatusCode.NotAcceptable);

}

}

}

}

**In RouteConfig.cs** - in App\_Start

public static void RegisterRoutes(RouteCollection routes)

{

routes.IgnoreRoute("{resource}.axd/{\*pathInfo}");

routes.MapRoute(

name: "Default",

url: "{controller}/{action}/{id}",

defaults: new { controller = "Blog", action = "BlogHomePage", id = UrlParameter.Optional }

);

}

**In \_Layout.cshtml - In Views -> Shared folder**

<!DOCTYPE html>

<html>

<head>

<meta charset="utf-8" />

<meta name="viewport" content="width=device-width" />

<title>@ViewBag.Title</title>

@Styles.Render("~/Content/css")

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

</head>

<body>

<div class="navbar navbar-inverse navbar-fixed-top">

<div class="container">

<div class="navbar-header">

<button type="button" class="navbar-toggle" data-toggle="collapse" data-target=".navbar-collapse" title="more options">

<span class="icon-bar"></span>

<span class="icon-bar"></span>

<span class="icon-bar"></span>

</button>

@Html.ActionLink("Blog Tracker", "BlogHomePage", "Blog", new { area = "" }, new { @class = "navbar-brand" })

</div>

<div class="navbar-collapse collapse">

<ul class="nav navbar-nav">

<li>@Html.ActionLink("Home", "BlogHomePage", "Blog", new { area = "" }, null)</li>

<li>@Html.ActionLink("Admin Login", "AdminLogin", "Admin", new { area = "" }, null)</li>

<li>@Html.ActionLink("Emp Login", "EmpLogin", "Emp", new { area = "" }, null)</li>

@\*<li>@Html.ActionLink("Log out", "BlogHomePage", "Blog", new { area = "" }, null)</li>\*@

</ul>

</div>

</div>

</div>

<div class="container body-content">

@RenderBody()

<hr />

<footer>

<p>&copy; @DateTime.Now.Year - Blog Tracker Application</p>

</footer>

</div>

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

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

@RenderSection("scripts", required: false)

</body>

</html>

**In Web.Config- Below Entity Framework**

<connectionStrings>

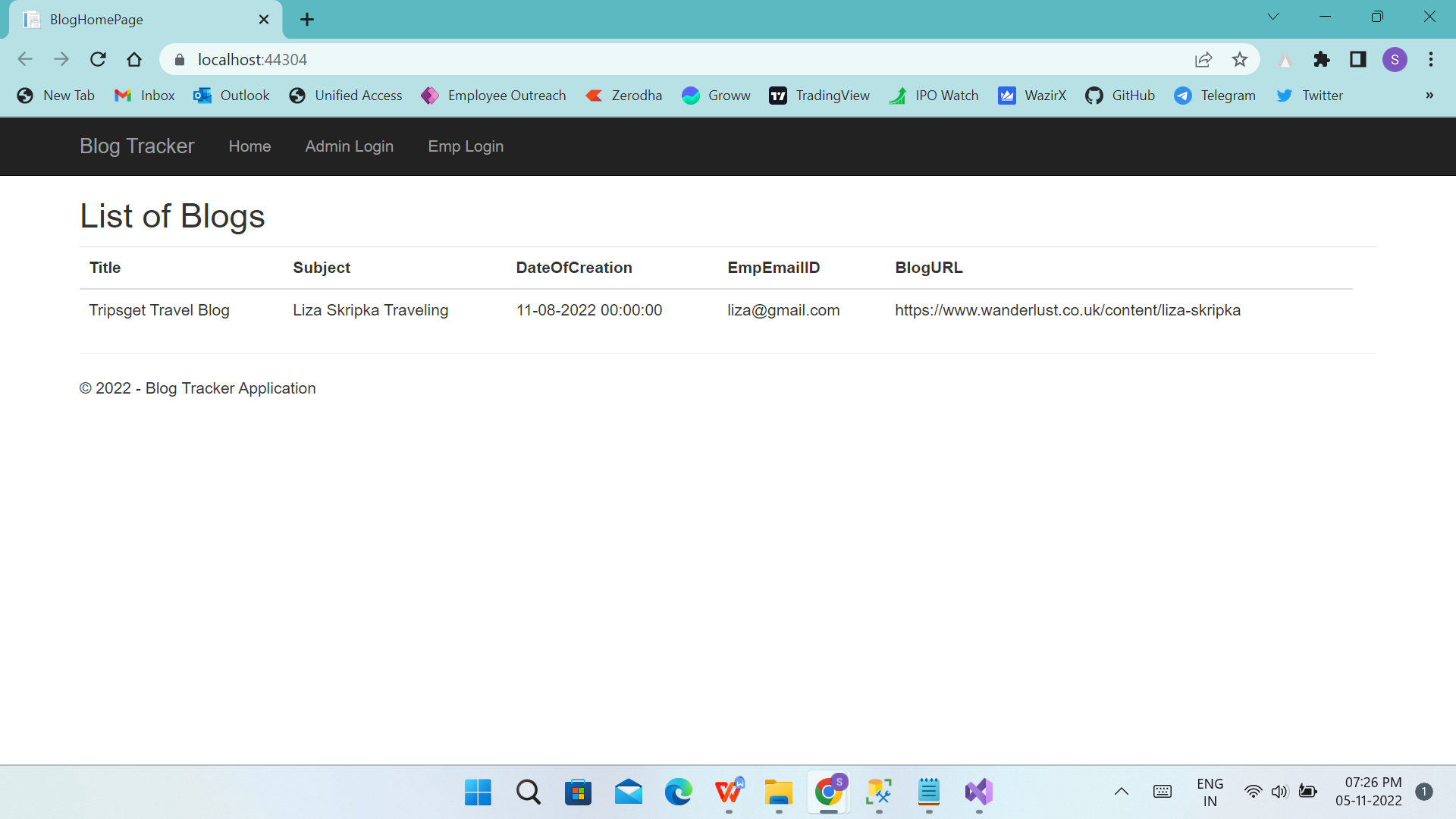
<add connectionString="Server=tcp:blogtracker.database.windows.net,1433;Initial Catalog=BlogDB;Persist Security Info=False;User ID=sahidxb;Password=Hello@123456;MultipleActiveResultSets=False;Encrypt=True;TrustServerCertificate=False;Connection Timeout=30;"

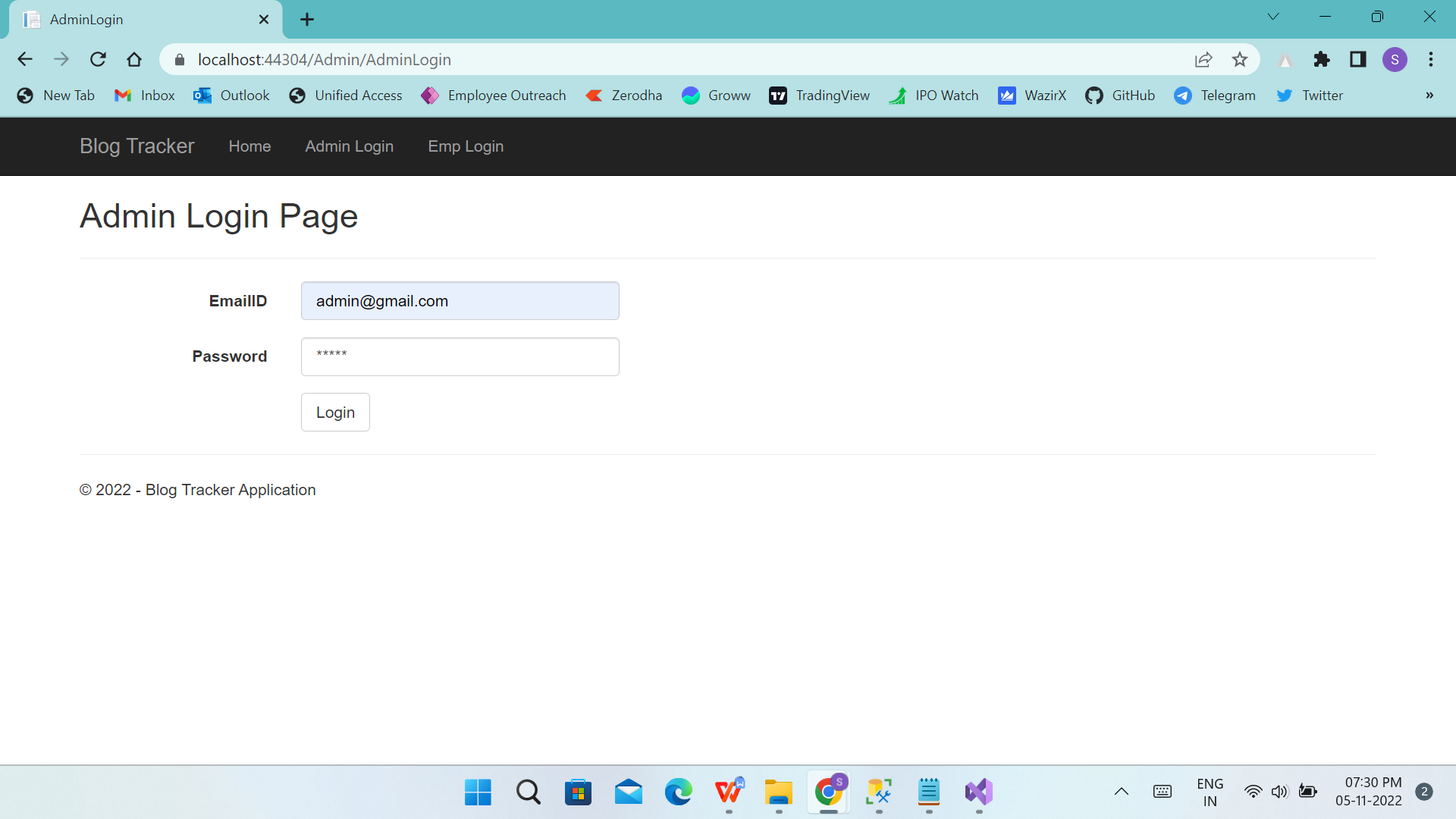
name="MyContext" providerName="System.data.SqlClient">

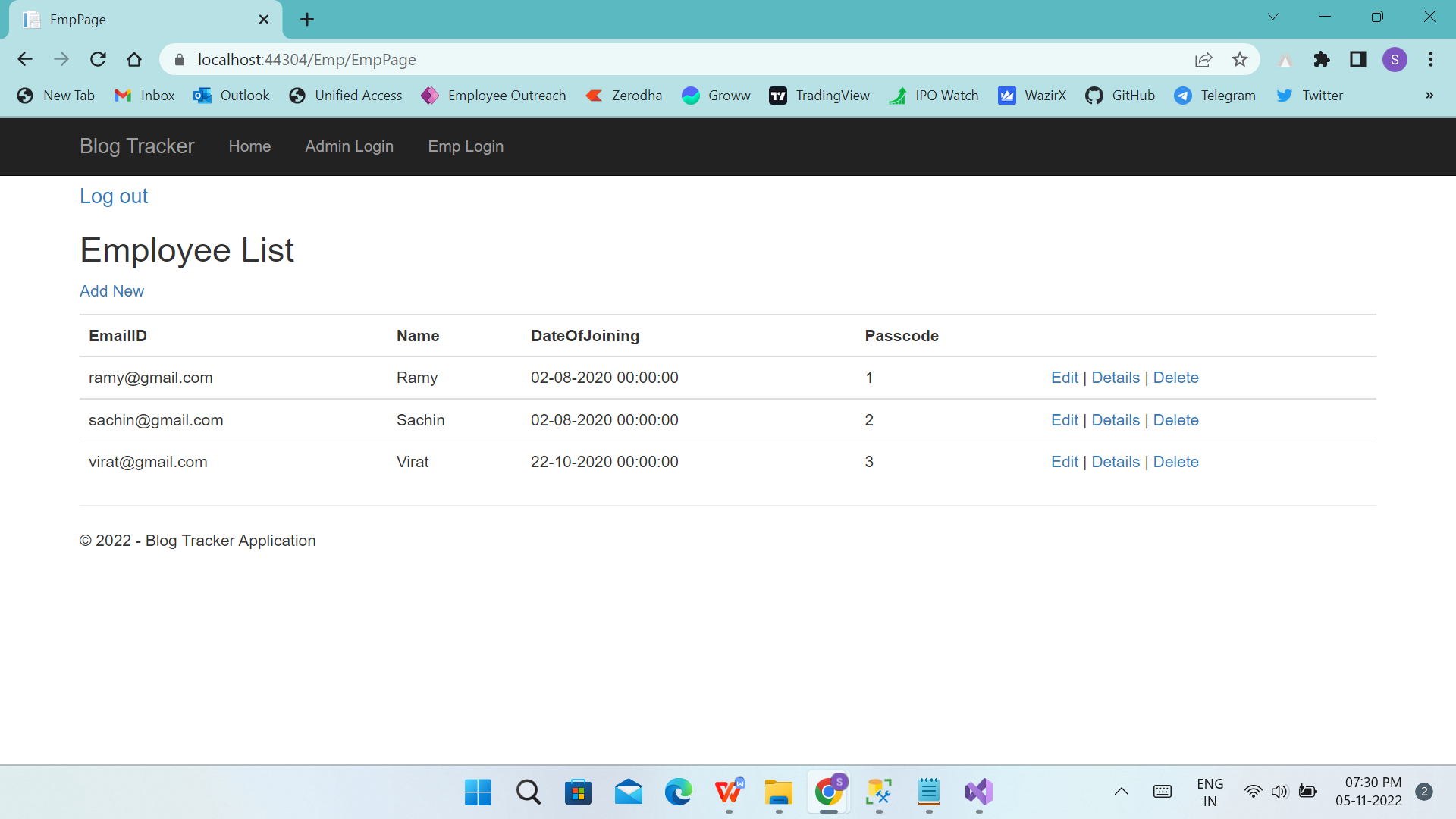
</add>

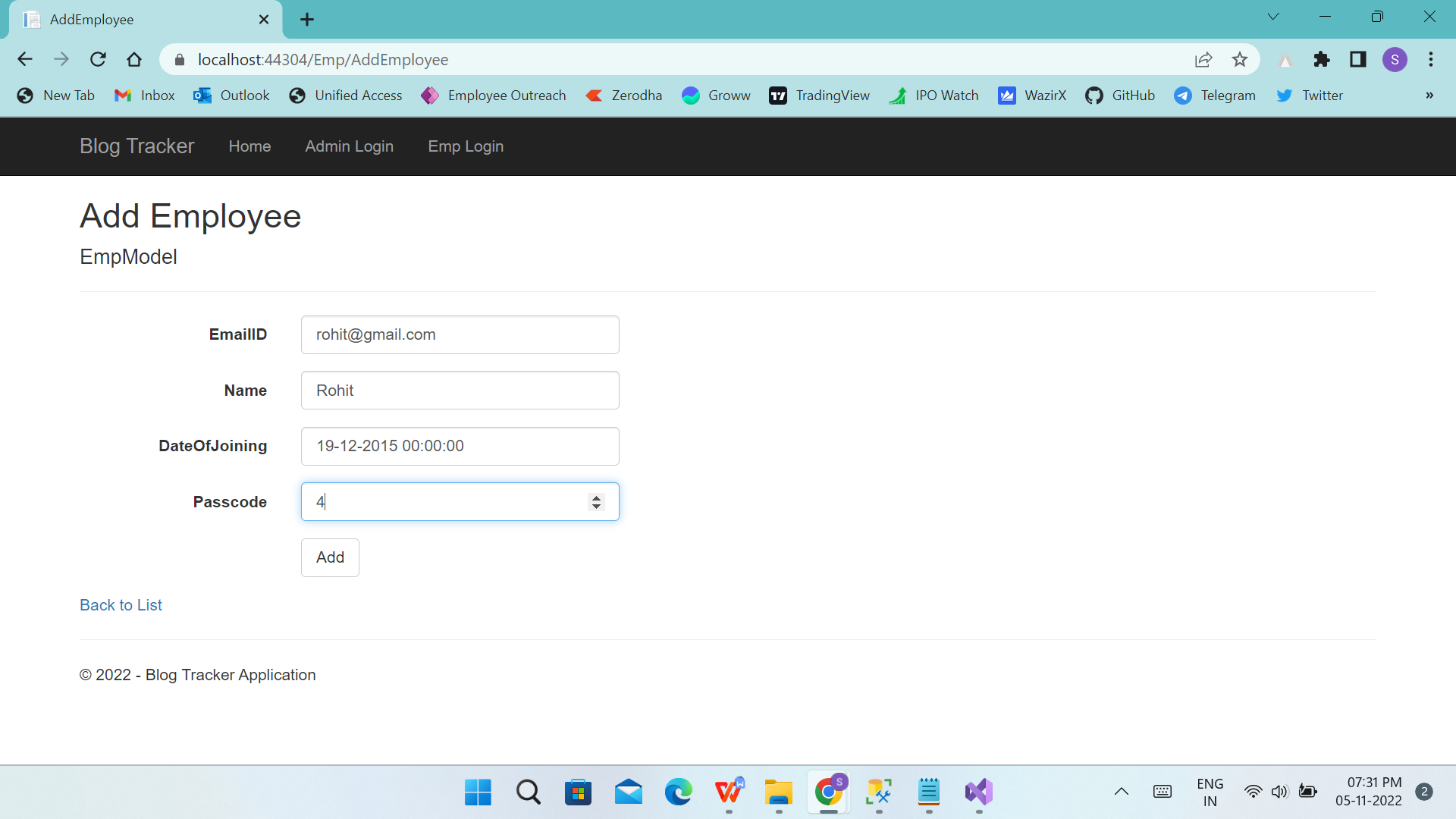
</connectionStrings>

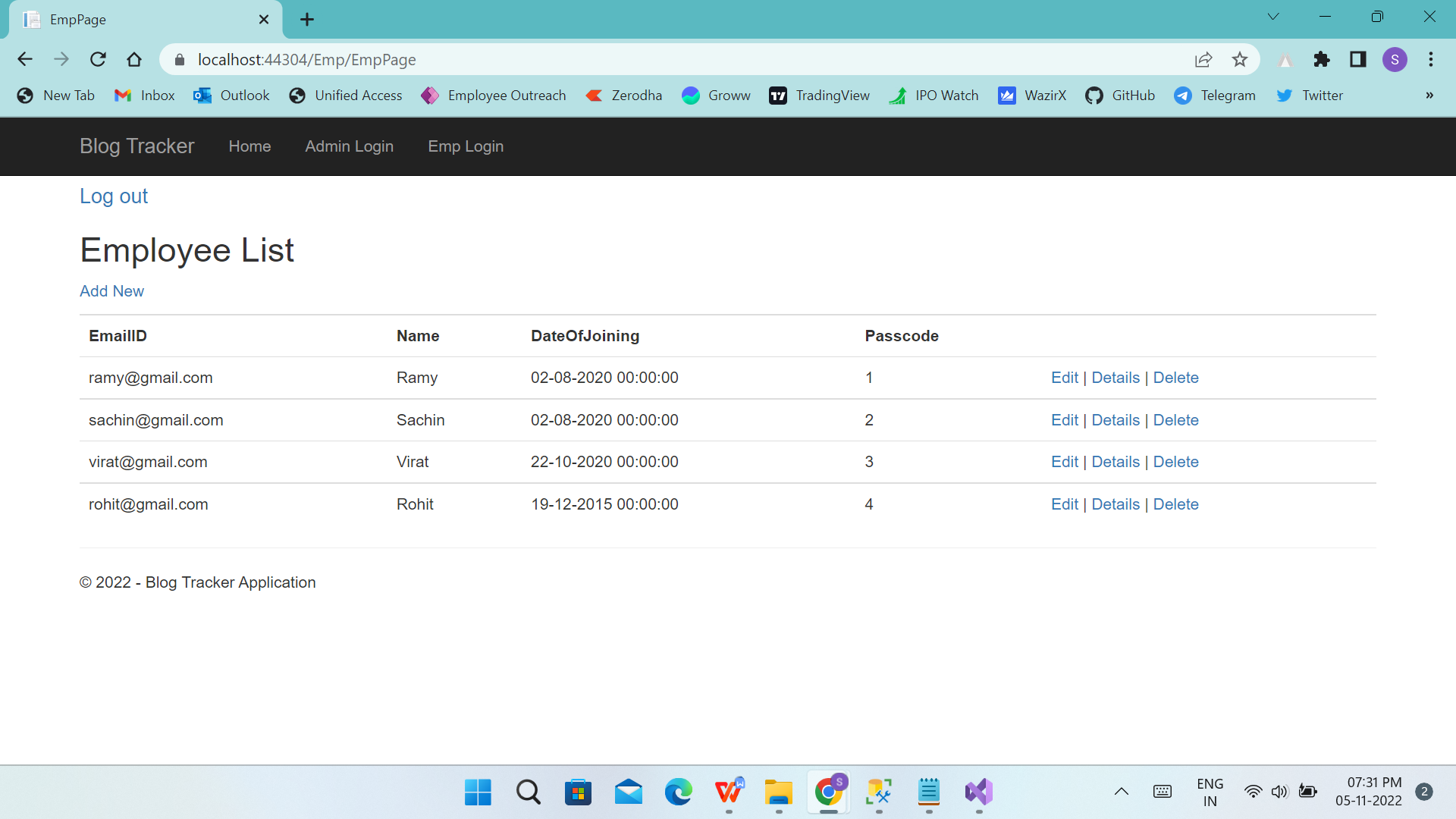
**Some Output Examples -**

****

****

****

****

****