**Hands-On: Stage 4 - Web API – Tools, Attributes - Day 75 – Handsons**

**Handson - 1**

1. **JsonWebToken**

**AuthController.cs**

using Microsoft.AspNetCore.Http;

using Microsoft.AspNetCore.Mvc;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Threading.Tasks;

using Microsoft.IdentityModel.Tokens;

using System.Security.Claims;

using System.IdentityModel.Tokens.Jwt;

using System.Text;

using Microsoft.AspNetCore.Authorization;

using WebApiFourth.Models;

// For more information on enabling Web API for empty projects, visit https://go.microsoft.com/fwlink/?LinkID=397860

namespace WebApiFourth.Controllers

{

    [AllowAnonymous]

    [Route("api/[controller]")]

    [ApiController]

    public class AuthController : ControllerBase

    {

        [HttpPost]

        public IActionResult Post([FromBody] UserModel user)

        {

            string userRole = "Admin";

            string tokenStr = GenerateJSONWebToken(user.UserName, "Admin");  // Token generated for Admin Role

            return new OkObjectResult(new { token = tokenStr });

        }

        private string GenerateJSONWebToken(string userId, string userRole)

        {

            var securityKey = new SymmetricSecurityKey(Encoding.UTF8.GetBytes("mysuperdupersecret"));

            var credentials = new SigningCredentials(securityKey, SecurityAlgorithms.HmacSha256);

            var claims = new List<Claim>

            {

                new Claim(ClaimTypes.Role, userRole),

                new Claim("UserId", userId)

            };

            var token = new JwtSecurityToken(

                        issuer: "mySystem",

                        audience: "myUsers",

                        claims: claims,

                        expires: DateTime.Now.AddMinutes(10),

                        signingCredentials: credentials);

            return new JwtSecurityTokenHandler().WriteToken(token);

        }

    }

}

**EmployeeController.cs**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Threading.Tasks;

using Microsoft.AspNetCore.Authorization;

using Microsoft.AspNetCore.Http;

using Microsoft.AspNetCore.Mvc;

using WebApiCorsJwtToken.Models;

namespace WebApiCoreActionExcptionFilter.Controllers

{

    [Route("api/[controller]")]

    [ApiController]

    [Authorize(Roles = "POC,Admin")]

    public class EmployeeController : ControllerBase

    {

        private Employee[] employees = new Employee[]

     {

          new Employee{Id=1,Name="John",Salary=50000,Permanent=true,Department=new Department{Id=1,Name="Finance"},Skills=new List<Skill>{new Skill{Id=1,Name="Communication"},new Skill { Id = 2, Name = "Translator" } },DateOfBirth=DateTime.Parse("02/12/1994") },

          new Employee{Id=2,Name="Paul",Salary=40000,Permanent=true,Department=new Department{Id=2,Name="IT"},Skills=new List<Skill>{new Skill{Id=1,Name="Communication"},new Skill { Id = 2, Name = "Developer" } },DateOfBirth=DateTime.Parse("02/12/1996") }

     };

        private IEnumerable<Employee> GetStandardEmployeeList()

        {

            return employees;

        }

        // GET: api/Employee

        [HttpGet]

        [ProducesResponseType(StatusCodes.Status200OK)]

        public IActionResult Get()

        {

            return Ok(GetStandardEmployeeList());

        }

        // GET: api/Employee/5

        [HttpGet("{id}", Name = "Get")]

        public string Get(int id)

        {

            return "value";

        }

        // POST: api/Employee

        [HttpPost]

        public void Post([FromBody] string value)

        {

        }

        // PUT: api/Employee/5

        [HttpPut("{id}")]

        public void Put(int id, [FromBody] string value)

        {

        }

        // DELETE: api/ApiWithActions/5

        [HttpDelete("{id}")]

        public void Delete(int id)

        {

        }

    }

}

**WeatherForecastController.cs**

using Microsoft.AspNetCore.Mvc;

using Microsoft.Extensions.Logging;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Threading.Tasks;

namespace WebApiFourth.Controllers

{

    [ApiController]

    [Route("[controller]")]

    public class WeatherForecastController : ControllerBase

    {

        private static readonly string[] Summaries = new[]

        {

            "Freezing", "Bracing", "Chilly", "Cool", "Mild", "Warm", "Balmy", "Hot", "Sweltering", "Scorching"

        };

        private readonly ILogger<WeatherForecastController> \_logger;

        public WeatherForecastController(ILogger<WeatherForecastController> logger)

        {

            \_logger = logger;

        }

        [HttpGet]

        public IEnumerable<WeatherForecast> Get()

        {

            var rng = new Random();

            return Enumerable.Range(1, 5).Select(index => new WeatherForecast

            {

                Date = DateTime.Now.AddDays(index),

                TemperatureC = rng.Next(-20, 55),

                Summary = Summaries[rng.Next(Summaries.Length)]

            })

            .ToArray();

        }

    }

}

**Department.cs**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Threading.Tasks;

namespace WebApiCorsJwtToken.Models

{

    public class Department

    {

        public int Id { get; set; }

        public string Name { get; set; }

    }

}

**Employee.cs**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Threading.Tasks;

namespace WebApiCorsJwtToken.Models

{

    public class Employee

    {

        public int Id { get; set; }

        public string Name { get; set; }

        public int Salary { get; set; }

        public bool Permanent { get; set; }

        public Department Department { get; set; }

        public List<Skill> Skills { get; set; }

        public DateTime DateOfBirth { get; set; }

    }

}

**SchoolDbContext.cs**

using Microsoft.EntityFrameworkCore;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Threading.Tasks;

using WebApiFourth;

namespace WebApiFourth.Models

{

    public class SchoolDbContext: DbContext

    {

        public virtual DbSet<UserModel> UserModels { get; set; }

    }

}

**UserModel.cs**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Threading.Tasks;

namespace WebApiFourth.Models

{

    public class UserModel

    {

        public string UserName { get; set; }

    }

}

**Skill.cs**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Threading.Tasks;

namespace WebApiCorsJwtToken.Models

{

    public class Skill

    {

        public int Id { get; set; }

        public string Name { get; set; }

    }

}

**Startup.cs**

using Microsoft.AspNetCore.Builder;

using Microsoft.AspNetCore.Hosting;

using Microsoft.Extensions.Configuration;

using Microsoft.Extensions.DependencyInjection;

using Microsoft.Extensions.Hosting;

using Microsoft.OpenApi.Models;

using Microsoft.EntityFrameworkCore;

using Microsoft.IdentityModel.Tokens;       // for SymmetricSecurityKey, TokenValidationParameters

using System.Text;      //  for Encoding

using Microsoft.AspNetCore.Authentication.JwtBearer;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Threading.Tasks;

using WebApiFourth.Models;

namespace WebApiFourth

{

    public class Startup

    {

        public Startup(IConfiguration configuration)

        {

            Configuration = configuration;

        }

        public IConfiguration Configuration { get; }

        // This method gets called by the runtime. Use this method to add services to the container.

        public void ConfigureServices(IServiceCollection services)

        {

            services.AddControllers();

            services.AddSwaggerGen(c =>

            {

                c.SwaggerDoc("v1", new OpenApiInfo { Title = "WebApiFourth", Version = "v1" });

            });

            //var connection = Configuration.GetConnectionString("SchoolDbConnection");

            //services.AddDbContext<SchoolDbContext>(o => o.UseSqlServer(""));

            // Jwt Authentication Settings

            string securityKey = "mysuperdupersecret";

            var symmetricSecurityKey = new SymmetricSecurityKey(Encoding.UTF8.GetBytes(securityKey));

            services.AddAuthentication(x =>

            {

                x.DefaultAuthenticateScheme = JwtBearerDefaults.AuthenticationScheme;

                x.DefaultChallengeScheme = JwtBearerDefaults.AuthenticationScheme;

                x.DefaultSignInScheme = JwtBearerDefaults.AuthenticationScheme;

            })

            .AddJwtBearer(JwtBearerDefaults.AuthenticationScheme, x =>

            {

                x.TokenValidationParameters = new TokenValidationParameters

                {

                    //what to validate

                    ValidateIssuer = true,

                    ValidateAudience = true,

                    ValidateLifetime = true,

                    ValidateIssuerSigningKey = true,

                    //setup validate data

                    ValidIssuer = "mySystem",

                    ValidAudience = "myUsers",

                    IssuerSigningKey = symmetricSecurityKey

                };

            });

        }

        // This method gets called by the runtime. Use this method to configure the HTTP request pipeline.

        public void Configure(IApplicationBuilder app, IWebHostEnvironment env)

        {

            if (env.IsDevelopment())

            {

                app.UseDeveloperExceptionPage();

                app.UseSwagger();

                app.UseSwaggerUI(c => c.SwaggerEndpoint("/swagger/v1/swagger.json", "WebApiFourth v1"));

            }

            app.UseHttpsRedirection();

            app.UseRouting();

            app.UseAuthorization();

            app.UseEndpoints(endpoints =>

            {

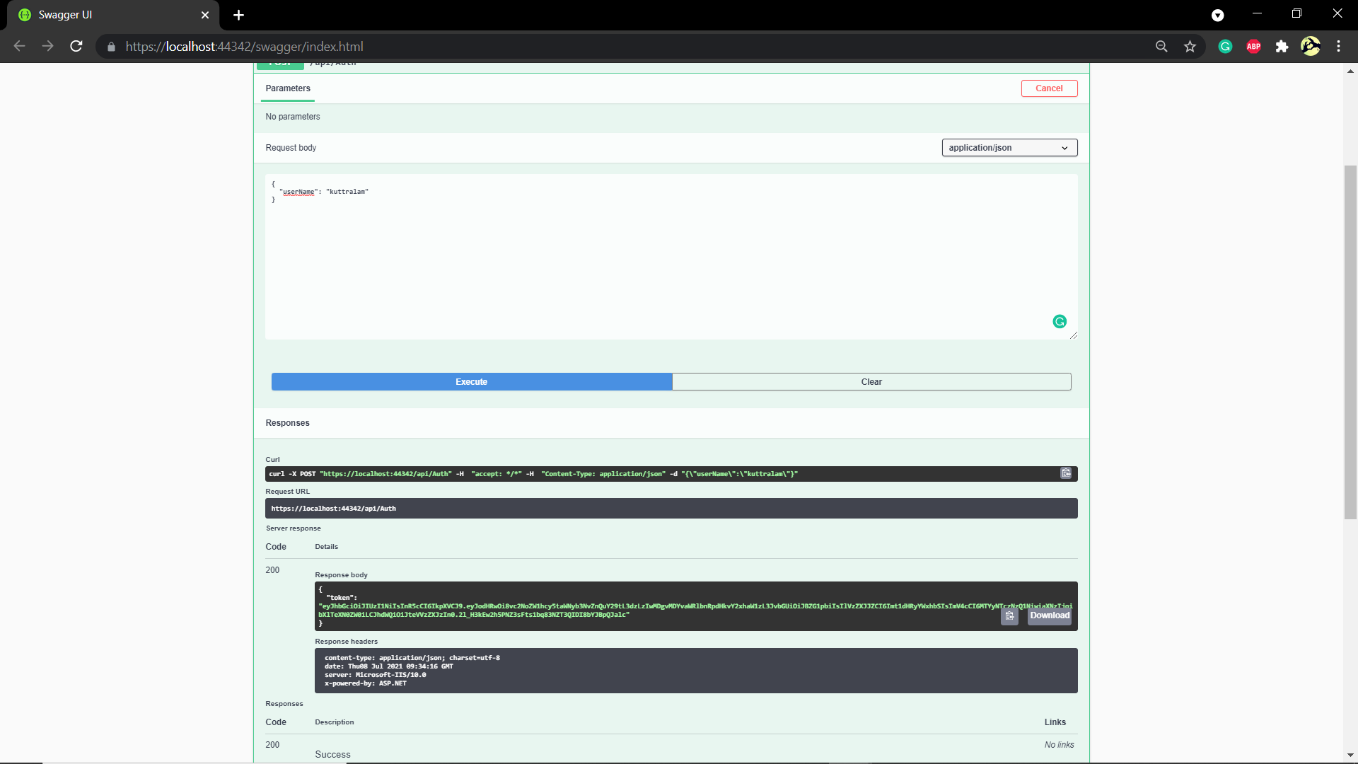
                endpoints.MapControllers();

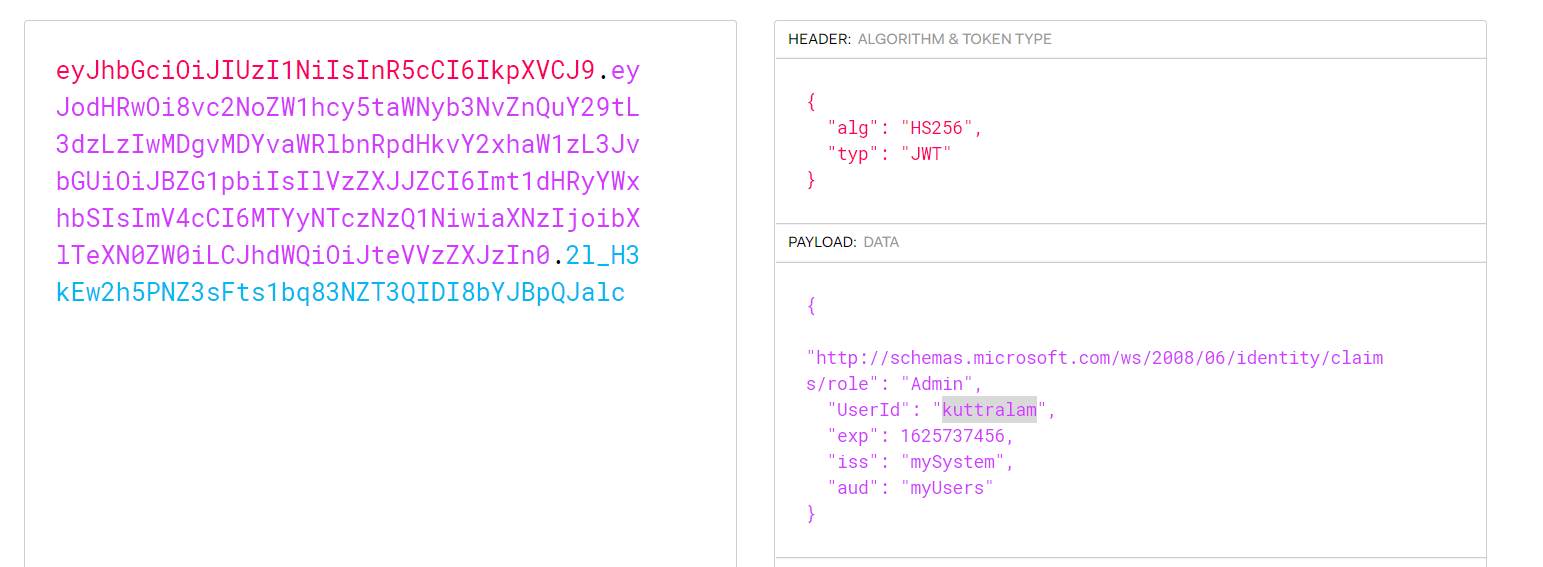
            });

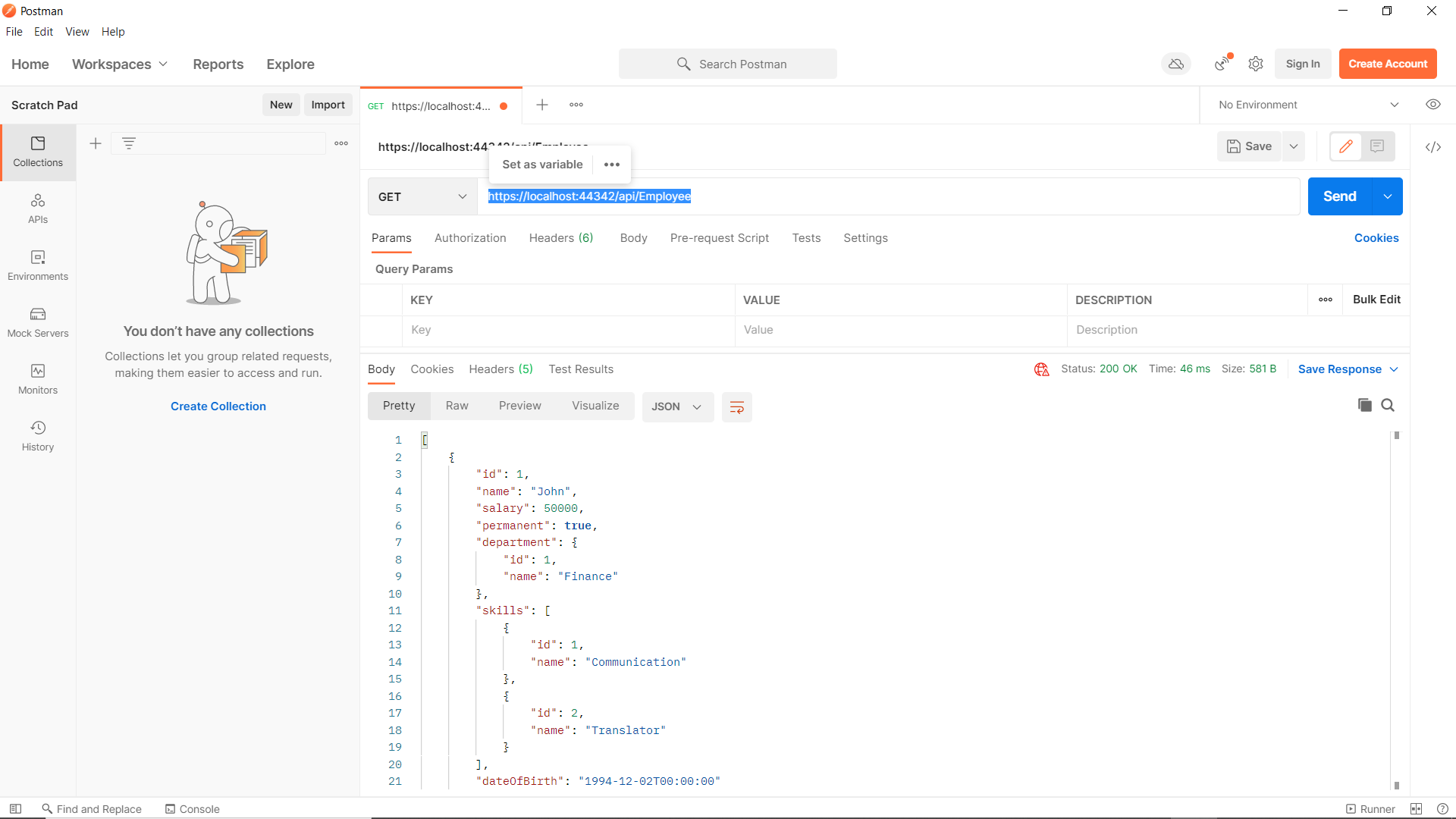
        }

    }

}

**OUTPUT:**





**Handson - 2**

1. **Reference WebAPI in ASP.Net MVC Core application for GET & POST**

* **MVC**

**EmployeeController.cs**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

using System.Web.Mvc;

using MVC.Models;

using System.Net.Http;

namespace MVC.Controllers

{

    public class EmployeeController : Controller

    {

        //

        // GET: /Employee/

        public ActionResult Index()

        {

            IEnumerable<mvcEmployeeModel> empList;

            HttpResponseMessage response = GlobalVariables.WebApiClient.GetAsync("Employee").Result;

            empList = response.Content.ReadAsAsync<IEnumerable<mvcEmployeeModel>>().Result;

            return View(empList);

        }

        public ActionResult AddOrEdit(int id = 0)

        {

            if (id == 0)

                return View(new mvcEmployeeModel());

            else

            {

                HttpResponseMessage response = GlobalVariables.WebApiClient.GetAsync("Employee/" + id.ToString()).Result;

                return View(response.Content.ReadAsAsync<mvcEmployeeModel>().Result);

            }

        }

        [HttpPost]

        public ActionResult AddOrEdit(mvcEmployeeModel emp)

        {

            if (emp.EmployeeID == 0)

            {

                HttpResponseMessage response = GlobalVariables.WebApiClient.PostAsJsonAsync("Employee", emp).Result;

                TempData["SuccessMessage"] = "Saved Successfully";

            }

            else

            {

                HttpResponseMessage response = GlobalVariables.WebApiClient.PutAsJsonAsync("Employee/" + emp.EmployeeID, emp).Result;

                TempData["SuccessMessage"] = "Updated Successfully";

            }

            return RedirectToAction("Index");

        }

        public ActionResult Delete(int id)

        {

            HttpResponseMessage response = GlobalVariables.WebApiClient.DeleteAsync("Employee/" + id.ToString()).Result;

            TempData["SuccessMessage"] = "Deleted Successfully";

            return RedirectToAction("Index");

        }

    }

}

**mvcEmployeeModel.cs**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

using System.ComponentModel.DataAnnotations;

using System.ComponentModel.DataAnnotations.Schema;

namespace MVC.Models

{

    [Table("Employee")]

    public class mvcEmployeeModel

    {

        public int EmployeeID { get; set; }

        [StringLength(50)]

        [Required(ErrorMessage = "This Field is Required")]

        public string Name { get; set; }

        [StringLength(50)]

        public string Position { get; set; }

        public int? Age { get; set; }

        public int? Salary { get; set; }

    }

}

**Index.cshtml**

@model IEnumerable<MVC.Models.mvcEmployeeModel>

@{

    ViewBag.Title = "Index";

}

<br />

<p>

    <a href="@Url.Action("AddOrEdit","Employee")" class="btn btn-default"><i class="fa fa-plus"></i> Create New</a>

</p>

<table class="table table-striped">

    <thead class="table-heading">

        <tr>

            <th>

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

            </th>

            <th>

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

            </th>

            <th>

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

            </th>

            <th>

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

            </th>

            <th>Actions</th>

        </tr>

    </thead>

    @foreach (var item in Model)

    {

        <tr>

            <td>

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

            </td>

            <td>

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

            </td>

            <td>

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

            </td>

            <td>

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

            </td>

            <td>

                <a href="@Url.Action("AddOrEdit", "Employee", new { id = item.EmployeeID})" class="btn btn-default"><i class="fa fa-pencil"></i>  Edit</a>

                <a onclick="Delete(@item.EmployeeID)" class="btn btn-default"><i class="fa fa-trash"></i>  Delete</a>

            </td>

        </tr>

    }

</table>

@section scripts{

    <script>

        $(function () {

            var successMessage = '@TempData["SuccessMessage"]'

            if (successMessage != '')

                alertify.success(successMessage);

        });

        function Delete(id) {

            alertify.confirm('Web Api CRUD Operations','Are You Sure to Delete this Record ?',function(){

                window.location.href  = '@Url.Action("Delete","Employee")/'+id;

            },null );

        }

    </script>

}

**AddOrEdit.cshtml**

@model MVC.Models.mvcEmployeeModel

@{

    ViewBag.Title = "AddOrEdit";

}

<div class="form-body">

    @using (Html.BeginForm())

    {

        @Html.HiddenFor(model => model.EmployeeID)

        <div class="form-group">

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

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

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

        </div>

        <div class="form-group">

            @Html.LabelFor(model => model.Position)

            @Html.EditorFor(model => model.Position)

            @Html.ValidationMessageFor(model => model.Position)

        </div>

        <div class="form-group">

            @Html.LabelFor(model => model.Age)

            @Html.EditorFor(model => model.Age)

            @Html.ValidationMessageFor(model => model.Age)

        </div>

        <div class="form-group">

            @Html.LabelFor(model => model.Salary)

            @Html.EditorFor(model => model.Salary)

            @Html.ValidationMessageFor(model => model.Salary)

        </div>

        <div class="form-group">

            <input type="submit" value="Submit" class="btn button" />

            <input type="reset" value="Reset" class="btn button" />

        </div>

    }

</div>

@section scripts{

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

}

**GlobalVariables.cs**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Net.Http;

using System.Net.Http.Headers;

using System.Web;

namespace MVC

{

    public static class GlobalVariables

    {

        public static HttpClient WebApiClient = new HttpClient();

        static GlobalVariables()

        {

            WebApiClient.BaseAddress = new Uri("https://localhost:44316/api/");

            WebApiClient.DefaultRequestHeaders.Clear();

            WebApiClient.DefaultRequestHeaders.Accept.Add(new MediaTypeWithQualityHeaderValue("application/json"));

        }

    }

}

**Site.css**

body {

    padding-top: 50px;

    padding-bottom: 20px;

}

/\* Set padding to keep content from hitting the edges \*/

.body-content {

    padding-left: 15px;

    padding-right: 15px;

}

/\* Set width on the form input elements since they're 100% wide by default \*/

input,

select,

textarea {

    max-width: 340px;

}

/\* styles for validation helpers \*/

.field-validation-error {

    color: #b94a48;

    font-size: 18px;

}

.field-validation-valid {

    display: none;

}

input.input-validation-error {

    border: 1px solid #b94a48;

}

input[type="checkbox"].input-validation-error {

    border: 0 none;

}

.validation-summary-errors {

    color: #b94a48;

}

.validation-summary-valid {

    display: none;

}

input, textarea {

    display: block;

    height: 100%;

    padding: 5px 10px;

    background: none;

    background-image: none;

    border: 1px solid #a0b3b0;

    color: #fff;

    border-radius: 0;

    transition: border-color .25s ease, box-shadow .25s ease;

}

.button {

    border: 1px solid #fff;

    outline: none;

    border-radius: 0;

    padding: 10px 10px;

    font-weight: 400;

    background: #2a3843;

    color: #fff;

    -webkit-appearance: none;

}

    .button:hover, .button:focus {

        color: #fff !important;

    }

.form-body {

    background: rgba(19, 35, 47, 0.9);

    padding: 20px 40px;

    max-width: 600px;

    margin: 30px auto;

    box-shadow: 0 4px 10px 4px rgba(19, 35, 47, 0.3);

}

body {

    background: #fff;

}

label {

    color: #fff;

    font-size: 22px;

}

.form-body input, textarea {

    font-size: 20px;

}

    textarea:focus, input:focus {

        outline: none;

    }

.table-heading {

    background-color: #2a3843;

    color: white;

}

.add-new {

    background-color: #1ab188;

}

* **MVC**

**EmployeeController.cs**

using System;

using System.Collections.Generic;

using System.Data;

using System.Data.Entity;

using System.Data.Entity.Infrastructure;

using System.Linq;

using System.Net;

using System.Net.Http;

using System.Web.Http;

using System.Web.Http.Description;

using WebApiInMVC.Models;

namespace WebApiInMVC.Controllers

{

    public class EmployeeController : ApiController

    {

        private DBModels db = new DBModels();

        // GET: api/Employee

        public IQueryable<Employee> GetEmployees()

        {

            return db.Employees;

        }

        // GET: api/Employee/5

        [ResponseType(typeof(Employee))]

        public IHttpActionResult GetEmployee(int id)

        {

            Employee employee = db.Employees.Find(id);

            if (employee == null)

            {

                return NotFound();

            }

            return Ok(employee);

        }

        // PUT: api/Employee/5

        [ResponseType(typeof(void))]

        public IHttpActionResult PutEmployee(int id, Employee employee)

        {

            if (id != employee.EmployeeID)

            {

                return BadRequest();

            }

            db.Entry(employee).State = EntityState.Modified;

            try

            {

                db.SaveChanges();

            }

            catch (DbUpdateConcurrencyException)

            {

                if (!EmployeeExists(id))

                {

                    return NotFound();

                }

                else

                {

                    throw;

                }

            }

            return StatusCode(HttpStatusCode.NoContent);

        }

        // POST: api/Employee

        [ResponseType(typeof(Employee))]

        public IHttpActionResult PostEmployee(Employee employee)

        {

            db.Employees.Add(employee);

            db.SaveChanges();

            return CreatedAtRoute("DefaultApi", new { id = employee.EmployeeID }, employee);

        }

        // DELETE: api/Employee/5

        [ResponseType(typeof(Employee))]

        public IHttpActionResult DeleteEmployee(int id)

        {

            Employee employee = db.Employees.Find(id);

            if (employee == null)

            {

                return NotFound();

            }

            db.Employees.Remove(employee);

            db.SaveChanges();

            return Ok(employee);

        }

        protected override void Dispose(bool disposing)

        {

            if (disposing)

            {

                db.Dispose();

            }

            base.Dispose(disposing);

        }

        private bool EmployeeExists(int id)

        {

            return db.Employees.Count(e => e.EmployeeID == id) > 0;

        }

    }

}

**ValuesController.cs**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Net;

using System.Net.Http;

using System.Web.Http;

namespace WebApiInMVC.Controllers

{

    public class ValuesController : ApiController

    {

        // GET api/values

        public IEnumerable<string> Get()

        {

            return new string[] { "value1", "value2" };

        }

        // GET api/values/5

        public string Get(int id)

        {

            return "value";

        }

        // POST api/values

        public void Post([FromBody] string value)

        {

        }

        // PUT api/values/5

        public void Put(int id, [FromBody] string value)

        {

        }

        // DELETE api/values/5

        public void Delete(int id)

        {

        }

    }

}

**DBModels.cs**

using System;

using System.ComponentModel.DataAnnotations.Schema;

using System.Data.Entity;

using System.Linq;

namespace WebApiInMVC.Models

{

    public partial class DBModels : DbContext

    {

        public DBModels()

            : base("name=DBModel")

        {

        }

        public virtual DbSet<Employee> Employees { get; set; }

        protected override void OnModelCreating(DbModelBuilder modelBuilder)

        {

            modelBuilder.Entity<Employee>()

                .Property(e => e.Name)

                .IsUnicode(false);

            modelBuilder.Entity<Employee>()

                .Property(e => e.Position)

                .IsUnicode(false);

        }

    }

}

**Employee.cs**

namespace WebApiInMVC.Models

{

    using System;

    using System.Collections.Generic;

    using System.ComponentModel.DataAnnotations;

    using System.ComponentModel.DataAnnotations.Schema;

    using System.Data.Entity.Spatial;

    [Table("Employee")]

    public partial class Employee

    {

        public int EmployeeID { get; set; }

        [StringLength(50)]

        public string Name { get; set; }

        [StringLength(50)]

        public string Position { get; set; }

        public int? Age { get; set; }

        public int? Salary { get; set; }

    }

}

**Web.config**

<?xml version="1.0" encoding="utf-8"?>

<!--

  For more information on how to configure your ASP.NET application, please visit

  https://go.microsoft.com/fwlink/?LinkId=301879

  -->

<configuration>

  <configSections>

    <section name="entityFramework" type="System.Data.Entity.Internal.ConfigFile.EntityFrameworkSection, EntityFramework, Version=6.0.0.0, Culture=neutral, PublicKeyToken=b77a5c561934e089" requirePermission="false" />

  <!-- For more information on Entity Framework configuration, visit http://go.microsoft.com/fwlink/?LinkID=237468 --></configSections>

  <appSettings>

    <add key="webpages:Version" value="3.0.0.0" />

    <add key="webpages:Enabled" value="false" />

    <add key="ClientValidationEnabled" value="true" />

    <add key="UnobtrusiveJavaScriptEnabled" value="true" />

  </appSettings>

  <system.web>

    <compilation debug="true" targetFramework="4.7.2" />

    <httpRuntime targetFramework="4.7.2" />

  </system.web>

  <system.webServer>

    <handlers>

      <remove name="ExtensionlessUrlHandler-Integrated-4.0" />

      <remove name="OPTIONSVerbHandler" />

      <remove name="TRACEVerbHandler" />

      <add name="ExtensionlessUrlHandler-Integrated-4.0" path="\*." verb="\*" type="System.Web.Handlers.TransferRequestHandler" preCondition="integratedMode,runtimeVersionv4.0" />

    </handlers>

  </system.webServer>

  <runtime>

    <assemblyBinding xmlns="urn:schemas-microsoft-com:asm.v1">

      <dependentAssembly>

        <assemblyIdentity name="Antlr3.Runtime" publicKeyToken="eb42632606e9261f" />

        <bindingRedirect oldVersion="0.0.0.0-3.5.0.2" newVersion="3.5.0.2" />

      </dependentAssembly>

      <dependentAssembly>

        <assemblyIdentity name="Newtonsoft.Json" culture="neutral" publicKeyToken="30ad4fe6b2a6aeed" />

        <bindingRedirect oldVersion="0.0.0.0-12.0.0.0" newVersion="12.0.0.0" />

      </dependentAssembly>

      <dependentAssembly>

        <assemblyIdentity name="System.Web.Optimization" publicKeyToken="31bf3856ad364e35" />

        <bindingRedirect oldVersion="1.0.0.0-1.1.0.0" newVersion="1.1.0.0" />

      </dependentAssembly>

      <dependentAssembly>

        <assemblyIdentity name="WebGrease" publicKeyToken="31bf3856ad364e35" />

        <bindingRedirect oldVersion="0.0.0.0-1.6.5135.21930" newVersion="1.6.5135.21930" />

      </dependentAssembly>

      <dependentAssembly>

        <assemblyIdentity name="System.Web.Helpers" publicKeyToken="31bf3856ad364e35" />

        <bindingRedirect oldVersion="1.0.0.0-3.0.0.0" newVersion="3.0.0.0" />

      </dependentAssembly>

      <dependentAssembly>

        <assemblyIdentity name="System.Web.WebPages" publicKeyToken="31bf3856ad364e35" />

        <bindingRedirect oldVersion="1.0.0.0-3.0.0.0" newVersion="3.0.0.0" />

      </dependentAssembly>

      <dependentAssembly>

        <assemblyIdentity name="System.Web.Mvc" publicKeyToken="31bf3856ad364e35" />

        <bindingRedirect oldVersion="1.0.0.0-5.2.7.0" newVersion="5.2.7.0" />

      </dependentAssembly>

    </assemblyBinding>

  </runtime>

  <system.codedom>

    <compilers>

      <compiler language="c#;cs;csharp" extension=".cs" type="Microsoft.CodeDom.Providers.DotNetCompilerPlatform.CSharpCodeProvider, Microsoft.CodeDom.Providers.DotNetCompilerPlatform, Version=2.0.1.0, Culture=neutral, PublicKeyToken=31bf3856ad364e35" warningLevel="4" compilerOptions="/langversion:default /nowarn:1659;1699;1701" />

      <compiler language="vb;vbs;visualbasic;vbscript" extension=".vb" type="Microsoft.CodeDom.Providers.DotNetCompilerPlatform.VBCodeProvider, Microsoft.CodeDom.Providers.DotNetCompilerPlatform, Version=2.0.1.0, Culture=neutral, PublicKeyToken=31bf3856ad364e35" warningLevel="4" compilerOptions="/langversion:default /nowarn:41008 /define:\_MYTYPE=\&quot;Web\&quot; /optionInfer+" />

    </compilers>

  </system.codedom>

  <entityFramework>

    <defaultConnectionFactory type="System.Data.Entity.Infrastructure.LocalDbConnectionFactory, EntityFramework">

      <parameters>

        <parameter value="mssqllocaldb" />

      </parameters>

    </defaultConnectionFactory>

    <providers>

      <provider invariantName="System.Data.SqlClient" type="System.Data.Entity.SqlServer.SqlProviderServices, EntityFramework.SqlServer" />

    </providers>

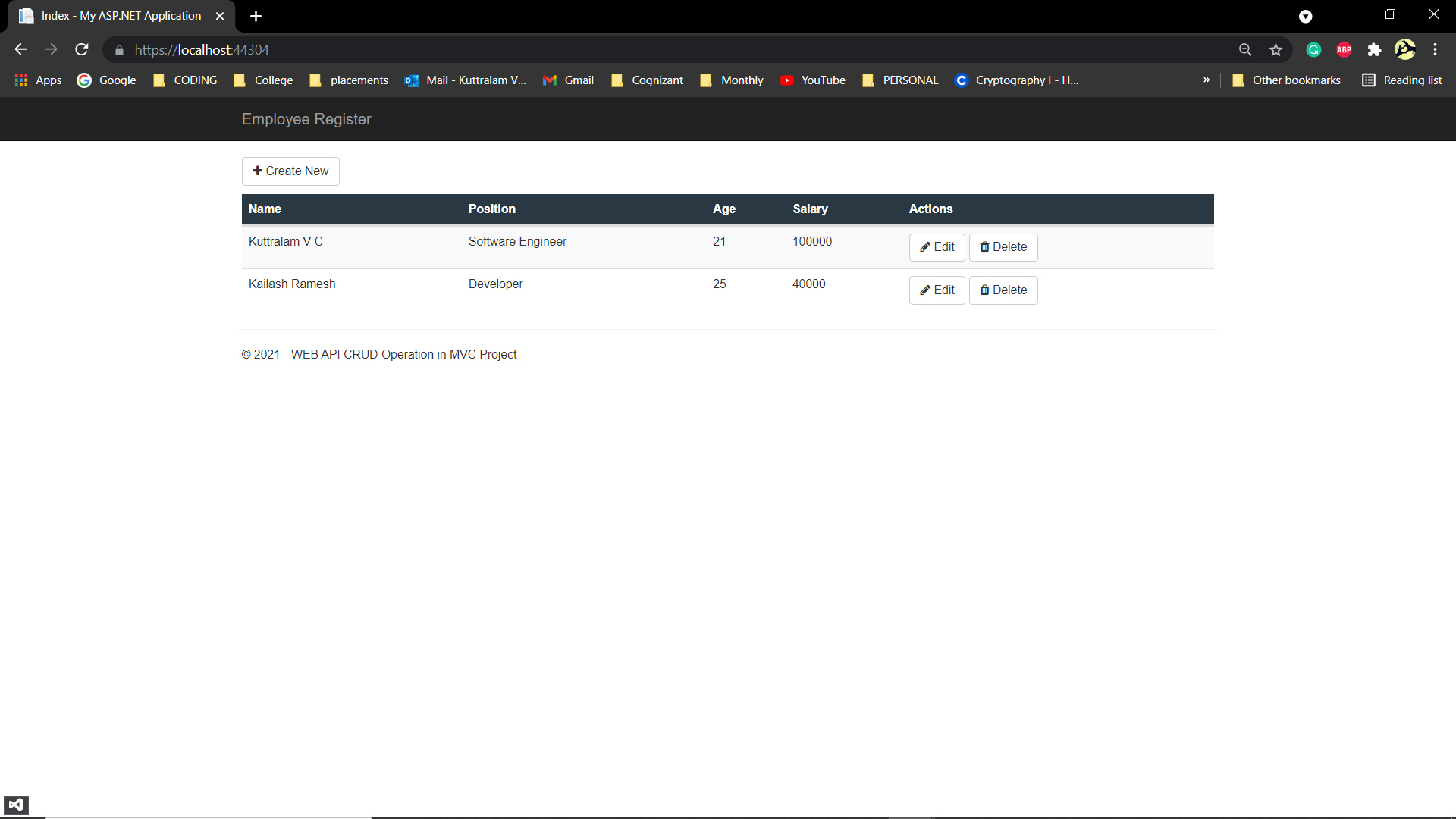
  </entityFramework>

  <connectionStrings>

    <add name="DBModel" connectionString="data source=DESKTOP-B0SQG0N;initial catalog=CRUDDB;integrated security=True;MultipleActiveResultSets=True;App=EntityFramework" providerName="System.Data.SqlClient" />

  </connectionStrings>

</configuration>

**OUTPUT:**

