1. **Web Api using .Net core with Swagger Sai Kumaravelu A.T.**

**Startup.cs**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Threading.Tasks;

using Microsoft.AspNetCore.Builder;

using Microsoft.AspNetCore.Hosting;

using Microsoft.Extensions.Configuration;

using Microsoft.Extensions.DependencyInjection;

using Microsoft.Extensions.Logging;

using Swashbuckle.AspNetCore.Swagger;

namespace WebApplication6

{

public class Startup

{

public Startup(IHostingEnvironment env)

{

var builder = new ConfigurationBuilder()

.SetBasePath(env.ContentRootPath)

.AddJsonFile("appsettings.json", optional: false, reloadOnChange: true)

.AddJsonFile($"appsettings.{env.EnvironmentName}.json", optional: true)

.AddEnvironmentVariables();

Configuration = builder.Build();

}

public IConfigurationRoot Configuration { get; }

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

public void ConfigureServices(IServiceCollection services)

{

// Add framework services.

services.AddMvc();

services.AddSwaggerGen(c =>

{

c.SwaggerDoc("v1", new Info

{

Title = "Swagger Demo",

Version = "v1",

Description = "TBD",

TermsOfService = "None",

Contact = new Contact() { Name = "John Doe", Email = "john@xyzmail.com", Url = "https://www.example.com" },

License = new License() { Name = "License Terms", Url = "https://www.example.com" }

});

});

}

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

public void Configure(IApplicationBuilder app, IHostingEnvironment env, ILoggerFactory loggerFactory)

{

loggerFactory.AddConsole(Configuration.GetSection("Logging"));

loggerFactory.AddDebug();

app.UseMvc();

app.UseSwagger();

app.UseSwaggerUI(c =>

{

// specifying the Swagger JSON endpoint.

c.SwaggerEndpoint("/swagger/v1/swagger.json", "Swagger Demo");

});

}

}

internal class Info : Swashbuckle.AspNetCore.Swagger.Info

{

public string Title { get; set; }

public string Version { get; set; }

public string Description { get; set; }

public string TermsOfService { get; set; }

public object Contact { get; set; }

public object License { get; set; }

}

}

**ValueController.cs**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Threading.Tasks;

using Microsoft.AspNetCore.Mvc;

namespace WebApplication6.Controllers

{

[Route("api/[controller]")]

public class ValuesController : Controller

{

// GET api/values

[HttpGet]

public IEnumerable<string> Get()

{

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

}

// GET api/values/5

[HttpGet("{id}")]

public string Get(int id)

{

return "value";

}

// POST api/values

[HttpPost]

public void Post([FromBody]string value)

{

}

// PUT api/values/5

[HttpPut("{id}")]

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

{

}

// DELETE api/values/5

[HttpDelete("{id}")]

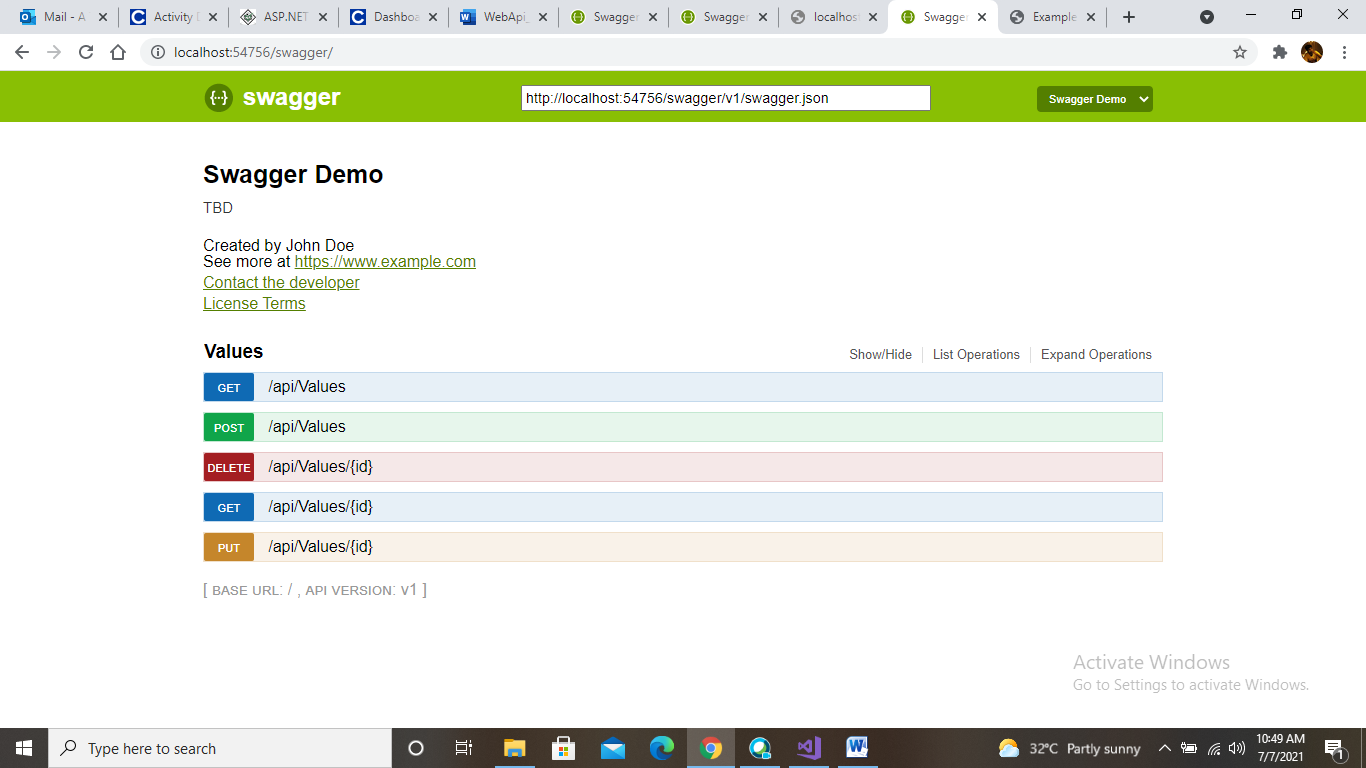
public void Delete(int id)

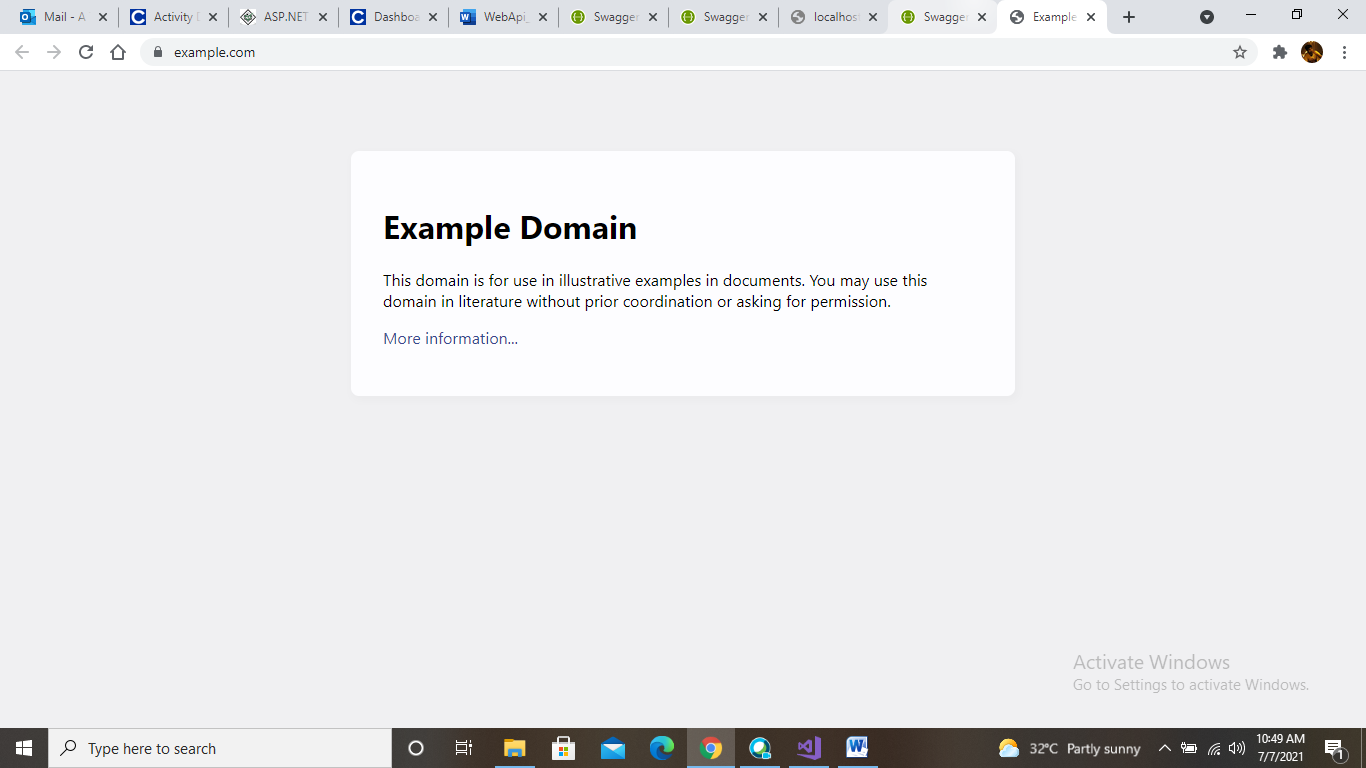
{

}

}

}

**OUTPUT:**

****

**2. Web Api using custom model class**

**Employee.cs**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Threading.Tasks;

namespace WebApplication6.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 string Department { get; set; }

public string Skills { get; set; }

public DateTime DateOfBirth { get; set; }

}

}

**EmployeesController.cs**

using Microsoft.AspNetCore.Mvc;

using Microsoft.AspNetCore.Http;

using Microsoft.AspNetCore.Authorization;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Threading.Tasks;

using WebApplication6.Models;

using WebApplication6.Filters;

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

namespace WebApplication6.Controllers

{

[Route("api/[controller]")]

public class EmployeesController : ControllerBase

{

private Employee[] emp = new Employee[]

{

new Employee { Id=1 , Name="sai" , Salary=20000 , Permanent=true, Department="CS",Skills="Dotnet" ,DateOfBirth=new DateTime(1998,08,07) },

new Employee { Id=2 , Name="kumara" , Salary=15000 , Permanent=true, Department="Electronics",Skills="java" ,DateOfBirth=new DateTime(1999,11,09) } ,

new Employee { Id=3 , Name="velu" , Salary=10000 , Permanent=false, Department="Mech",Skills="php" ,DateOfBirth=new DateTime(1997,02,12) } ,

};

private IEnumerable<Employee> GetStandardEmployeeList()

{

return emp;

}

// GET: api/<ValuesController1>

[CustomAuthFilter]

[HttpGet]

public IEnumerable<Employee> Get()

{

return GetStandardEmployeeList();

}

// GET api/<ValuesController1>/5

[HttpGet("{id}")]

[ProducesResponseType(StatusCodes.Status200OK)]

[ProducesResponseType(StatusCodes.Status404NotFound)]

public IActionResult GetById(int id)

{

var prod = emp.FirstOrDefault((p) => p.Id == id);

if (prod == null)

{

return NotFound();

}

return Ok(prod);

}

// POST api/<ValuesController1>

[HttpPost]

public void Post([FromBody] string value)

{

}

// PUT api/<ValuesController1>/5

[HttpPut("{id}")]

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

{

}

// DELETE api/<ValuesController1>/5

[HttpDelete("{id}")]

public void Delete(int id)

{

}

}

}

**CustomAuthFilter.cs**

using Microsoft.AspNetCore.Mvc;

using Microsoft.AspNetCore.Mvc.Filters;

using Microsoft.Extensions.Primitives;

using System;

using System.Collections.Generic;

using System.Diagnostics;

using System.Linq;

using System.Threading.Tasks;

namespace WebApplication6.Filters

{

public class CustomAuthFilter : ActionFilterAttribute

{

public override void OnActionExecuting(ActionExecutingContext context)

{

if (context.HttpContext.Request.Query.ContainsKey("Authorization") && context.HttpContext.Request.Query["Authorization"] == "true")

{

context.Result = new UnauthorizedResult();

}

else

{

base.OnActionExecuting(context);

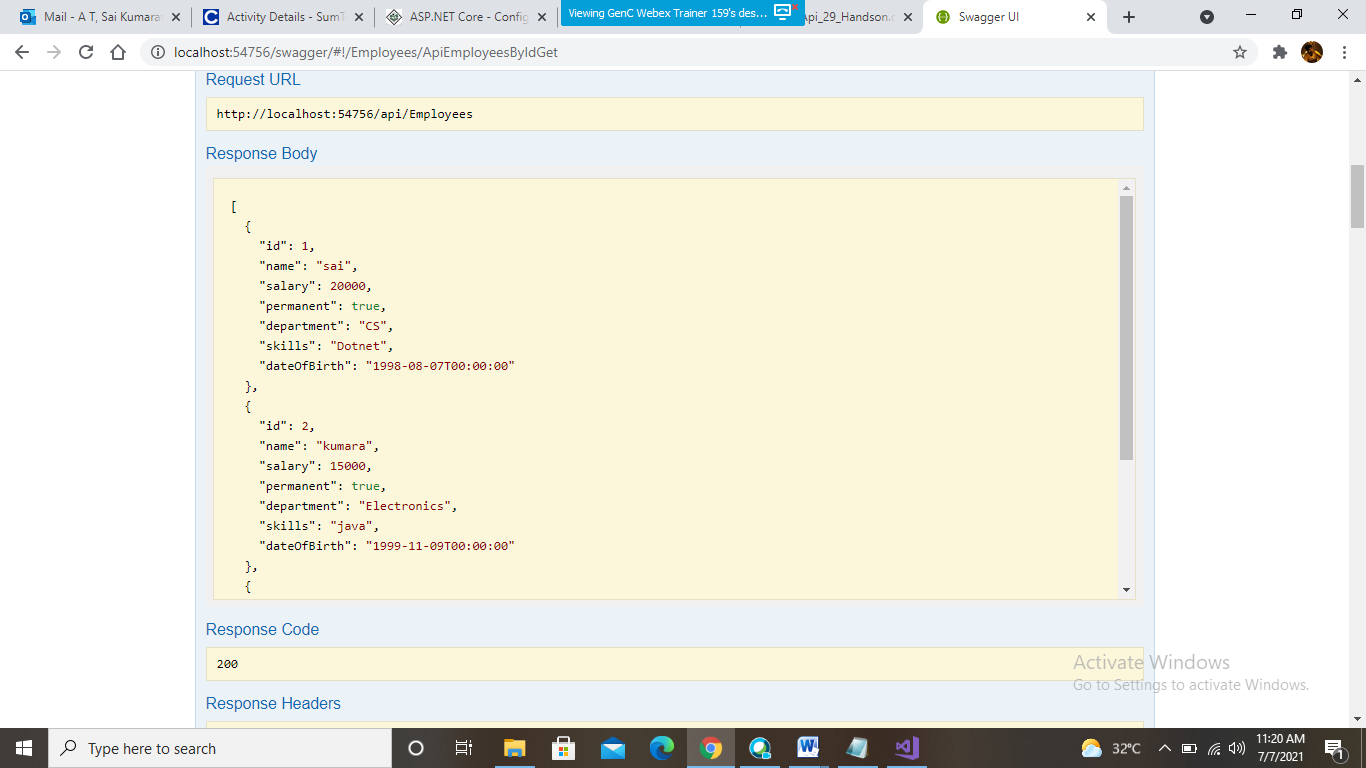
}

}

}

}

**OUTPUT:**

****

**3. Web Api CRUD operation**

**Employee.cs**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Threading.Tasks;

namespace WebApplication6.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 string Department { get; set; }

public string Skills { get; set; }

public DateTime DateOfBirth { get; set; }

}

}

**EmployeesController.cs**

using Microsoft.AspNetCore.Mvc;

using Microsoft.AspNetCore.Http;

using Microsoft.AspNetCore.Authorization;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Threading.Tasks;

using ThirdWebAPI.Models;

using ThirdWebAPI.Filters;

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

namespace ThirdWebAPI.Controllers

{

[Route("api/[controller]")]

[ApiController]

public class EmployeesController : ControllerBase

{

private static List<Employee> \_emp = new List<Employee>();

// GET: api/<ValuesController1>

[HttpGet(Name = "GetAllStudent")]

public IActionResult Get()

{

return new ObjectResult(\_emp);

}

// GET api/<ValuesController1>/5

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

public IActionResult Get(int id)

{

return new ObjectResult(\_emp.FirstOrDefault(p => p.Id == id));

}

// POST api/<ValuesController1>

[HttpPost(Name = "CreateStudent")]

public IActionResult Post([FromBody] Employee emps)

{

\_emp.Add(emps);

return CreatedAtRoute("GetStudent", new { id = emps.Id }, emps);

}

// PUT api/<ValuesController1>/5

[HttpPut("{id}", Name = "UpdateStudent")]

public IActionResult Put(int id, [FromBody] Employee emps)

{

\_emp.FirstOrDefault(p => p.Id == id).Name = emps.Name;

return CreatedAtRoute("GetStudent", new { id = emps.Id }, emps);

}

// DELETE api/<ValuesController1>/5

[HttpDelete("{id}", Name = "DeleteStudent")]

public IActionResult Delete(int id)

{

var \_emps = \_emp.FirstOrDefault(p => p.Id == id);

\_emp.Remove(\_emps);

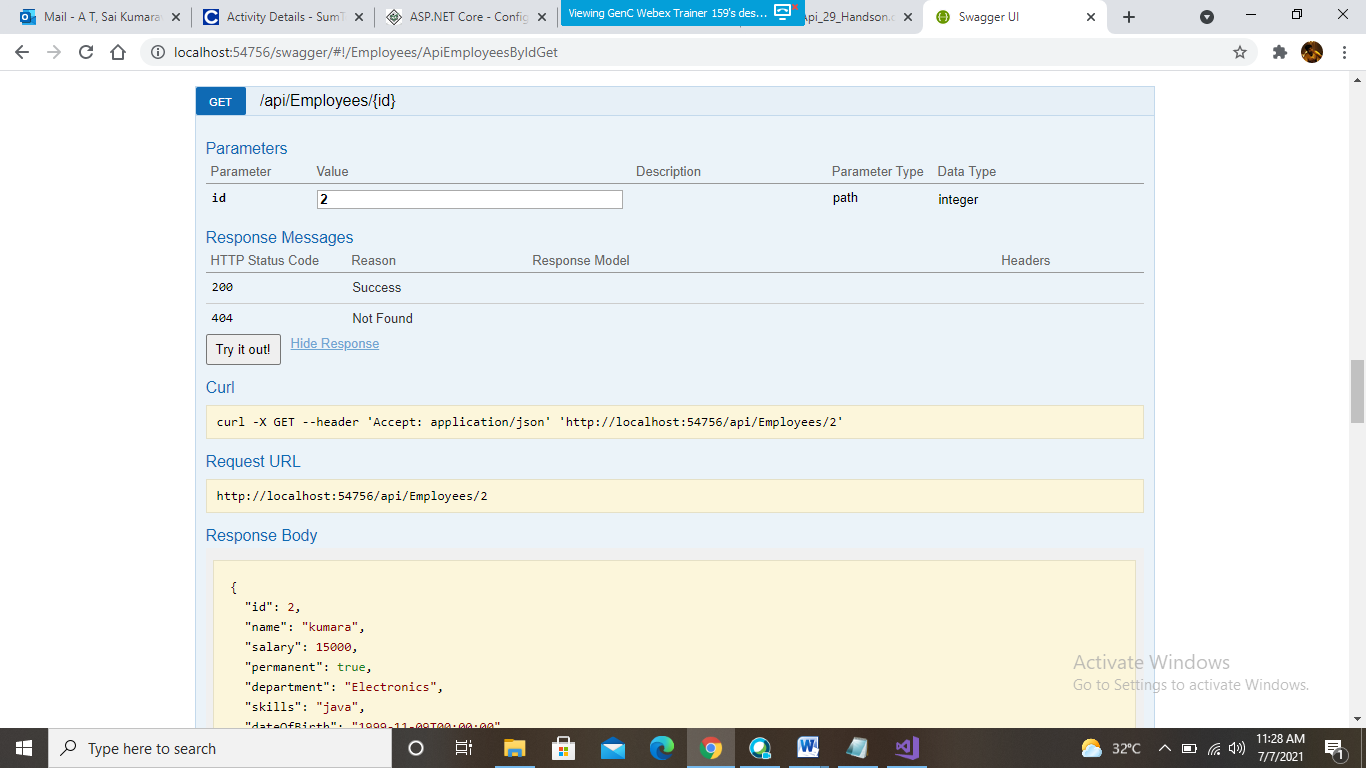
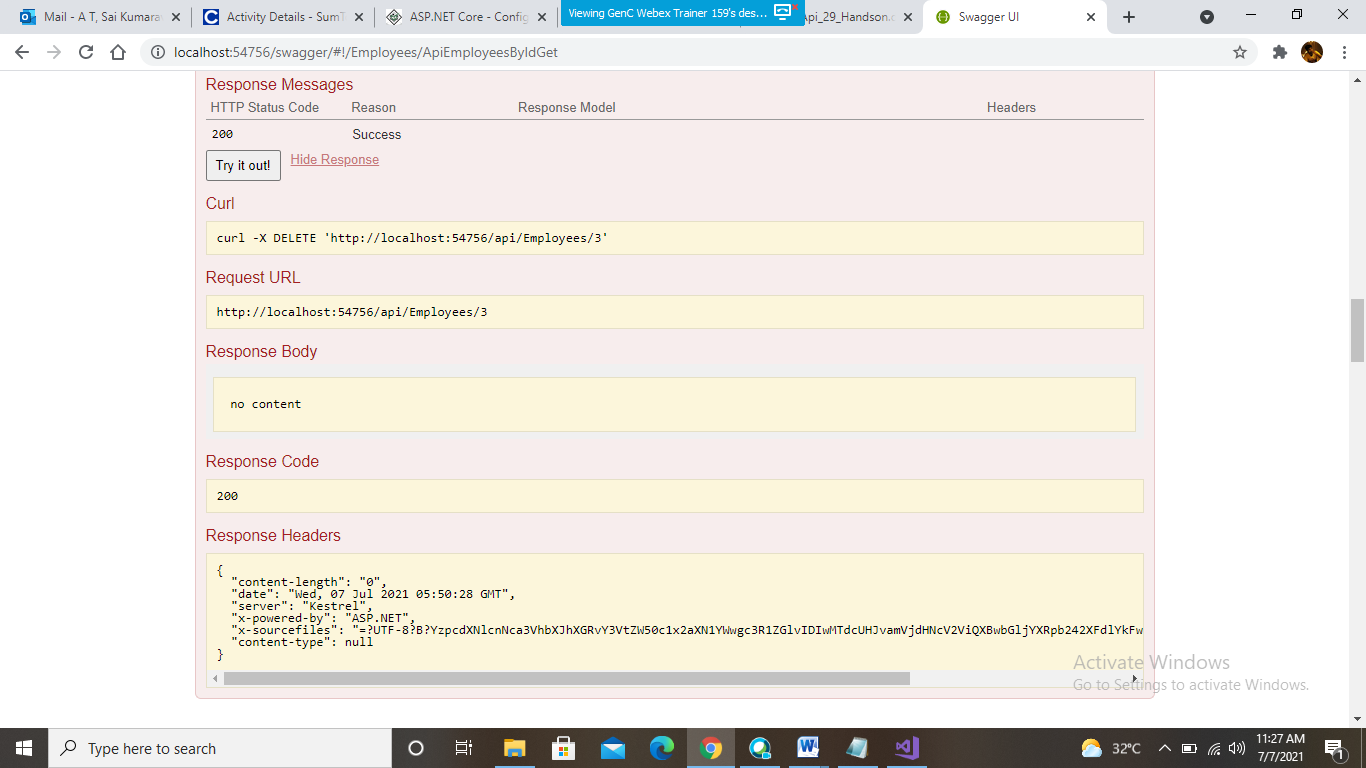
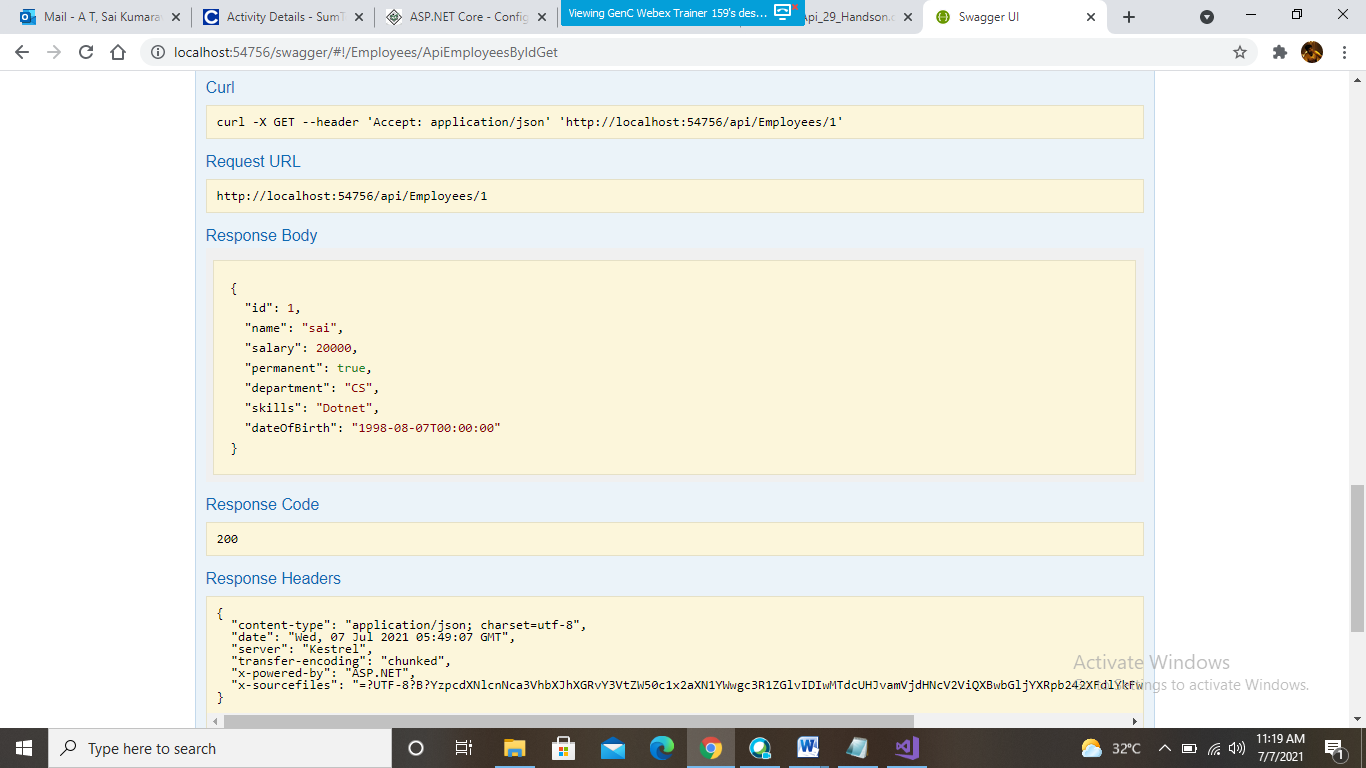
return new NoContentResult();

}

}

}

**OUTPUT:**

****