**Mandatory Handson Week 4 Solutions:**

**Shashwat Bharadwaj**

**Superset ID:6360517**

**Roll no:2228059**

**1. WebApi handson 1**

**Controllers Folder**

**-ValuesController.cs**

using Microsoft.AspNetCore.Mvc;

using System.Collections.Generic;

namespace MyWebApiApp.Controllers

{

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

[ApiController]

public class ValuesController : ControllerBase

{

[HttpGet]

public ActionResult<IEnumerable<string>> Get()

{

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

}

[HttpGet("{id}")]

public ActionResult<string> Get(int id)

{

return "value" + id;

}

[HttpPost]

public void Post([FromBody] string value)

{

}

[HttpPut("{id}")]

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

{

}

[HttpDelete("{id}")]

public void Delete(int id)

{

}

}

}

**-Program.cs**

var builder = WebApplication.CreateBuilder(args);

// ✅ Add controller support

builder.Services.AddControllers();

// Swagger setup

builder.Services.AddEndpointsApiExplorer();

builder.Services.AddSwaggerGen();

var app = builder.Build();

if (app.Environment.IsDevelopment())

{

app.UseSwagger();

app.UseSwaggerUI();

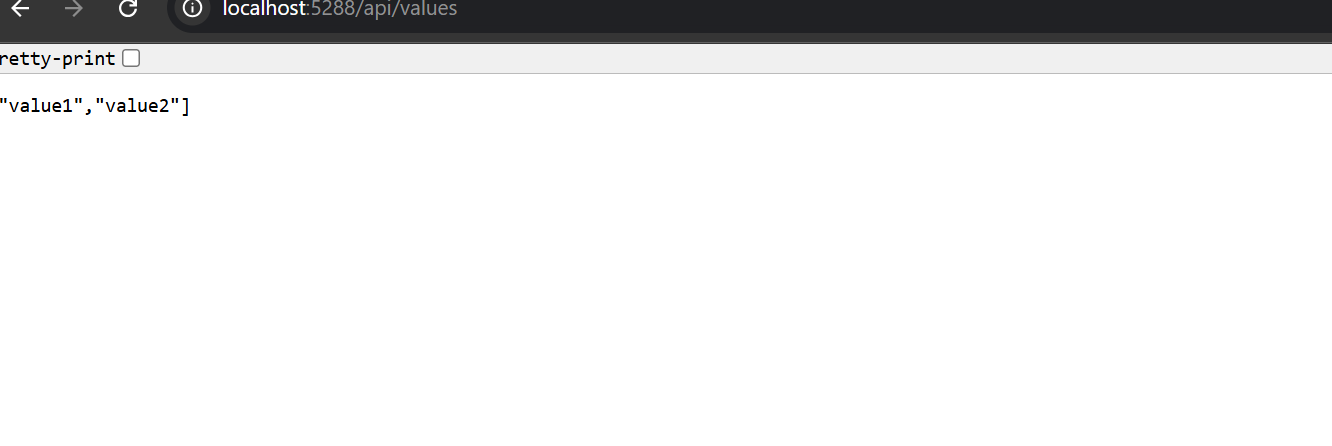
}

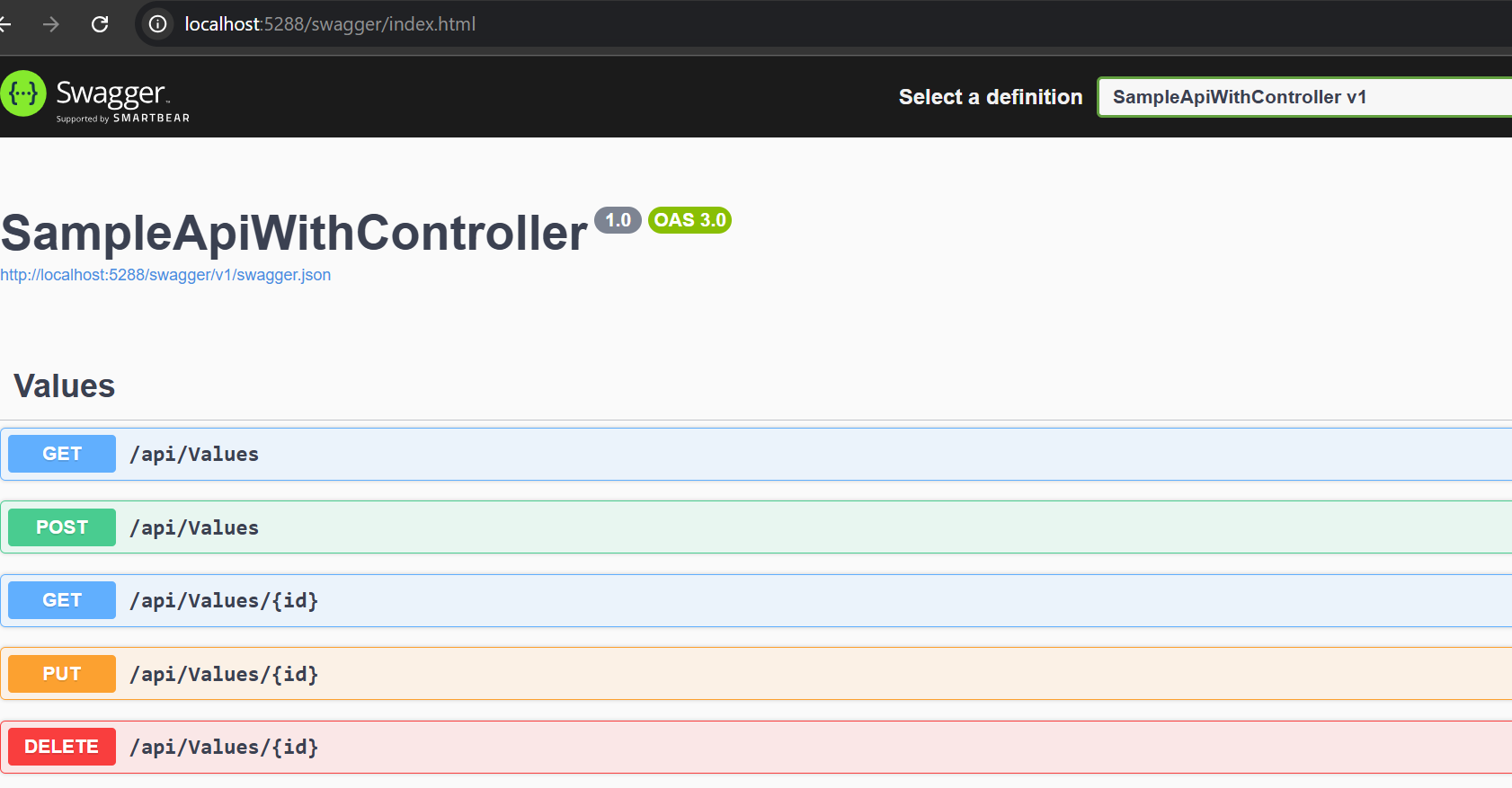
app.UseHttpsRedirection();

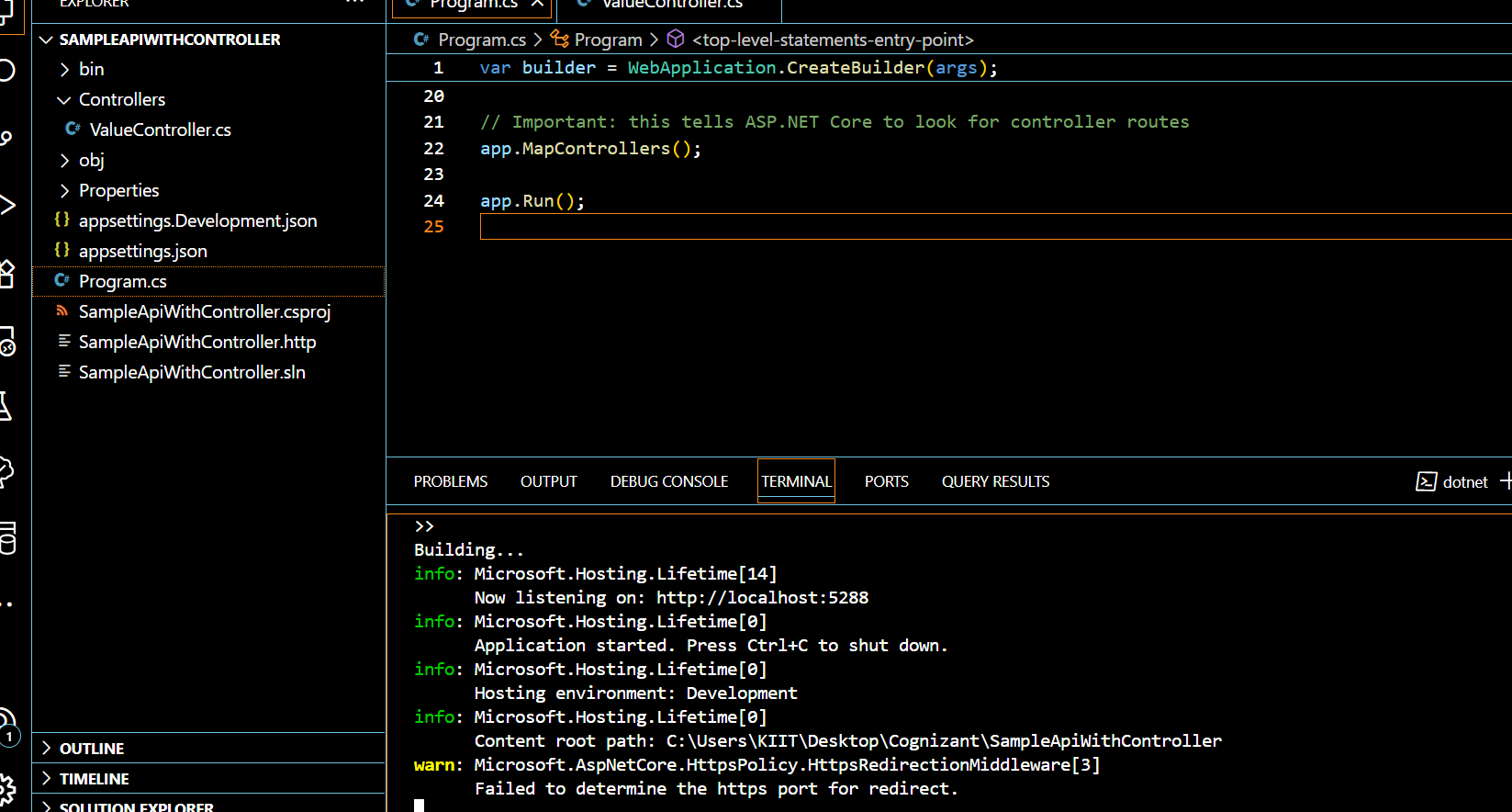
// ✅ Map controller endpoints

app.MapControllers();

app.Run();







**2.WebApi handson 2**

**-Program.cs**

using Microsoft.OpenApi.Models;

var builder = WebApplication.CreateBuilder(args);

// Add services to the container.

builder.Services.AddControllers();

// Add SwaggerGen

builder.Services.AddSwaggerGen(c =>

{

c.SwaggerDoc("v1", new OpenApiInfo

{

Title = "Swagger Demo",

Version = "v1",

Description = "TBD",

TermsOfService = new Uri("https://example.com/terms"),

Contact = new OpenApiContact

{

Name = "John Doe",

Email = "john@xyzmail.com",

Url = new Uri("https://example.com/contact")

},

License = new OpenApiLicense

{

Name = "License Terms",

Url = new Uri("https://example.com/license")

}

});

});

var app = builder.Build();

// Configure middleware

if (app.Environment.IsDevelopment())

{

app.UseSwagger();

app.UseSwaggerUI(c =>

{

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

});

}

app.UseHttpsRedirection();

app.UseAuthorization();

app.MapControllers();

app.Run();

**Controllers**

**-EmployeeController.cs**

using Microsoft.AspNetCore.Mvc;

namespace SwaggerDemoApi.Controllers

{

[ApiController]

[Route("api/[controller]")] // This will be /api/employee

public class EmployeeController : ControllerBase

{

static List<string> employees = new List<string> { "Alice", "Bob", "Charlie" };

[HttpGet]

public ActionResult<IEnumerable<string>> GetEmployees()

{

return Ok(employees);

}

[HttpPost]

public ActionResult AddEmployee([FromBody] string name)

{

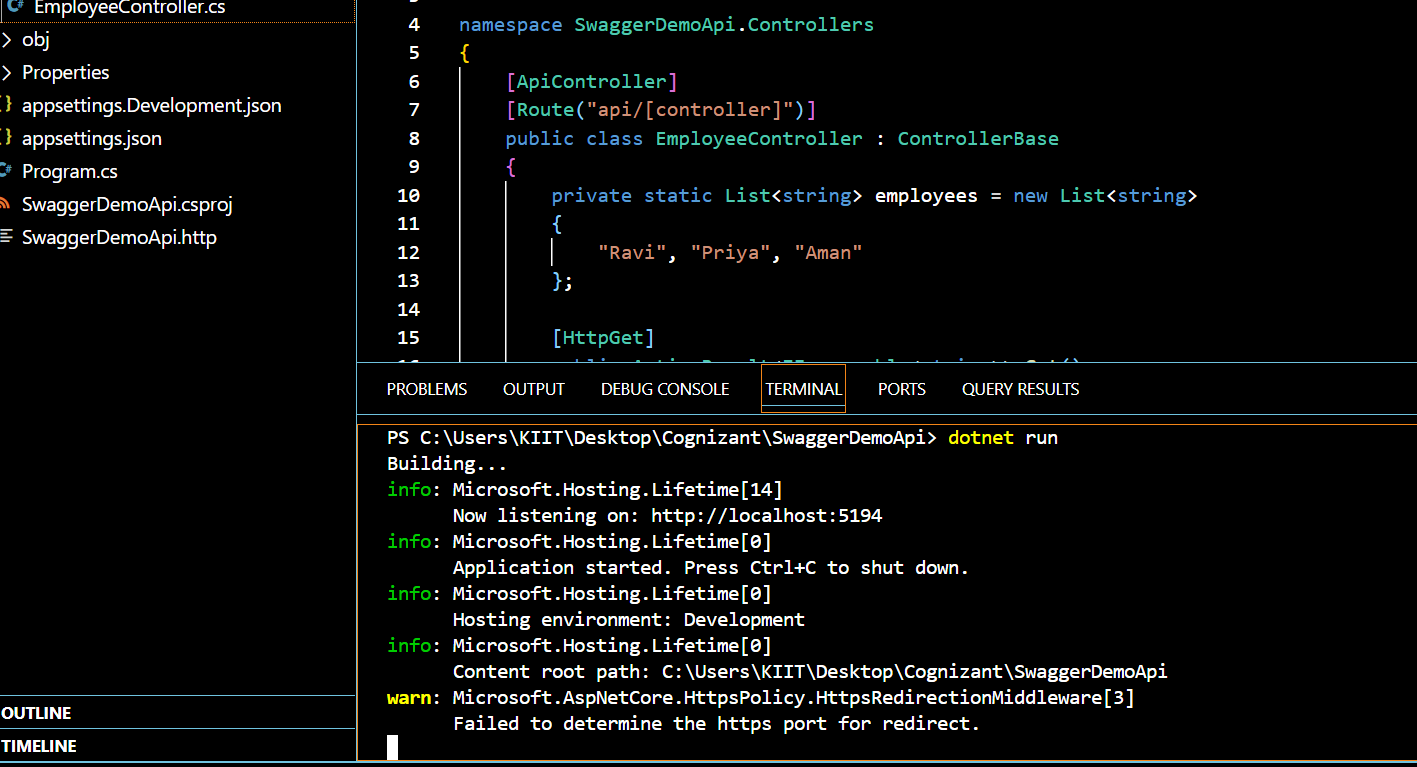
employees.Add(name);

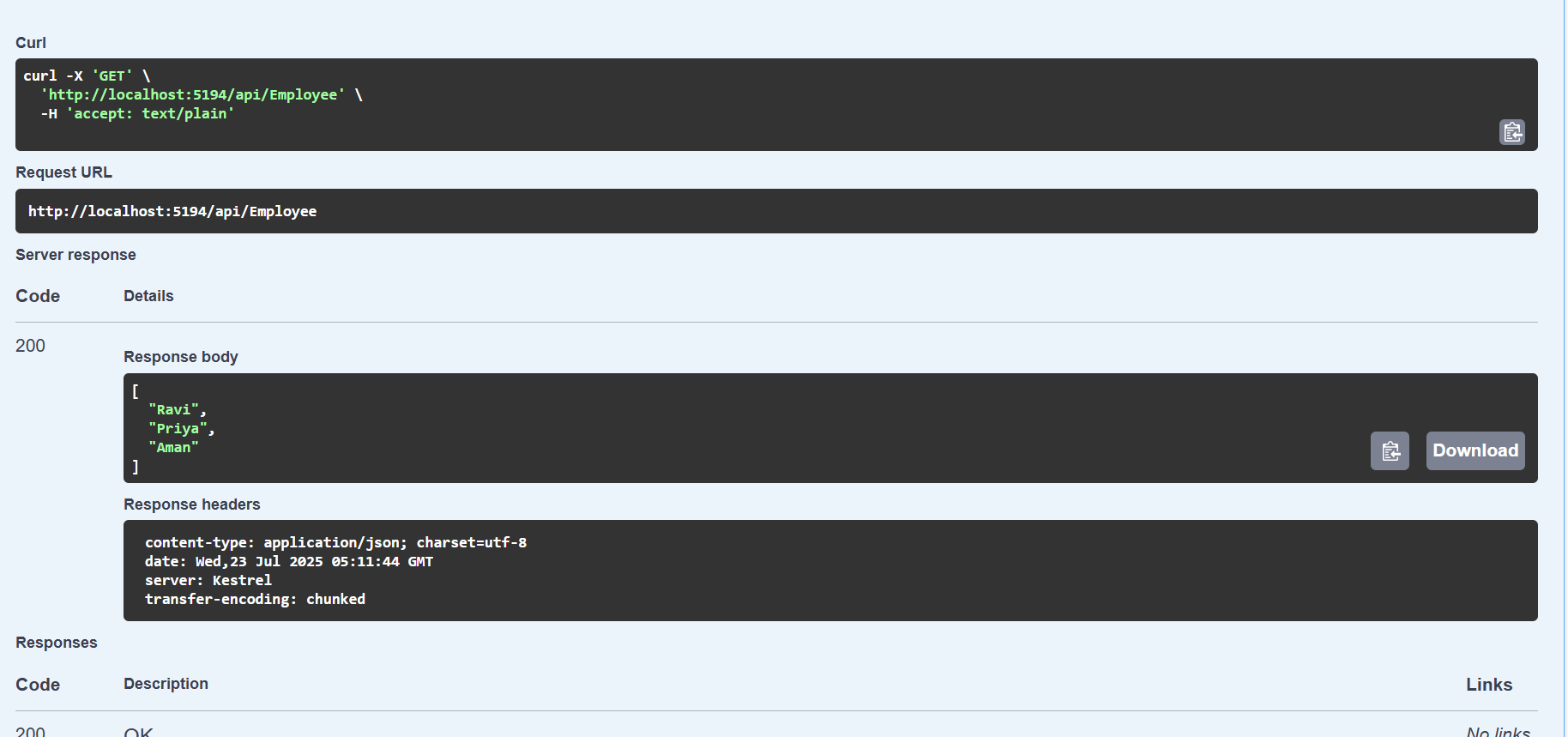
return Ok(employees);

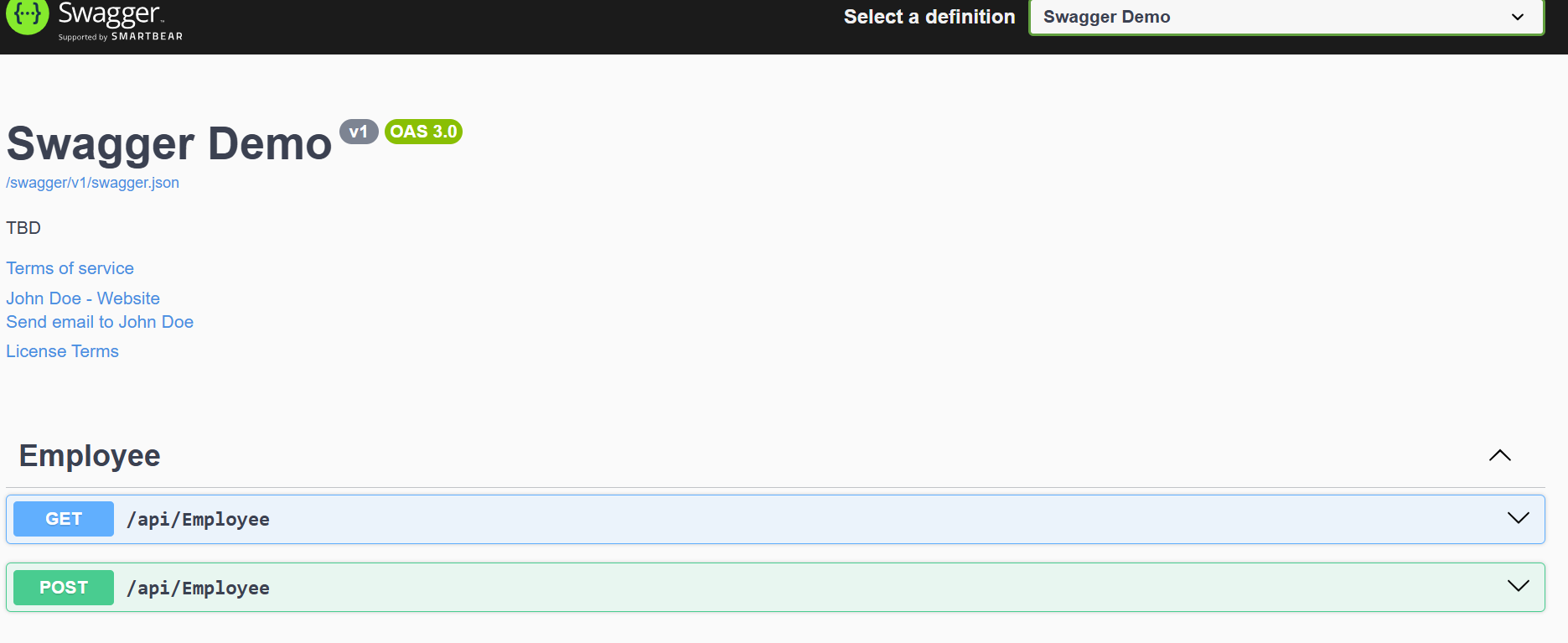
}

}

}







**3. WebApi handson 3**

**Models**

**-Department.cs**

namespace WebApiHandsonProject.Models

{

public class Department

{

public int Id { get; set; }

public string Name { get; set; }

}

}

**-Skills.cs**

namespace WebApiHandsonProject.Models

{

public class Skill

{

public int Id { get; set; }

public string Name { get; set; }

}

}

**-Employee.cs**

using System;

using System.Collections.Generic;

namespace WebApiHandsonProject.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; }

}

}

**Filters**

**-CustomAuthFilter.cs**

using Microsoft.AspNetCore.Mvc;

using Microsoft.AspNetCore.Mvc.Filters;

namespace WebApiHandsonProject.Filters

{

public class CustomAuthFilter : ActionFilterAttribute

{

public override void OnActionExecuting(ActionExecutingContext context)

{

var authHeader = context.HttpContext.Request.Headers["Authorization"].FirstOrDefault();

if (string.IsNullOrEmpty(authHeader))

{

context.Result = new BadRequestObjectResult("Invalid request - No Auth token");

return;

}

if (!authHeader.Contains("Bearer"))

{

context.Result = new BadRequestObjectResult("Invalid request - Token present but Bearer unavailable");

return;

}

base.OnActionExecuting(context);

}

}

}

**-CustomExceptionFilter.cs**

using Microsoft.AspNetCore.Mvc;

using Microsoft.AspNetCore.Mvc.Filters;

namespace WebApiHandsonProject.Filters

{

public class CustomExceptionFilter : IExceptionFilter

{

public void OnException(ExceptionContext context)

{

string path = Path.Combine(Directory.GetCurrentDirectory(), "errorlog.txt");

File.WriteAllText(path, context.Exception.ToString());

context.Result = new ObjectResult("An unexpected error occurred.")

{

StatusCode = 500

};

}

}

}

**Controllers**

**-EmployeeController.cs**

using Microsoft.AspNetCore.Mvc;

using WebApiHandsonProject.Models;

using WebApiHandsonProject.Filters;

namespace WebApiHandsonProject.Controllers

{

[ApiController]

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

[TypeFilter(typeof(CustomAuthFilter))]

public class EmployeeController : ControllerBase

{

private readonly List<Employee> \_employees;

public EmployeeController()

{

\_employees = GetStandardEmployeeList();

}

private List<Employee> GetStandardEmployeeList()

{

return new List<Employee>

{

new Employee

{

Id = 1,

Name = "Riya",

Salary = 50000,

Permanent = true,

DateOfBirth = new DateTime(1998, 5, 21),

Department = new Department { Id = 1, Name = "IT" },

Skills = new List<Skill>

{

new Skill { Id = 1, Name = "C#" },

new Skill { Id = 2, Name = "SQL" }

}

}

};

}

[HttpGet]

public ActionResult<List<Employee>> GetStandard()

{

throw new Exception("Intentional error to test CustomExceptionFilter.");

// return Ok(\_employees);

}

[HttpPost]

public IActionResult Post([FromBody] Employee emp)

{

if (emp == null)

return BadRequest();

return Ok(emp);

}

}

}

**Program.cs**

using WebApiHandsonProject.Filters;

var builder = WebApplication.CreateBuilder(args);

// Add services

builder.Services.AddControllers(options =>

{

options.Filters.Add(typeof(CustomExceptionFilter));

});

builder.Services.AddEndpointsApiExplorer();

builder.Services.AddSwaggerGen();

var app = builder.Build();

// Middlewares

if (app.Environment.IsDevelopment())

{

app.UseSwagger();

app.UseSwaggerUI();

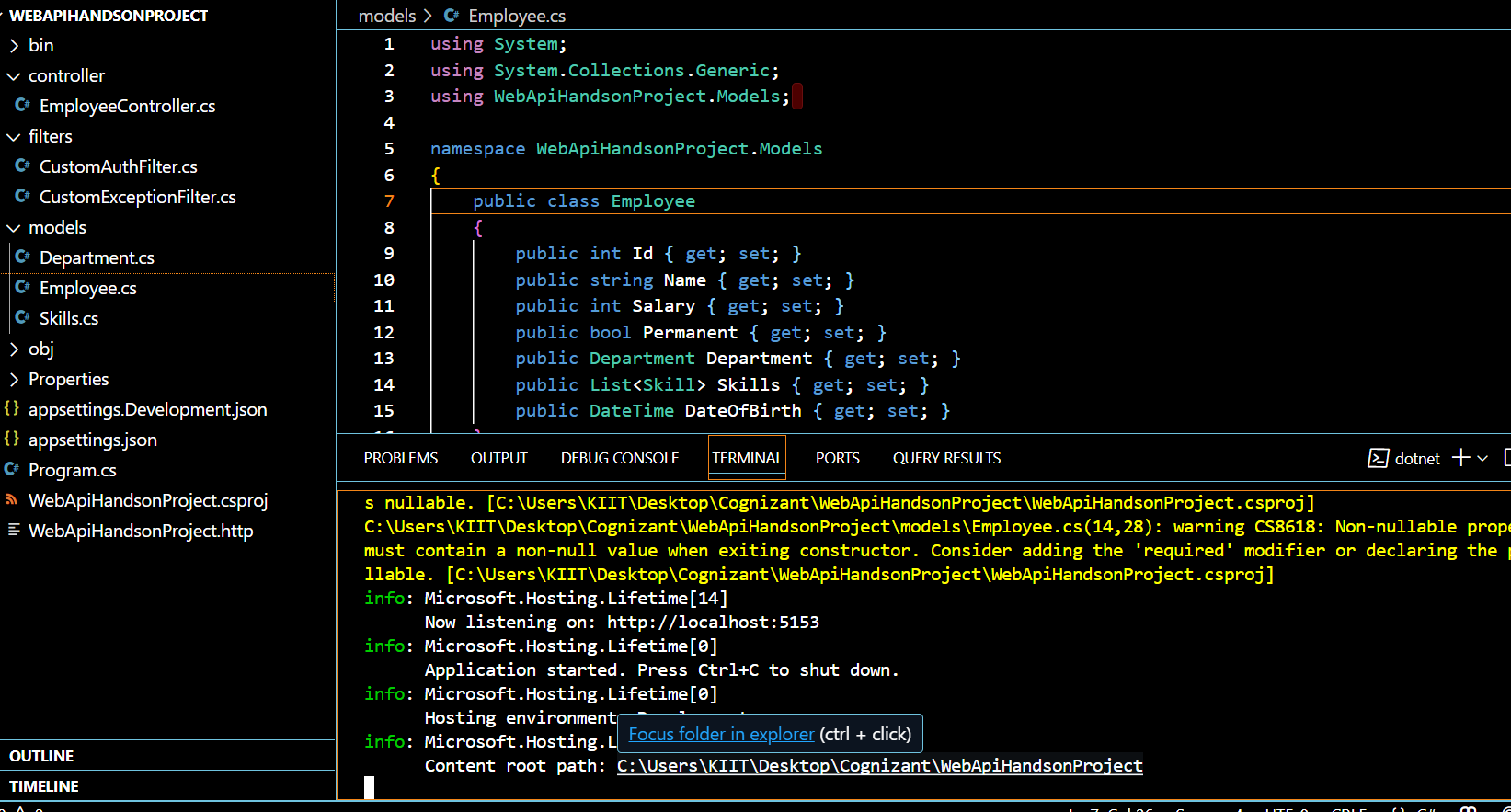
}

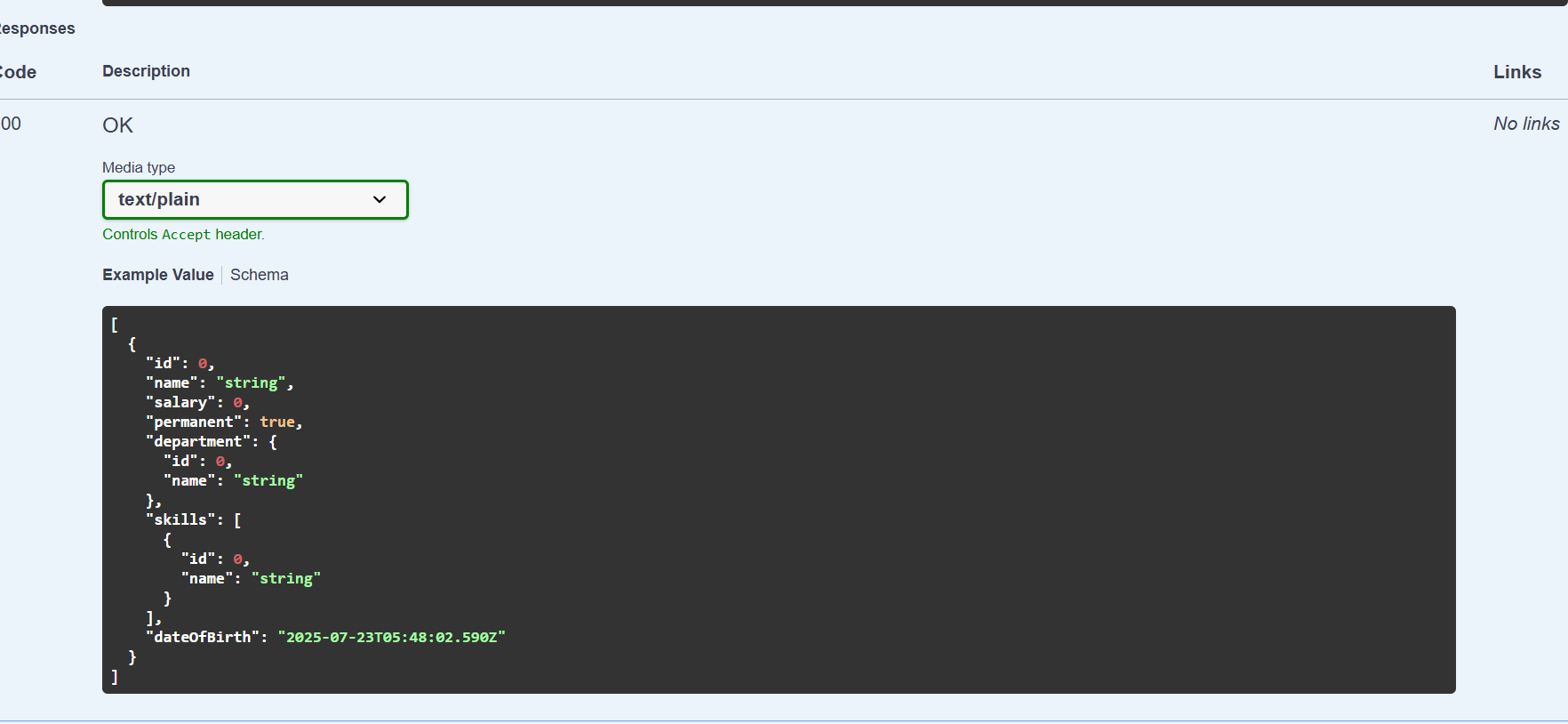
app.UseHttpsRedirection();

app.UseAuthorization();

app.MapControllers();

app.Run();





**4. WebApi handson 4**

Updating **EmployeeController.cs** to include memory list

using Microsoft.AspNetCore.Mvc;

using WebApiHandsonProject.Models;

using System;

using System.Collections.Generic;

using System.Linq;

namespace WebApiHandsonProject.Controllers

{

[ApiController]

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

public class EmployeeController : ControllerBase

{

// Hardcoded in-memory list

public static List<Employee> employeeList = new List<Employee>

{

new Employee

{

Id = 1,

Name = "Meera",

Salary = 20000,

Permanent = true,

Department = new Department { Id = 1, Name = "Payroll" },

Skills = new List<Skill>

{

new Skill { Id = 1, Name = "Communication" },

new Skill { Id = 2, Name = "Teamwork" }

},

DateOfBirth = new DateTime(2000, 10, 20)

},

new Employee

{

Id = 2,

Name = "Raj",

Salary = 25000,

Permanent = false,

Department = new Department { Id = 2, Name = "HR" },

Skills = new List<Skill>

{

new Skill { Id = 3, Name = "Recruitment" },

new Skill { Id = 4, Name = "Excel" }

},

DateOfBirth = new DateTime(1999, 5, 10)

}

};

[HttpGet]

public IActionResult GetEmployees()

{

return Ok(employeeList);

}

[HttpPut("{id}")]

public ActionResult<Employee> UpdateEmployee(int id, [FromBody] Employee updatedEmp)

{

if (id <= 0)

return BadRequest("Invalid employee id");

var emp = employeeList.FirstOrDefault(e => e.Id == id);

if (emp == null)

return BadRequest("Invalid employee id");

// Update employee details

emp.Name = updatedEmp.Name;

emp.Salary = updatedEmp.Salary;

emp.Permanent = updatedEmp.Permanent;

emp.Department = updatedEmp.Department;

emp.Skills = updatedEmp.Skills;

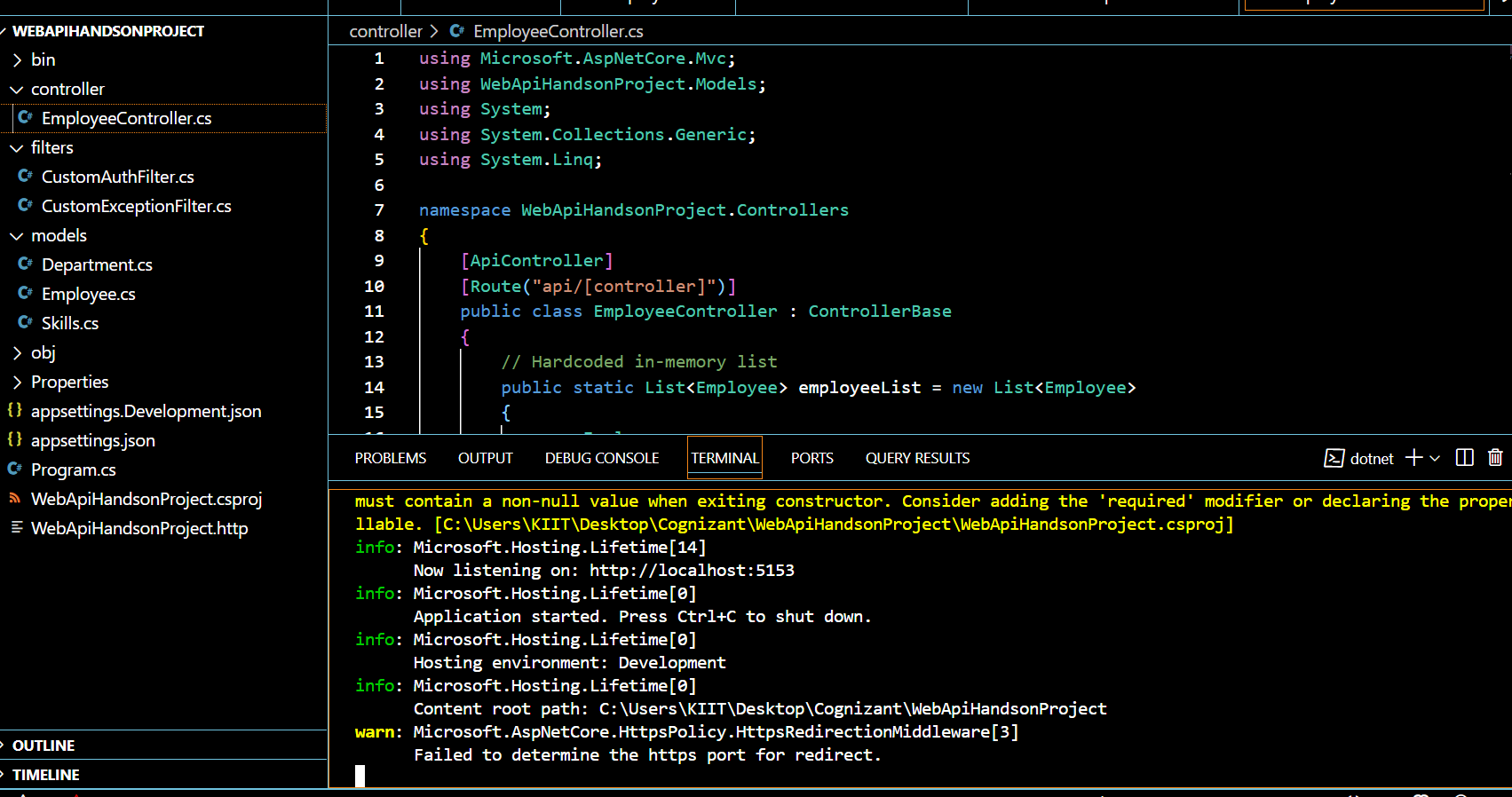
emp.DateOfBirth = updatedEmp.DateOfBirth;

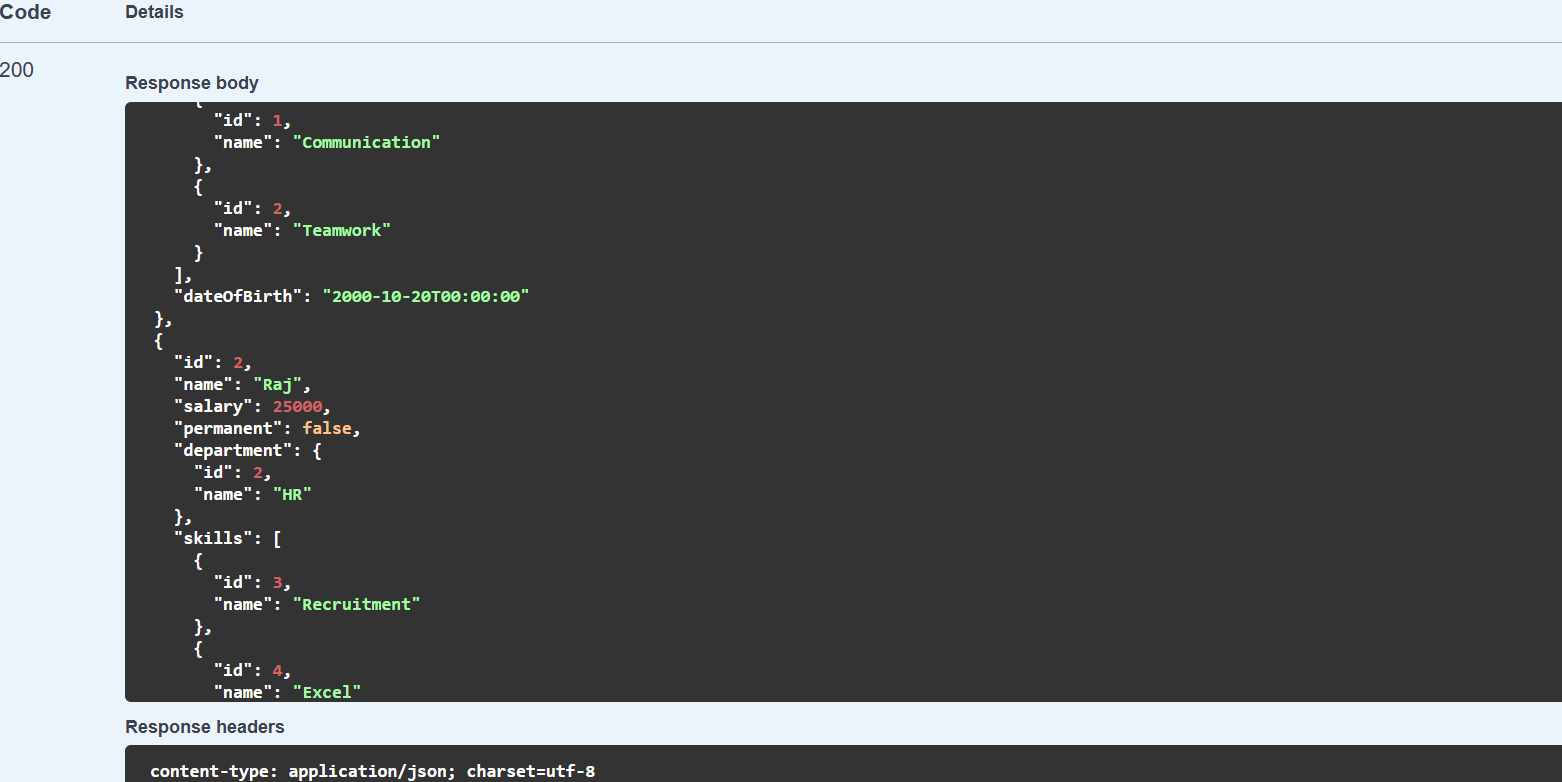
return Ok(emp);

}

}

}







**5. WebApi handson 5**

**-Program.cs**

using Microsoft.AspNetCore.Authentication.JwtBearer;

using Microsoft.IdentityModel.Tokens;

using System.Text;

var builder = WebApplication.CreateBuilder(args);

var securityKey = "mysuperdupersecretkey1234567890abcd";

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

// Add services

builder.Services.AddControllers();

builder.Services.AddAuthentication(options =>

{

options.DefaultAuthenticateScheme = JwtBearerDefaults.AuthenticationScheme;

options.DefaultChallengeScheme = JwtBearerDefaults.AuthenticationScheme;

}).AddJwtBearer(options =>

{

options.TokenValidationParameters = new TokenValidationParameters

{

ValidateIssuer = true,

ValidateAudience = true,

ValidateLifetime = true,

ValidateIssuerSigningKey = true,

ValidIssuer = "mySystem",

ValidAudience = "myUsers",

IssuerSigningKey = key

};

});

builder.Services.AddEndpointsApiExplorer();

builder.Services.AddSwaggerGen();

var app = builder.Build();

app.UseSwagger();

app.UseSwaggerUI();

app.UseHttpsRedirection();

app.UseAuthentication();

app.UseAuthorization();

app.MapControllers(); // <- critical line

app.Run();

**Controllers**

**-AuthController.cs**

using Microsoft.AspNetCore.Mvc;

using Microsoft.IdentityModel.Tokens;

using System.IdentityModel.Tokens.Jwt;

using System.Security.Claims;

using System.Text;

using Microsoft.AspNetCore.Authorization;

namespace JwtAuthWorkingApp.Controllers

{

[ApiController]

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

[AllowAnonymous]

public class AuthController : ControllerBase

{

[HttpGet]

public IActionResult GetToken()

{

string token = GenerateJSONWebToken(1, "Admin");

return Ok(new { token });

}

private string GenerateJSONWebToken(int userId, string userRole)

{

var securityKey = new SymmetricSecurityKey(Encoding.UTF8.GetBytes("mysuperdupersecretkey1234567890abcd"));

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

var claims = new[]

{

new Claim(ClaimTypes.Role, userRole),

new Claim("UserId", userId.ToString())

};

var token = new JwtSecurityToken(

issuer: "mySystem",

audience: "myUsers",

claims: claims,

expires: DateTime.Now.AddMinutes(2),

signingCredentials: credentials);

return new JwtSecurityTokenHandler().WriteToken(token);

}

}

}

**-EmployeeController.cs**

using Microsoft.AspNetCore.Authorization;

using Microsoft.AspNetCore.Mvc;

namespace SampleApiWithController.Controllers

{

[ApiController]

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

[Authorize(Roles = "Admin,POC")] // Accepts both roles

public class EmployeeController : ControllerBase

{

[HttpGet]

public IActionResult Get()

{

return Ok(new { Message = "Access granted to Employee data." });

}

}

}

