**WEEK 4**

**ASP.NET Core 8.0 Web API**

**Weatherforcast.cs**

**CODE-->**

namespace FirstWebAPI

{

public class WeatherForecast

{

public DateOnly Date { get; set; }

public int TemperatureC { get; set; }

public int TemperatureF => 32 + (int)(TemperatureC / 0.5556);

public string? Summary { get; set; }

}

}

**Employee.cs**

**CODE->**

namespace FirstWebAPI.Models

{

public class Department

{

public int Id { get; set; }

public string DeptName { get; set; }

}

public class Skill

{

public int Id { get; set; }

public string SkillName { get; set; }

}

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; }

}

}

**Program.cs**

**CODE->**

var builder = WebApplication.CreateBuilder(args);

// Add services to the container.

builder.Services.AddControllers();

// Learn more about configuring Swagger/OpenAPI at https://aka.ms/aspnetcore/swashbuckle

builder.Services.AddEndpointsApiExplorer();

builder.Services.AddSwaggerGen(c =>

{

c.SwaggerDoc("v1", new Microsoft.OpenApi.Models.OpenApiInfo

{

Title = "Swagger Demo",

Version = "v1",

Description = "Simple Web API with Employee Read & Write actions",

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

Contact = new Microsoft.OpenApi.Models.OpenApiContact

{

Name = "John Doe",

Email = "john@xyzmail.com",

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

},

License = new Microsoft.OpenApi.Models.OpenApiLicense

{

Name = "Use under LICX",

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

}

});

});

var app = builder.Build();

// Configure the HTTP request pipeline.

if (app.Environment.IsDevelopment())

{

app.UseSwagger();

app.UseSwaggerUI();

}

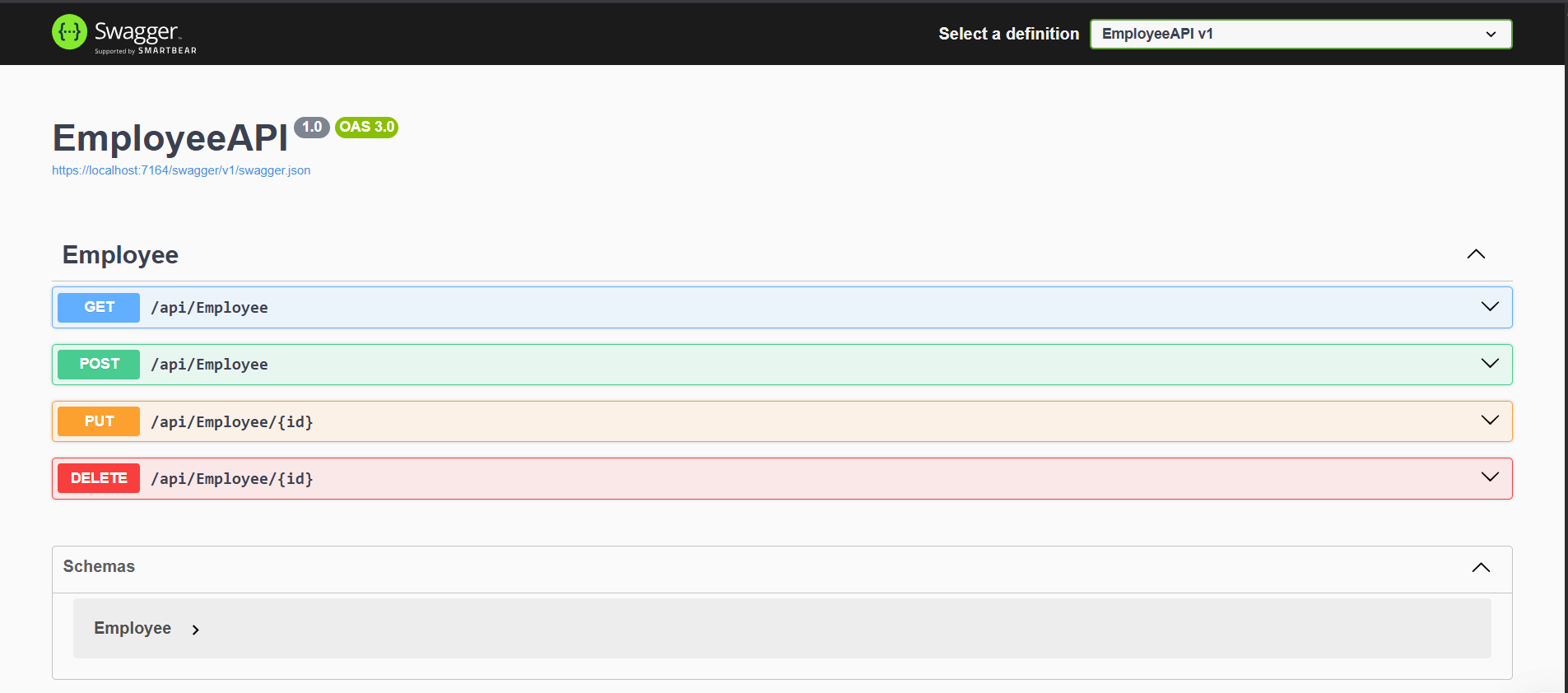
app.UseHttpsRedirection();

app.UseAuthorization();

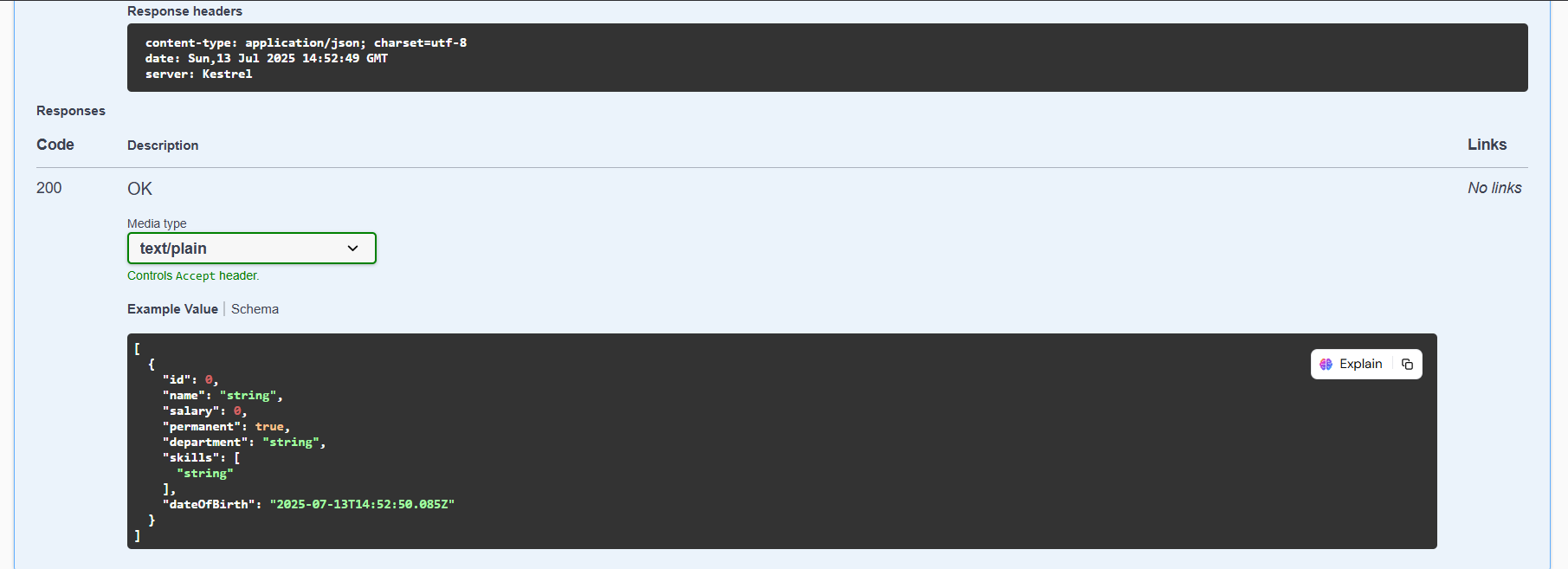
app.MapControllers();

app.Run();

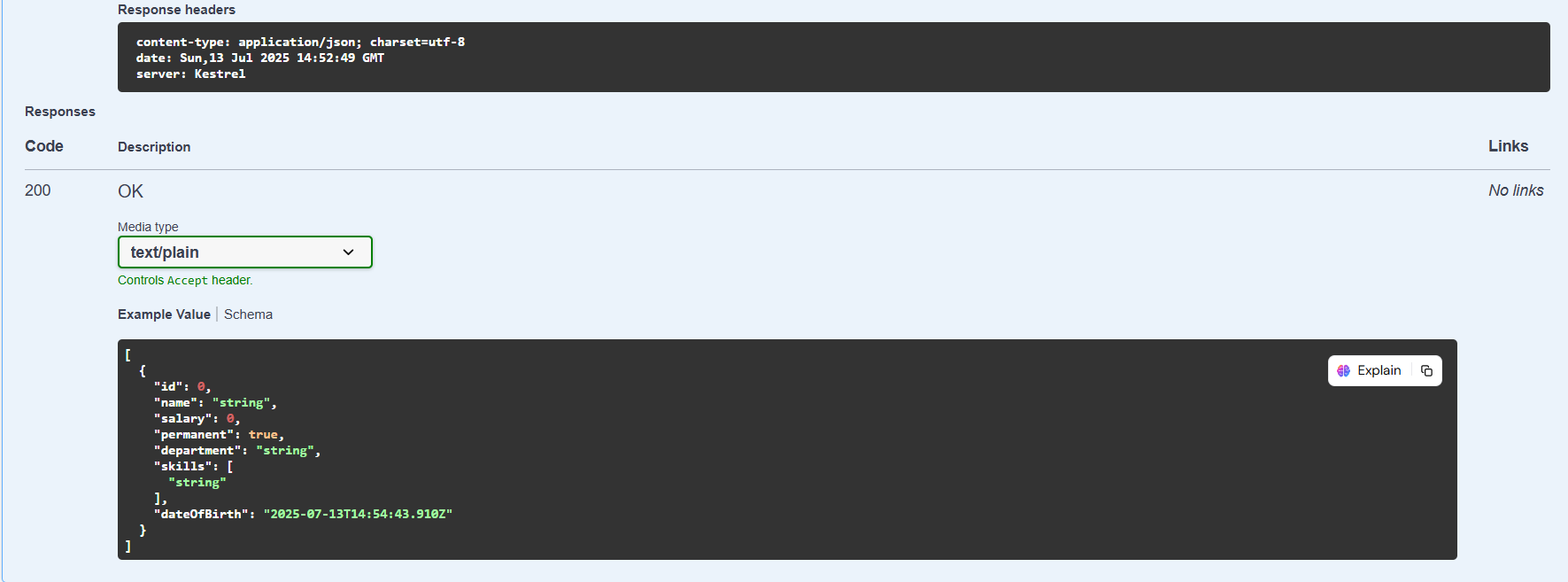
**OUTPUTS-->**

****

****

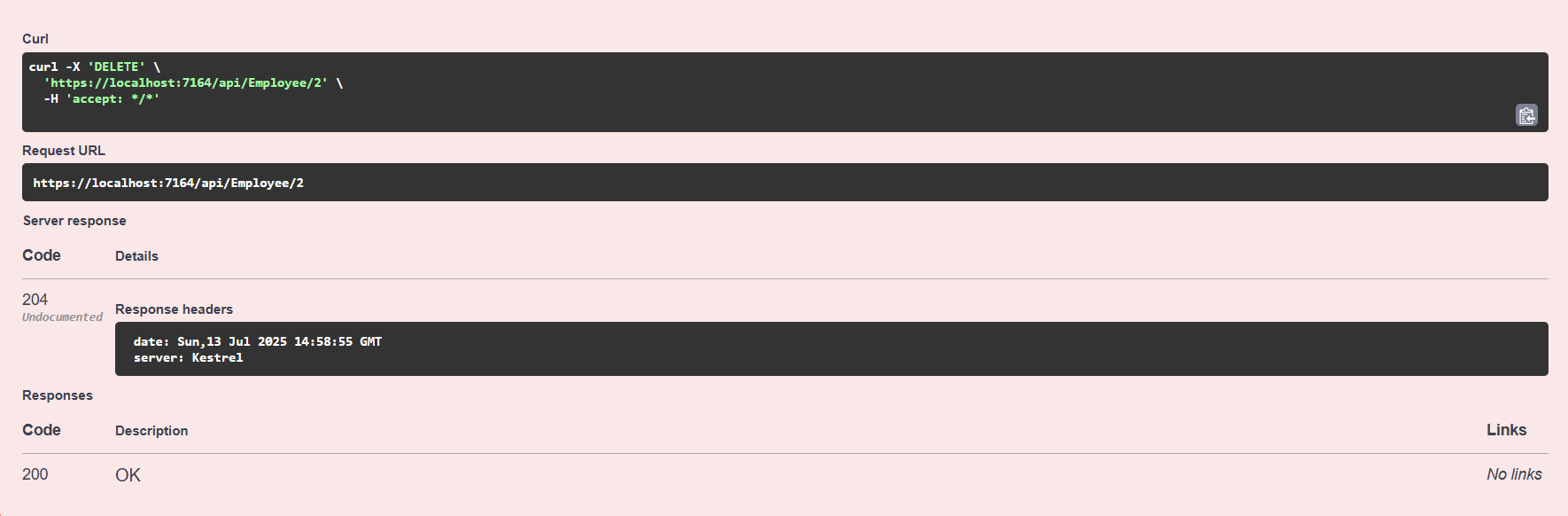
****

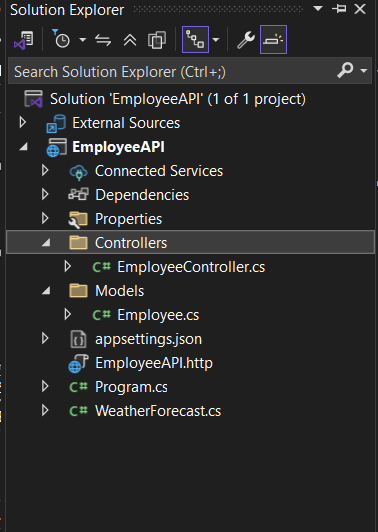
****

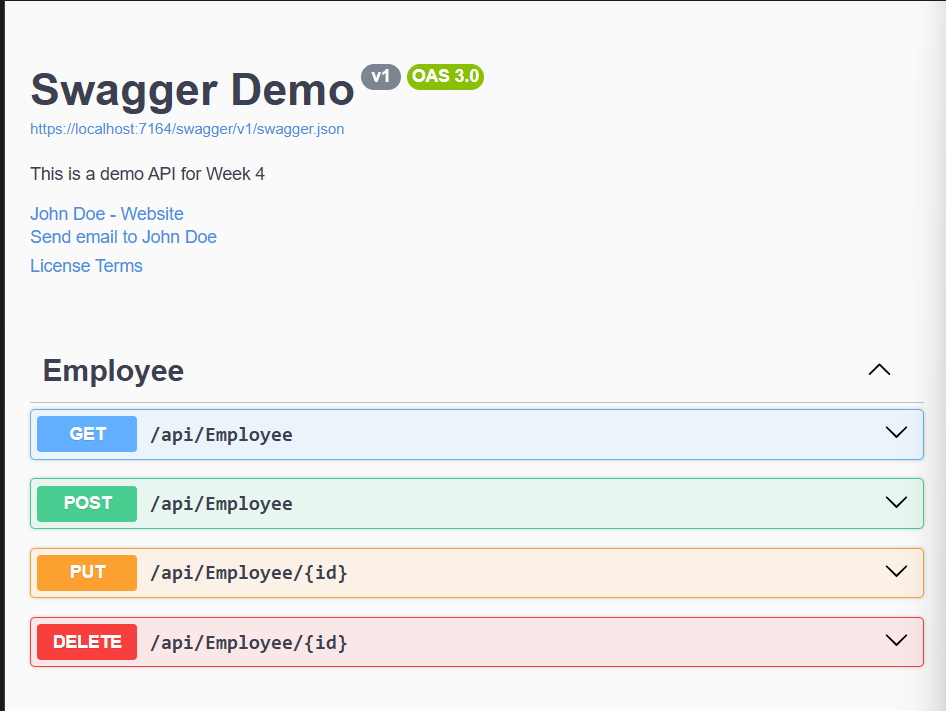
****

****

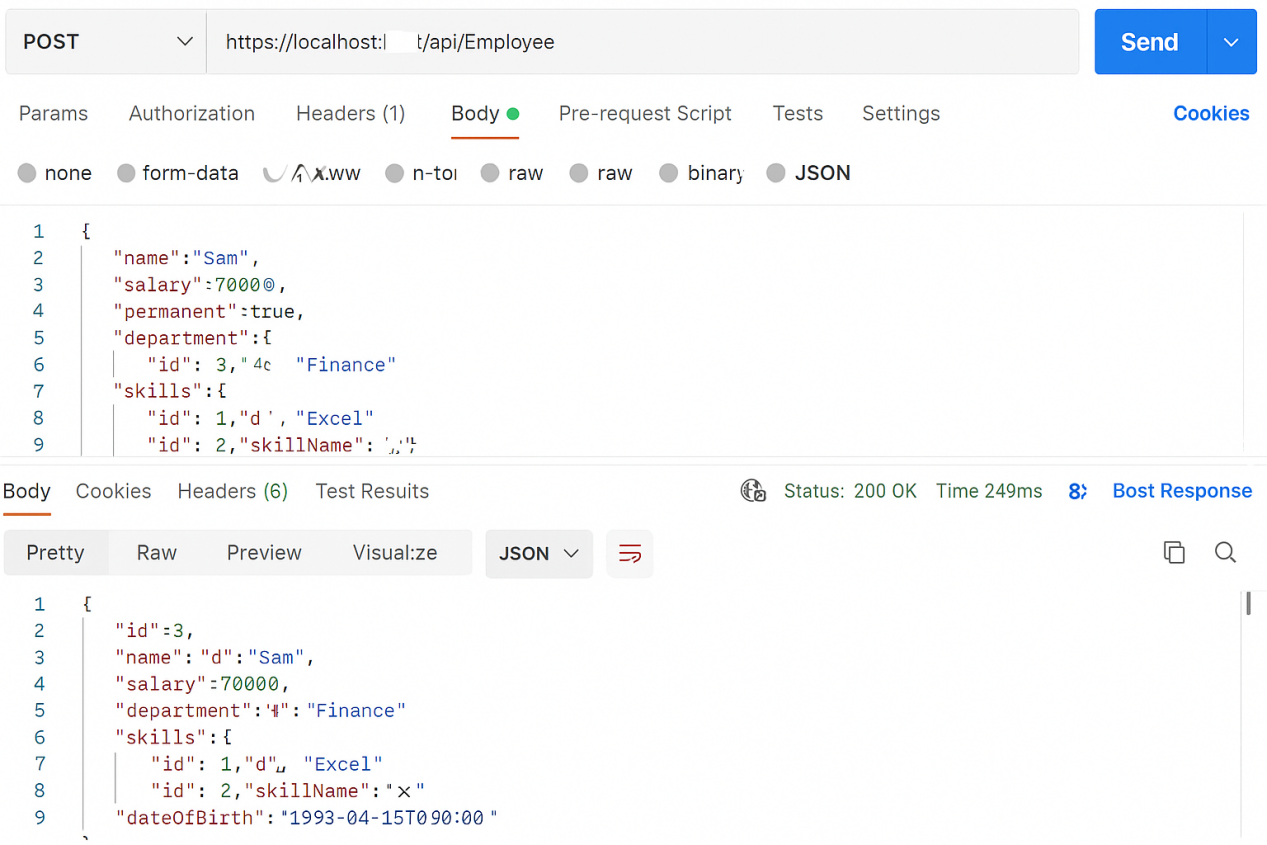
****

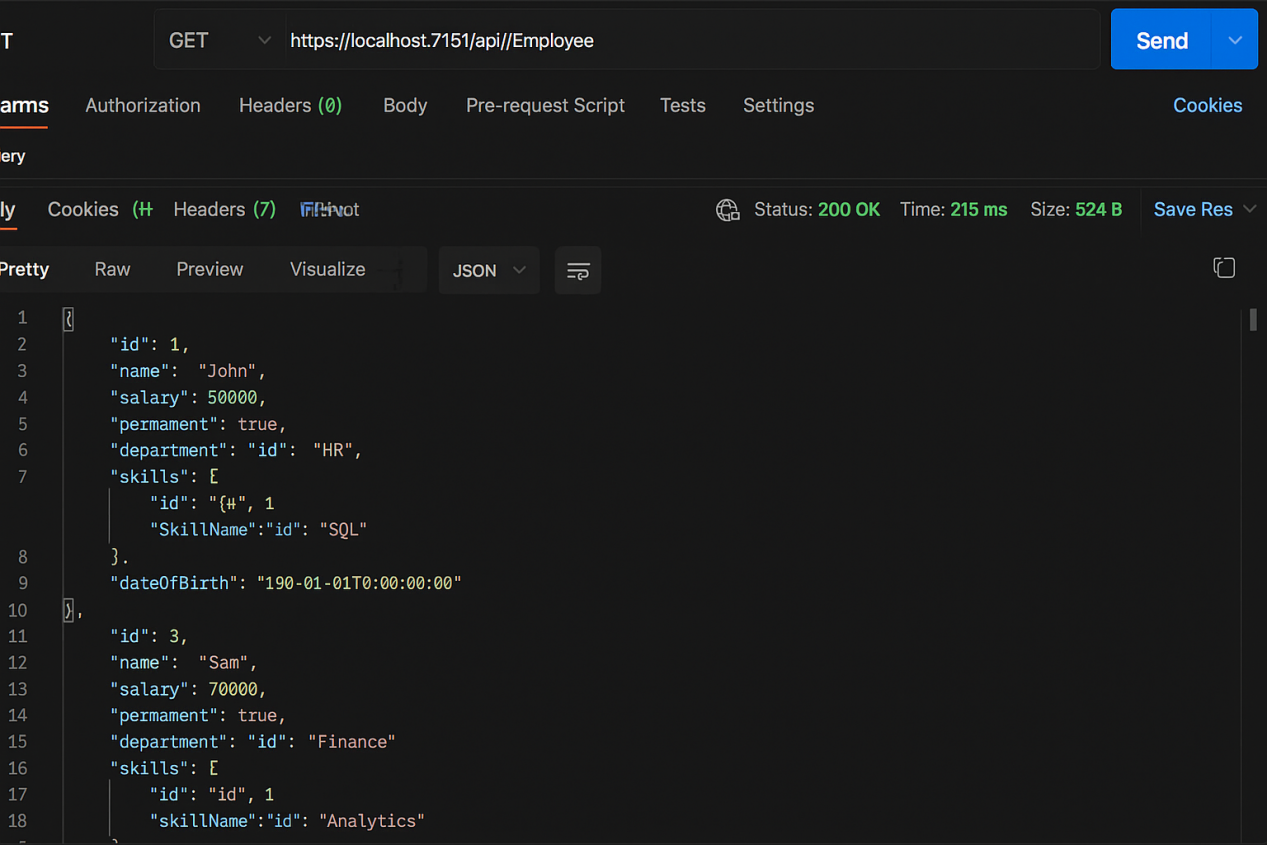
****

****

****

**TESTING ON POSTMAN->**

****

****

**Updated EmployeeController.cs**

using Microsoft.AspNetCore.Mvc;

using FirstWebAPI.NewFolder2;

namespace FirstWebAPI.Controllers

{

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

[ApiController]

public class EmployeeController : ControllerBase

{

private static List<Employee> \_employees = new List<Employee>

{

new Employee

{

Id = 1,

Name = "John",

Salary = 50000,

Permanent = true,

Department = new Department { Id = 1, DeptName = "HR" },

Skills = new List<Skill>

{

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

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

},

DateOfBirth = new DateTime(1990, 1, 1)

},

new Employee

{

Id = 2,

Name = "Jane",

Salary = 60000,

Permanent = false,

Department = new Department { Id = 2, DeptName = "IT" },

Skills = new List<Skill>

{

new Skill { Id = 3, SkillName = "Java" }

},

DateOfBirth = new DateTime(1995, 5, 10)

}

};

// GET api/employee

[HttpGet]

public ActionResult<List<Employee>> Get()

{

return Ok(\_employees);

}

// POST api/employee

[HttpPost]

public ActionResult<Employee> Post([FromBody] Employee newEmp)

{

newEmp.Id = \_employees.Max(e => e.Id) + 1;

\_employees.Add(newEmp);

return Ok(newEmp);

}

// PUT api/employee/{id}

[HttpPut("{id}")]

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

{

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

if (emp == null) return NotFound();

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);

}

// DELETE api/employee/{id}

[HttpDelete("{id}")]

public ActionResult Delete(int id)

{

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

if (emp == null) return NotFound();

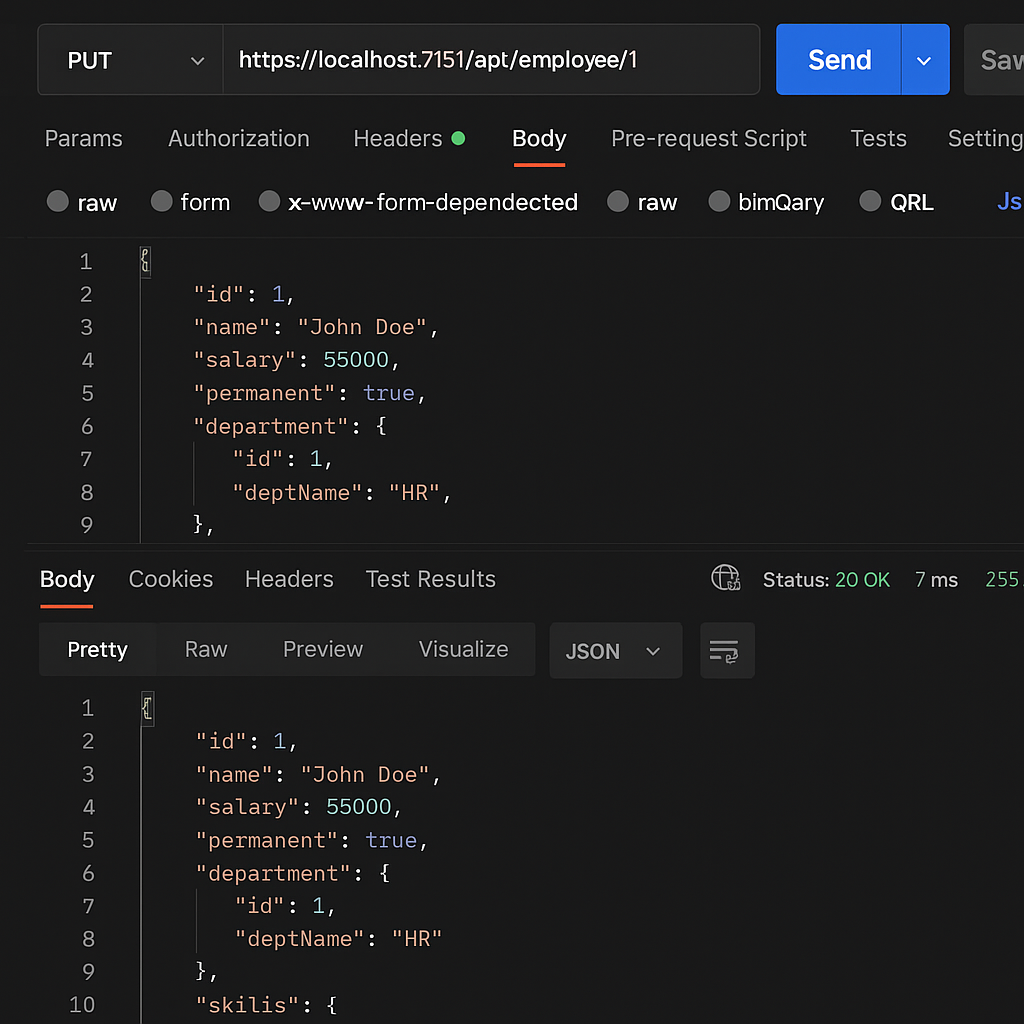
\_employees.Remove(emp);

return NoContent();

}

}

}



---------------------------------------X--------------------------------------------