**WebAPI \_Day2\_Handson**

**Name:Asritha cherukuri**

**Date:06-07-2021**

**EmpID:916179**

Handson1

1)

Given:

Web Api using .Net core with Swagger

Create a .Net core web application with API template. (Use existing application if created). Install Swashbuckle.AspNetCore Nuget package. Post this do the following steps in Startup.cs

Implementation:

Startup.cs

using Microsoft.AspNetCore.Builder;

using Microsoft.AspNetCore.Hosting;

using Microsoft.AspNetCore.HttpsPolicy;

using Microsoft.AspNetCore.Mvc;

using Microsoft.Extensions.Configuration;

using Microsoft.Extensions.DependencyInjection;

using Microsoft.Extensions.Hosting;

using Microsoft.Extensions.Logging;

using Microsoft.OpenApi.Models;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Threading.Tasks;

namespace SecondAPI

{

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 = "SecondAPI", Version = "v1" });

});\*/

services.AddSwaggerGen(c =>

{

c.SwaggerDoc("v1", new OpenApiInfo

{

Title = "Swagger Demo",

Version = "v1",

Description = "TBD",

TermsOfService = new Uri("http://www.google.com"),

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

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

});

});

}

// 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", "SecondAPI v1"));

}

app.UseSwagger();

app.UseSwaggerUI(c =>

{

// specifying the Swagger JSON endpoint.

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

});

app.UseHttpsRedirection();

app.UseRouting();

app.UseAuthorization();

app.UseEndpoints(endpoints =>

{

endpoints.MapControllers();

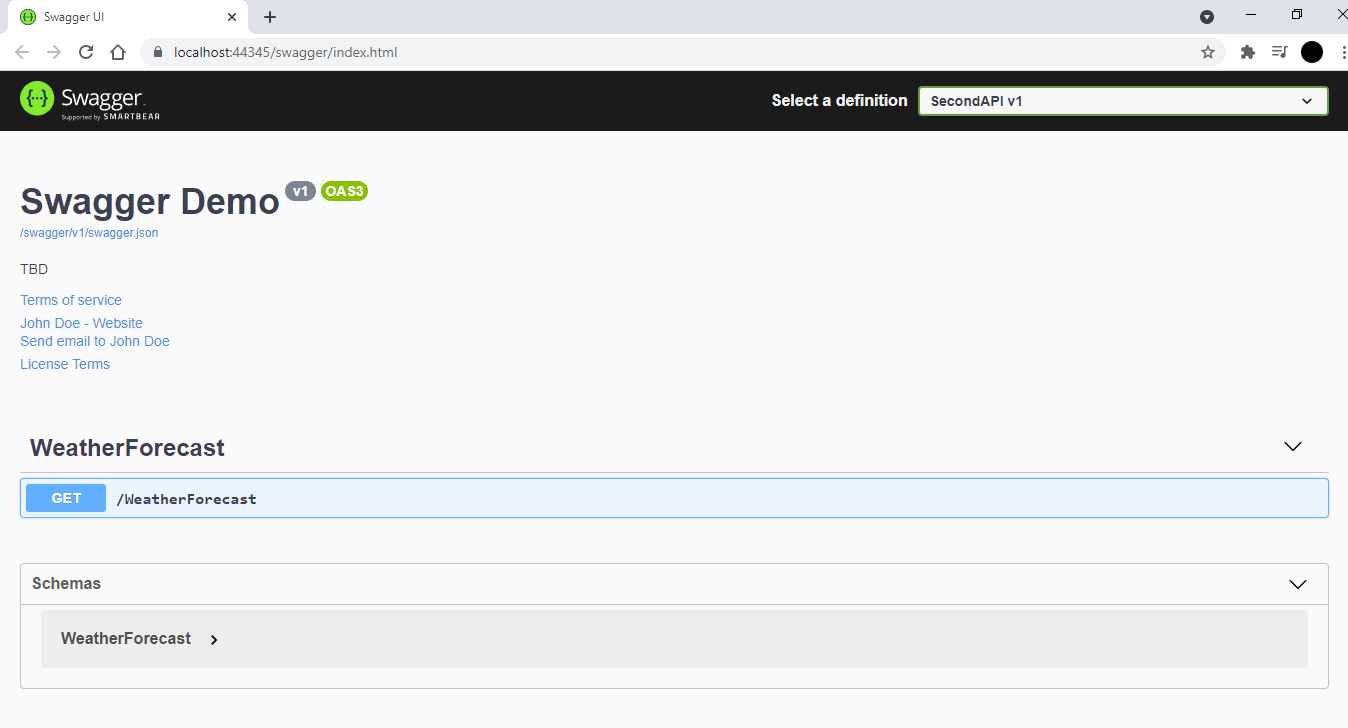
});

}

}

}

Output:



2)

Given:

Use POSTMAN tool, to point to the local Web API that was created with Employee controller. Test the GET action method using POSTMAN.

Verify the output if the List of employees are listed in the ‘Body’ part of the GET window on POSTMAN tool.

Implementation:

EmployeeController.cs

using Microsoft.AspNetCore.Mvc;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Threading.Tasks;

using SecondAPI.Data.Models;

using Microsoft.AspNetCore.Http;

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

namespace SecondAPI.Controllers

{

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

[ApiController]

public class EmployeeController : ControllerBase

{

private List<Employee> Employees;

public EmployeeController() => Employees = new List<Employee>()

{

new Employee() { Id= 1, Name="Asritha",Salary=30000,Permanent=true, Department= new Department( 1,"CDE") ,Skills=new List<Skills>(){ new Skills ( "java",1), new Skills("c#",4 ) }, DateOfBirth=new DateTime(1999,10,26) },

new Employee() { Id= 2, Name="Harika",Salary=25000,Permanent=true, Department= new Department( 2,"IT") ,Skills=new List<Skills>(){ new Skills ( "python",2), new Skills("ML",3 ) }, DateOfBirth=new DateTime(1997,06,09) },

new Employee() { Id= 3, Name="Anvitha",Salary=20000,Permanent=true, Department= new Department( 3,"CSD") ,Skills=new List<Skills>(){ new Skills ( "python",2), new Skills("c#",4 ) }, DateOfBirth=new DateTime(1998,12,03) }

};

private List<Employee> GetStandardEmployeeList()

{

return Employees;

}

[HttpGet]

[ProducesResponseType(StatusCodes.Status200OK)]

public IEnumerable<Employee> Get()

{

return GetStandardEmployeeList();

}

[HttpGet("Employee/getstandard")]

public ActionResult<Employee> GetStandard()

{

return Employees.Where(i => i.Permanent == true).FirstOrDefault();

}

Employee.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Threading.Tasks;

namespace SecondAPI.Data.Models

{

public class Employee

{

public Employee() { }

public Employee(int id,string name,int salary,bool permanent,Department dept,List<Skills>skill,DateTime dt)

{

this.Id = id;

this.Name = name;

this.Salary = salary;

this.Permanent = permanent;

this.Department = dept;

this.Skills = skill;

this.DateOfBirth = dt;

}

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<Skills> Skills { get; set; }

public DateTime DateOfBirth { get; set; }

}

}

Skills.cs:

using System;

using System.Collections.Generic;

using System.Linq;

using System.Threading.Tasks;

namespace SecondAPI.Data.Models

{

public class Skills

{

public Skills(string sname, int sid)

{

this.Sname = sname;

this.Sid = sid;

}

public string Sname{get;set;}

public int Sid { get; set; }

}

}

Department.cs:

namespace SecondAPI.Data.Models

{

public class Department

{

private int id;

private string name;

public Department(int id, string name)

{

this.id = DeptId;

this.name = DeptName;

}

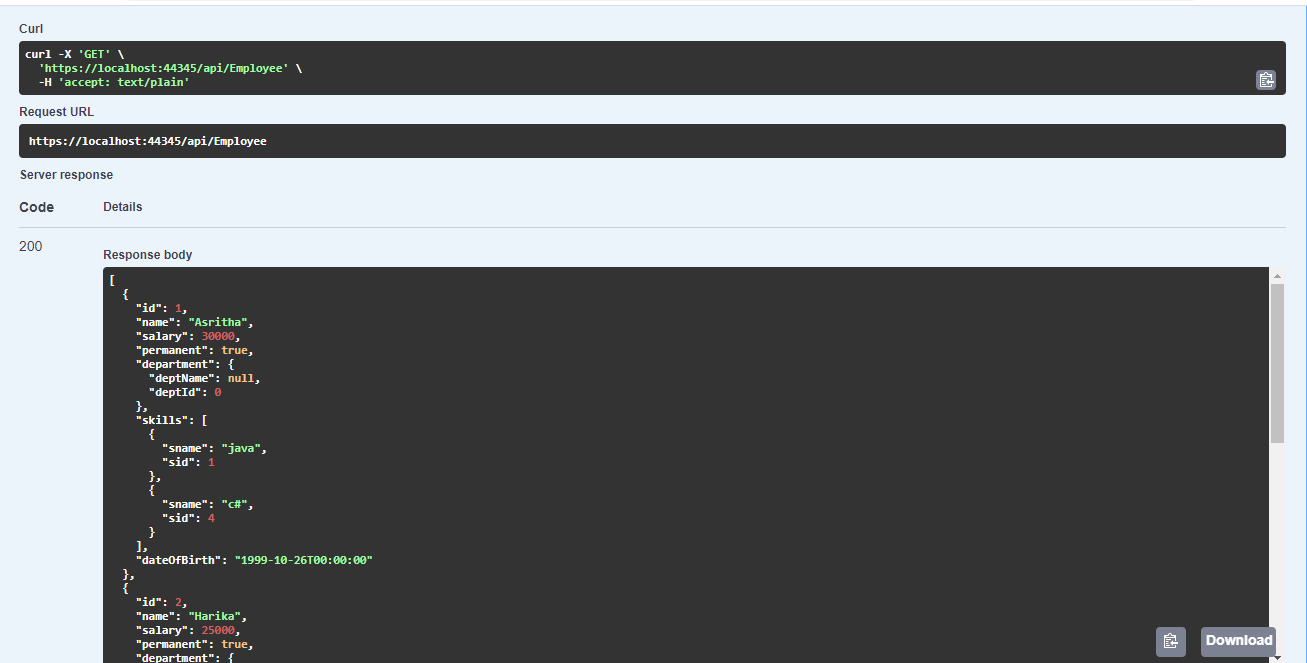
public string DeptName { get; set; }

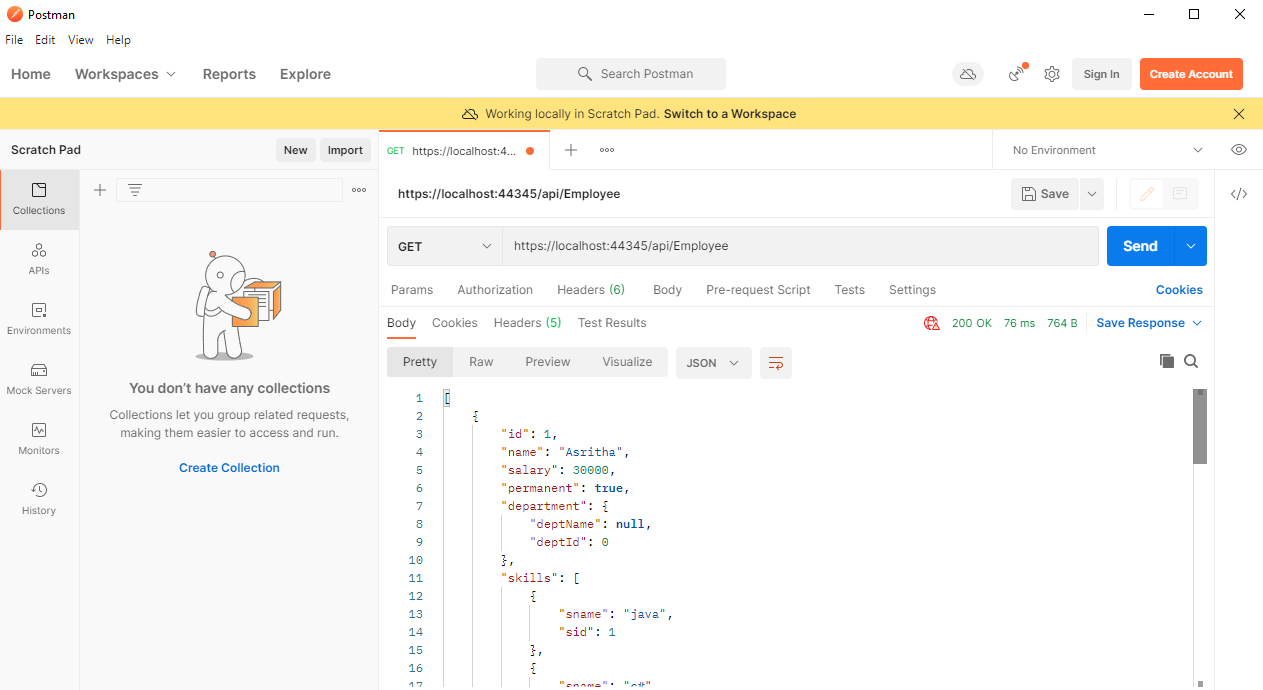
public int DeptId { get; set; }

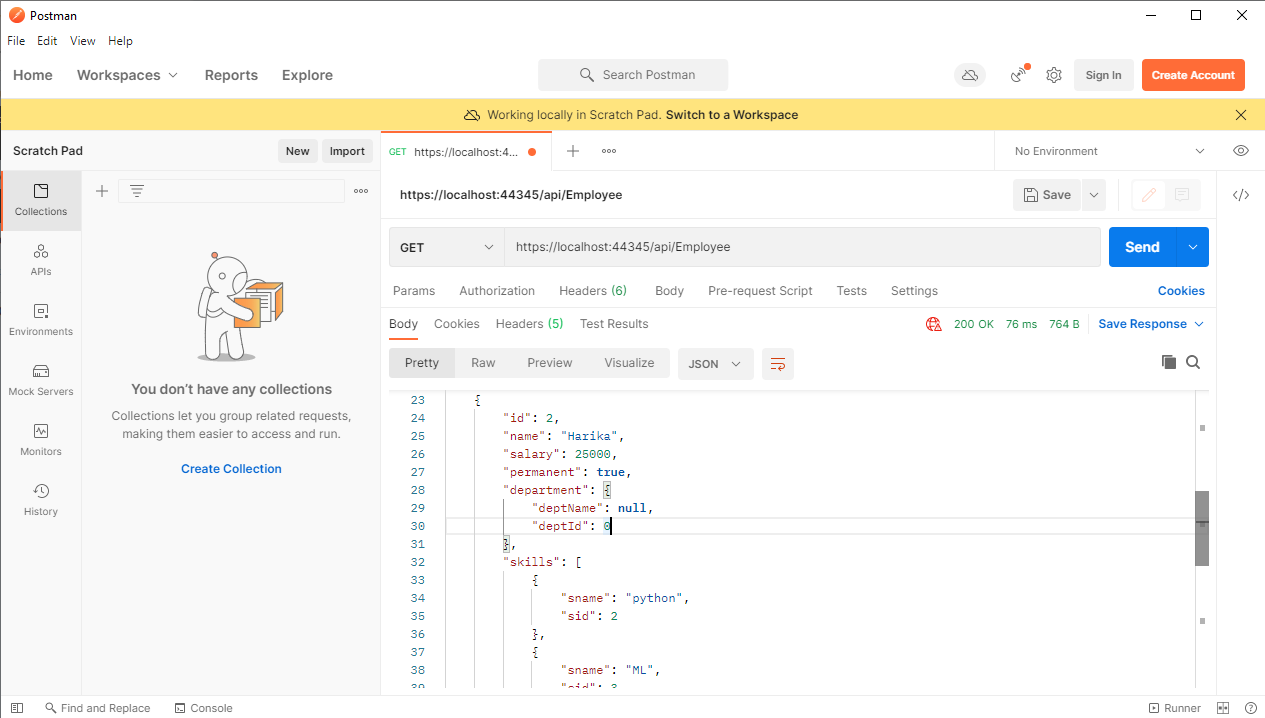
}

}

Output:





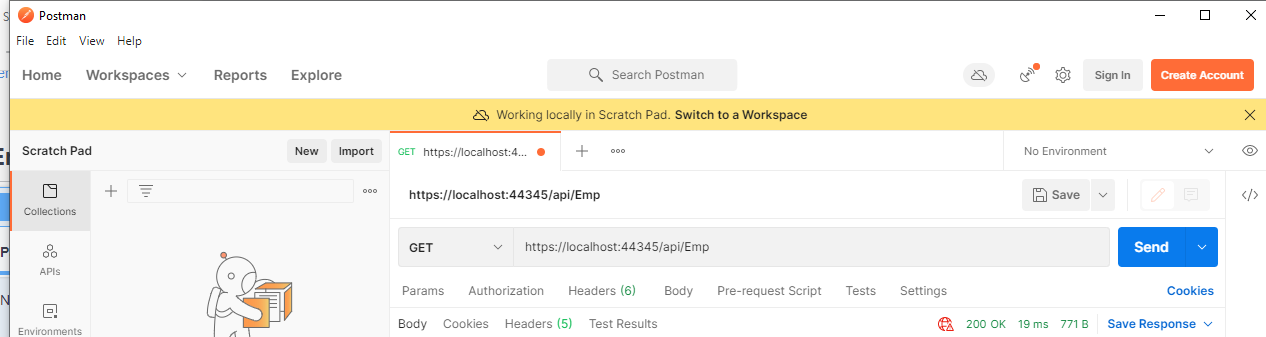


3)

Given:

Modify the Controller name in the Route attribute of the Employee controller to ‘Emp’ and check its access thru POSTMAN

Output:



===================================================================

Handson2

Given:

. Web Api using custom model class

Implementation:

IEmployee.cs:

IEnumerable<Emp> GetStandardEmployeeList();

ActionResult<Emp> GetStandard();

Emp GetEmployeeById(int id);

EmployeeTest.cs:

public IEnumerable<Emp> GetStandardEmployeeList()

{

return Employees;

}

public ActionResult<Emp> GetStandard()

{

return Employees.Where(i => i.Permanent == true).FirstOrDefault();

}

public Emp GetEmployeeById(int id)

{

Emp em = null;

em = Employees.Find(e => e.id == id);

return em;

}

Emp.cs:

public class Emp

{

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; }

EmployeeController.cs:

[HttpGet]

[ProducesResponseType(StatusCodes.Status200OK)]

[ProducesResponseType(StatusCodes.Status204NoContent)]

[ProducesResponseType(StatusCodes.Status404NotFound)]

[ProducesResponseType(StatusCodes.Status400BadRequest)]

public IEnumerable<Emp> Get() => emptest.GetStandardEmployeeList();

[HttpGet("Employee/getstandard")]

public ActionResult<Emp> GetStandard()

{

return emptest.GetStandard();

}

[HttpGet("{id}")]

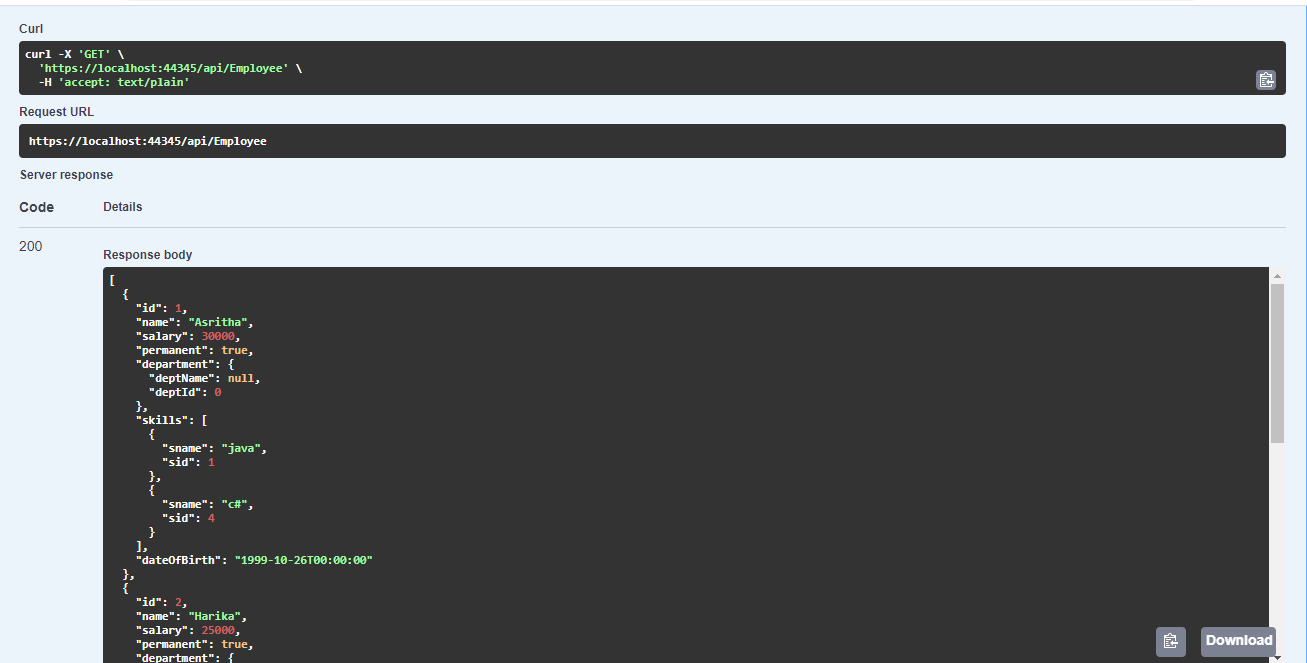
public Emp GetEmployeeById(int id)

{

return emptest.GetEmployeeById(id);

}

Output:



===================================================================

Handson3

Given:

. Update Employee data as per the input thru Web API PUT action method call

Implementation:

EmployeeController.cs

[HttpPut("{id}")]

public void Put(int id, [FromBody] Emp em)

{

emptest.updateEmployee(id,em);

}

[HttpDelete("{id}")]

public void Delete(int id)

{

emptest.RemoveEmployee(id);

}

IEmployee.cs:

namespace Employee.Data.models

{

interface IEmployee

{

IEnumerable<Emp> GetStandardEmployeeList();

ActionResult<Emp> GetStandard();

Emp GetEmployeeById(int id);

void AddEmployee(Emp emp);

void updateEmployee(int empid, Emp em);

void RemoveEmployee(int empid);

}

}

EmployeeTest.cs:

public void updateEmployee(int empid, Emp em)

{

Emp emp = null;

emp = Employees.Find(e => e.id == empid);

if (emp != null)

{

emp.id = em.id;

emp.Name = em.Name;

emp.Permanent = em.Permanent;

emp.Salary = em.Salary;

emp.Skills = em.Skills;

emp.Department = em.Department;

emp.DateOfBirth = em.DateOfBirth;

}

}

public void RemoveEmployee(int empid)

{

Emp emp = null;

emp = Employees.Find(e => e.id == empid);

if (emp != null)

{

Employees.Remove(emp);

}

}

}

}

Emp.cs:

using System;

using System.Collections.Generic;

using System.Linq;

using System.Threading.Tasks;

namespace Employee.Data.models

{

public class Emp

{

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; }

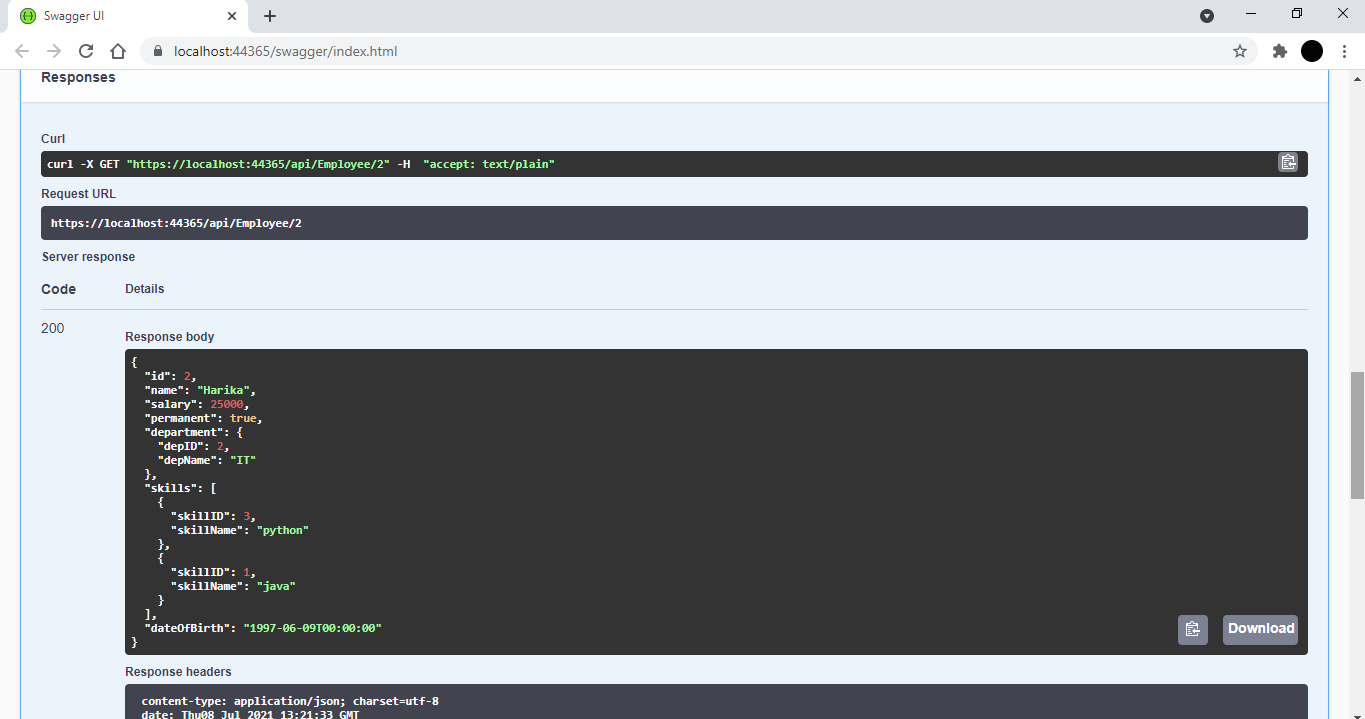
public DateTime DateOfBirth { get; set; }

}

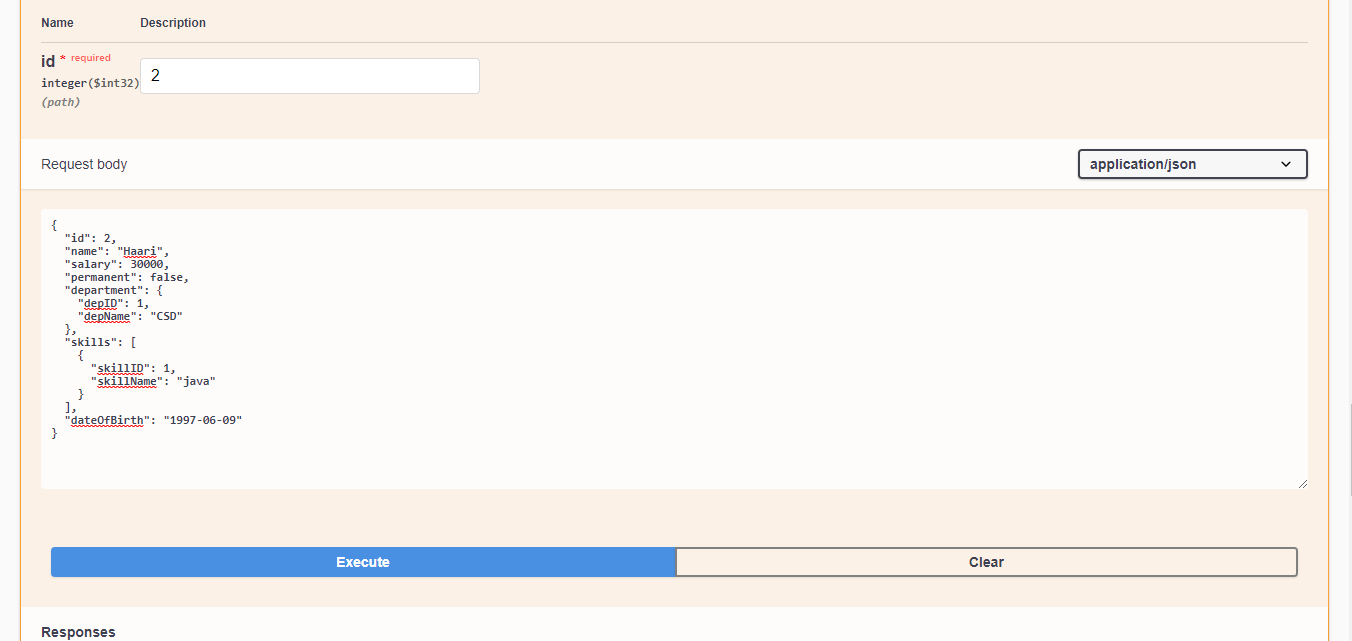
}

Output:

EmpId2 before Updating



Update



EmpId2 After Updating

