Superset id-6361766

Week-4

**ASP .NET Core :-**

1. **First Web Api using .Net core**

Create a .Net core web application with API template. Use the option to create controller with Read Write permissions. Notice the ValuesController creation with Action methods corresponding to the Action verbs.

On creation of the Web API, execute the application and check if the GET action method result is returned as expected.

**Solution:-**

**Program.cs**

using Microsoft.OpenApi.Models;

var builder = WebApplication.CreateBuilder(args);

// Add services to the container.

builder.Services.AddControllers(); // ✅ Add controllers support

builder.Services.AddEndpointsApiExplorer(); // For Swagger

builder.Services.AddSwaggerGen(); // For Swagger

var app = builder.Build();

// Configure the HTTP request pipeline.

if (app.Environment.IsDevelopment())

{

app.UseSwagger();

app.UseSwaggerUI();

}

else

{

app.UseExceptionHandler("/Error");

app.UseHsts();

}

app.UseHttpsRedirection();

app.UseStaticFiles();

app.UseRouting();

app.UseAuthorization();

app.MapControllers(); // ✅ Map API controllers

// app.MapRazorPages(); // Optional: keep if you still use Razor Pages

app.Run();

**Valuecontroller.cs:-**

using Microsoft.AspNetCore.Mvc;

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

namespace FirstWebApi.Controllers

{

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

[ApiController]

public class ValuesController : ControllerBase

{

// GET: api/<ValuesController>

[HttpGet]

public IEnumerable<string> Get()

{

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

}

// GET api/<ValuesController>/5

[HttpGet("{id}")]

public string Get(int id)

{

return "value";

}

// POST api/<ValuesController>

[HttpPost]

public void Post([FromBody] string value)

{

}

// PUT api/<ValuesController>/5

[HttpPut("{id}")]

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

{

}

// DELETE api/<ValuesController>/5

[HttpDelete("{id}")]

public void Delete(int id)

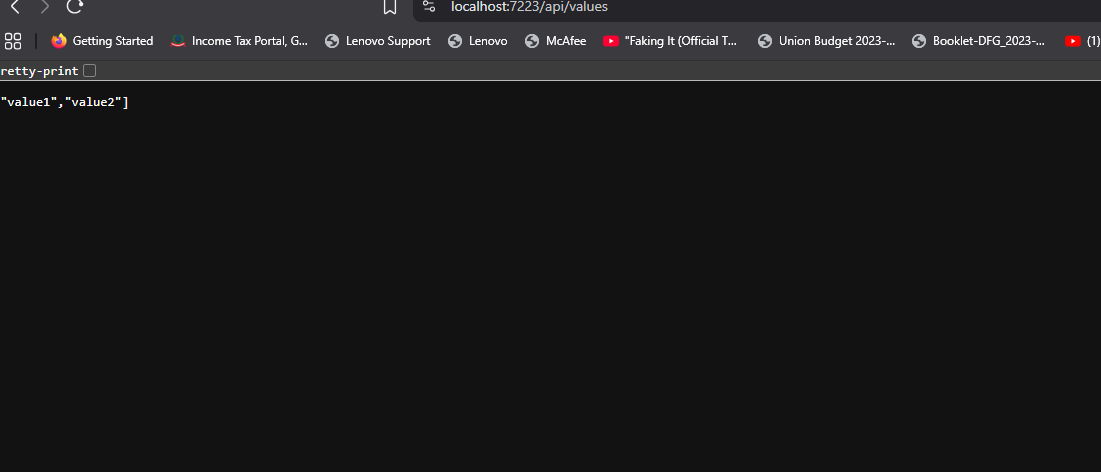
{

}

}

}

**Output:-**



1. **Web Api using .Net core with Swagger**

1.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

2.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.

Verify the Status on the right side of the output pane on POSTMAN tool.

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

**Program.cs:-**

using Microsoft.OpenApi.Models;

var builder = WebApplication.CreateBuilder(args);

builder.Services.AddControllers();

builder.Services.AddEndpointsApiExplorer();

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://www.example.com")

},

License = new OpenApiLicense

{

Name = "License Terms",

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

}

});

});

var app = builder.Build();

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();

**EmployeeController.cs**

using Microsoft.AspNetCore.Mvc;

using System.Collections.Generic;

using System.Linq;

namespace FirstWebApi.Controllers

{

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

[ApiController]

public class EmployeeController : ControllerBase

{

// In-memory list of employees (simulating a database)

private static List<Employee> employees = new List<Employee>

{

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

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

};

// GET: api/Employee

[HttpGet]

public ActionResult<IEnumerable<Employee>> Get()

{

return Ok(employees);

}

// GET: api/Employee/1

[HttpGet("{id}")]

public ActionResult<Employee> Get(int id)

{

var employee = employees.FirstOrDefault(e => e.Id == id);

if (employee == null)

return NotFound();

return Ok(employee);

}

// POST: api/Employee

[HttpPost]

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

{

newEmployee.Id = employees.Max(e => e.Id) + 1;

employees.Add(newEmployee);

return CreatedAtAction(nameof(Get), new { id = newEmployee.Id }, newEmployee);

}

// PUT: api/Employee/1

[HttpPut("{id}")]

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

{

var existing = employees.FirstOrDefault(e => e.Id == id);

if (existing == null)

return NotFound();

existing.Name = updatedEmployee.Name;

existing.Department = updatedEmployee.Department;

return NoContent();

}

// DELETE: api/Employee/1

[HttpDelete("{id}")]

public IActionResult Delete(int id)

{

var employee = employees.FirstOrDefault(e => e.Id == id);

if (employee == null)

return NotFound();

employees.Remove(employee);

return NoContent();

}

}

// Simple Employee model

public class Employee

{

public int Id { get; set; }

public string Name { get; set; }

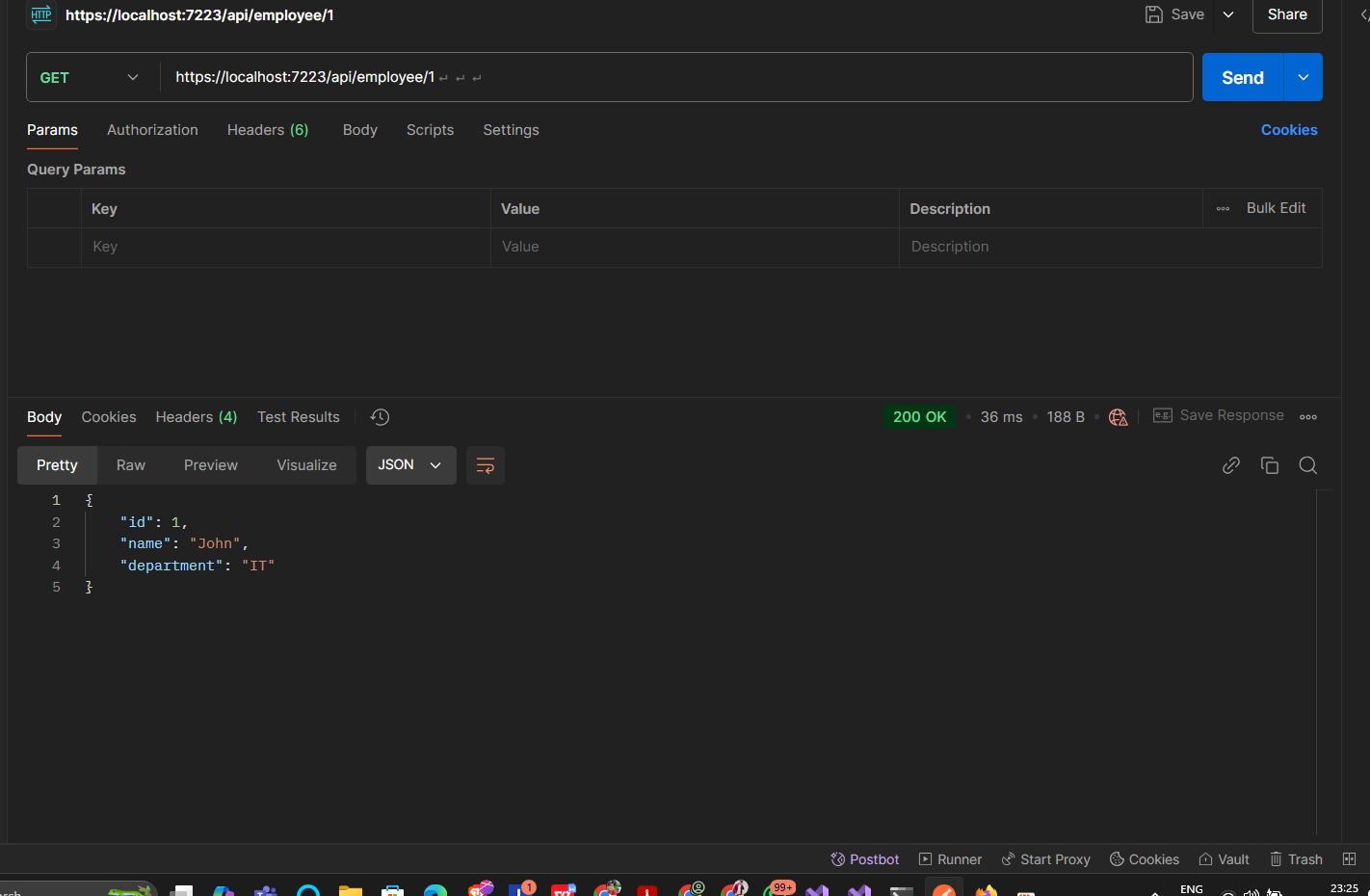
public string Department { get; set; }

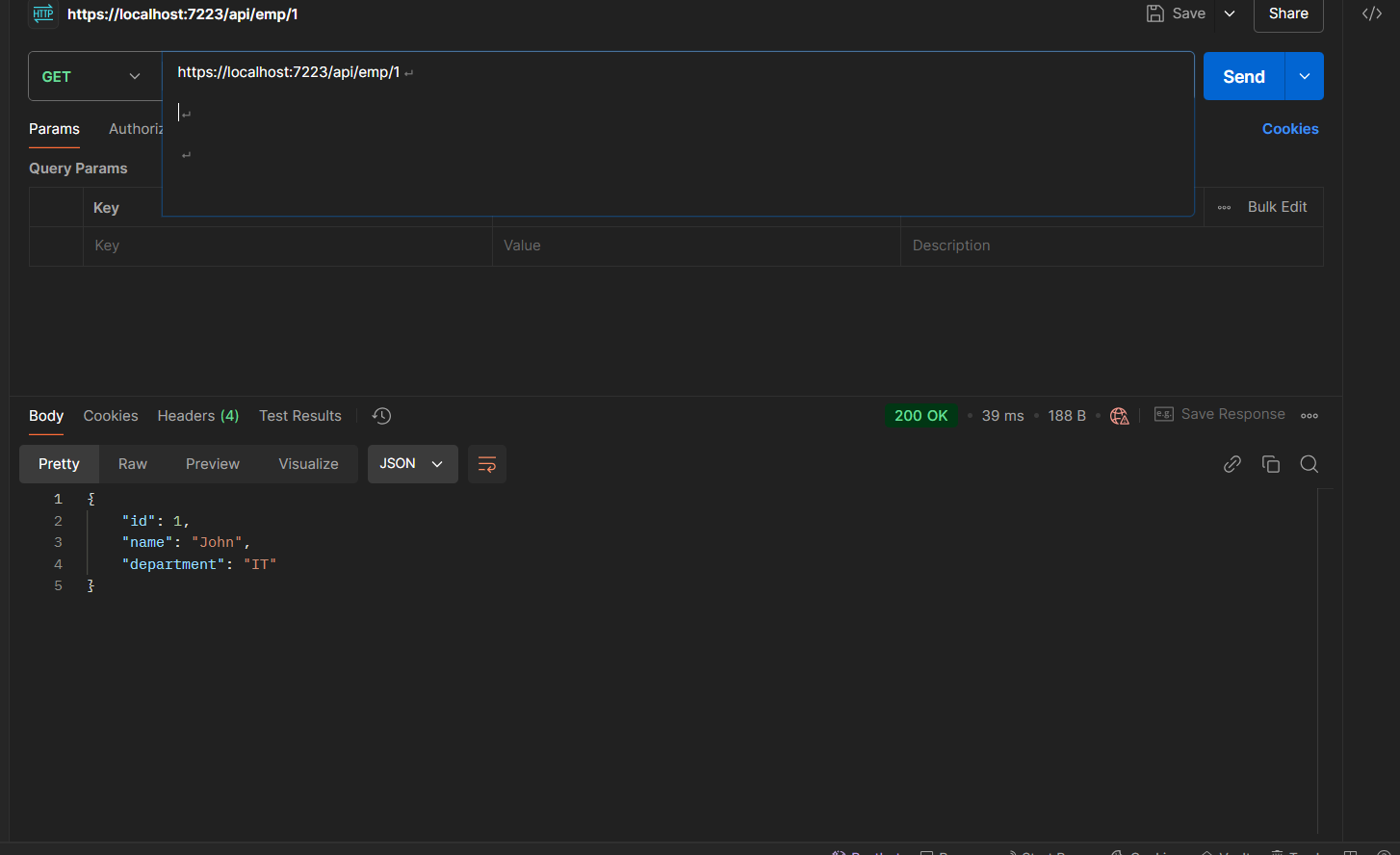
}

}

3.

Ouput:-





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

**Create a Custom action filter for Authorization.**

**Custom Exception filter**

Program.cs:-

using FirstWebApi.Filters;

using Microsoft.AspNetCore.Builder;

using Microsoft.Extensions.DependencyInjection;

using Microsoft.Extensions.Hosting;

using Microsoft.OpenApi.Models;

using FirstWebApi.Filters;

var builder = WebApplication.CreateBuilder(args);

// Add services to the container

builder.Services.AddControllers(options =>

{

// Register the global exception filter

options.Filters.Add<CustomExceptionFilter>();

});

// Add Swagger & Bearer token support

builder.Services.AddEndpointsApiExplorer();

builder.Services.AddSwaggerGen(c =>

{

c.SwaggerDoc("v1", new OpenApiInfo

{

Title = "My API",

Version = "v1"

});

// Add Bearer token input in Swagger

c.AddSecurityDefinition("Bearer", new OpenApiSecurityScheme

{

In = ParameterLocation.Header,

Description = "Please enter 'Bearer' followed by your token",

Name = "Authorization",

Type = SecuritySchemeType.ApiKey,

Scheme = "Bearer"

});

c.AddSecurityRequirement(new OpenApiSecurityRequirement

{

{

new OpenApiSecurityScheme

{

Reference = new OpenApiReference

{

Type = ReferenceType.SecurityScheme,

Id = "Bearer"

}

},

new string[] {}

}

});

});

var app = builder.Build();

// Configure the HTTP request pipeline

if (app.Environment.IsDevelopment())

{

app.UseSwagger();

app.UseSwaggerUI();

}

app.UseHttpsRedirection();

// Use authorization middleware (needed if using [Authorize] or token checking)

app.UseAuthorization();

app.MapControllers();

app.Run();

Models.Employee.cs:-

using FirstWebApi.Models;

using System;

using System.Collections.Generic;

namespace FirstWebApi.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; }

}

}

**EmployeeController.cs**

using Microsoft.AspNetCore.Mvc;

using FirstWebApi.Models;

using System;

using System.Collections.Generic;

using FirstWebApi.Filters;

namespace FirstWebApi.Controllers

{

[CustomAuthFilter]

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

[ApiController]

public class EmployeeController : ControllerBase

{

private List<Employee> employees;

public EmployeeController()

{

employees = GetStandardEmployeeList();

}

// Private helper method

private List<Employee> GetStandardEmployeeList()

{

return new List<Employee>

{

new Employee

{

Id = 1,

Name = "John Doe",

Salary = 50000,

Permanent = true,

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

Skills = new List<Skill>

{

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

new Skill { Id = 2, Name = "ASP.NET Core" }

},

DateOfBirth = new DateTime(1990, 5, 20)

},

new Employee

{

Id = 2,

Name = "Jane Smith",

Salary = 40000,

Permanent = false,

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

Skills = new List<Skill>

{

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

},

DateOfBirth = new DateTime(1995, 8, 15)

}

};

}

[HttpGet]

[ProducesResponseType(typeof(List<Employee>), 200)]

[ProducesResponseType(500)]

public ActionResult<List<Employee>> Get()

{

throw new Exception("test exception from get method"); // assuming 'employees' is your in-memory list

}

[HttpGet("standard")]

public ActionResult<Employee> GetStandard()

{

return Ok(employees[0]);

}

[HttpPost]

public IActionResult Post([FromBody] Employee emp)

{

emp.Id = employees.Count + 1;

employees.Add(emp);

return CreatedAtAction(nameof(Get), new { id = emp.Id }, emp);

}

[HttpPut("{id}")]

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

{

var emp = employees.Find(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 NoContent();

}

}

}

**Filters.**

**CustomExceptionFilter.cs:-**

using Microsoft.AspNetCore.Mvc;

using Microsoft.AspNetCore.Mvc.Filters;

using System;

using System.IO;

namespace FirstWebApi.Filters

{

public class CustomExceptionFilter : IExceptionFilter

{

public void OnException(ExceptionContext context)

{

var logPath = Path.Combine(AppDomain.CurrentDomain.BaseDirectory, "error\_log.txt");

// Log exception details to file

File.AppendAllText(logPath,

$"{DateTime.Now}: {context.Exception.Message}{Environment.NewLine}{context.Exception.StackTrace}{Environment.NewLine}-----------------{Environment.NewLine}");

// Set 500 Internal Server Error response

context.Result = new ObjectResult("Internal Server Error. Please contact support.")

{

StatusCode = 500

};

context.ExceptionHandled = true;

}

}

}

**CustomAuthFilter.cs:-**

using Microsoft.AspNetCore.Mvc;

using Microsoft.AspNetCore.Mvc.Filters;

namespace FirstWebApi.Filters

{

public class CustomAuthFilter : ActionFilterAttribute

{

public override void OnActionExecuting(ActionExecutingContext context)

{

var hasAuthHeader = context.HttpContext.Request.Headers.TryGetValue("Authorization", out var authHeader);

if (!hasAuthHeader)

{

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

return;

}

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

{

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

return;

}

base.OnActionExecuting(context);

}

}

}

Output:-

