1. WebApi Handson

**Code :**

**Program.cs :**

using MyFirstWebAPI.Filters;

var builder = WebApplication.CreateBuilder(args);

builder.Services.AddScoped<CustomAuthFilter>();

builder.Services.AddControllers(options =>

{

options.Filters.Add<CustomExceptionFilter>();

});

builder.Services.AddEndpointsApiExplorer();

builder.Services.AddSwaggerGen();

var app = builder.Build();

// Middleware pipeline

if (app.Environment.IsDevelopment())

{

app.UseSwagger();

app.UseSwaggerUI();

}

app.UseAuthorization();

app.MapControllers();

app.Run();

**CustomerAuthFilter.cs :**

using Microsoft.AspNetCore.Mvc;

using MyFirstWebAPI.Filters;

using Microsoft.AspNetCore.Mvc.Filters;

namespace MyFirstWebAPI.Filters

{

public class CustomAuthFilter : ActionFilterAttribute

{

public override void OnActionExecuting(ActionExecutingContext context)

{

if (!context.HttpContext.Request.Headers.ContainsKey("Authorization"))

{

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

return;

}

var token = context.HttpContext.Request.Headers["Authorization"].ToString();

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

{

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

}

}

}

}

**CustomerExceptionFilter.cs**

using Microsoft.AspNetCore.Mvc;

using Microsoft.AspNetCore.Mvc.Filters;

using System.IO;

namespace MyFirstWebAPI.Filters

{

public class CustomExceptionFilter : IExceptionFilter

{

public void OnException(ExceptionContext context)

{

var exception = context.Exception;

Console.WriteLine($"Exception caught: {exception.Message}");

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

{

StatusCode = StatusCodes.Status500InternalServerError

};

}

}

}

**EmployeeController.cs**

using Microsoft.AspNetCore.Mvc;

using MyFirstWebAPI.Models;

using MyFirstWebAPI.Filters;

[ServiceFilter(typeof(CustomAuthFilter))]

[ApiController]

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

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 = "John",

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 = "SQL" }

},

DateOfBirth = new DateTime(1990, 01, 01)

}

};

}

[HttpGet]

[ProducesResponseType(StatusCodes.Status200OK)]

[ProducesResponseType(StatusCodes.Status500InternalServerError)]

public ActionResult<List<Employee>> GetStandard()

{

// Uncomment this line to test custom exception

// throw new Exception("Test Exception");

throw new Exception("Simulated exception for testing");

//return Ok(\_employees);

}

[HttpPost]

public IActionResult AddEmployee([FromBody] Employee employee)

{

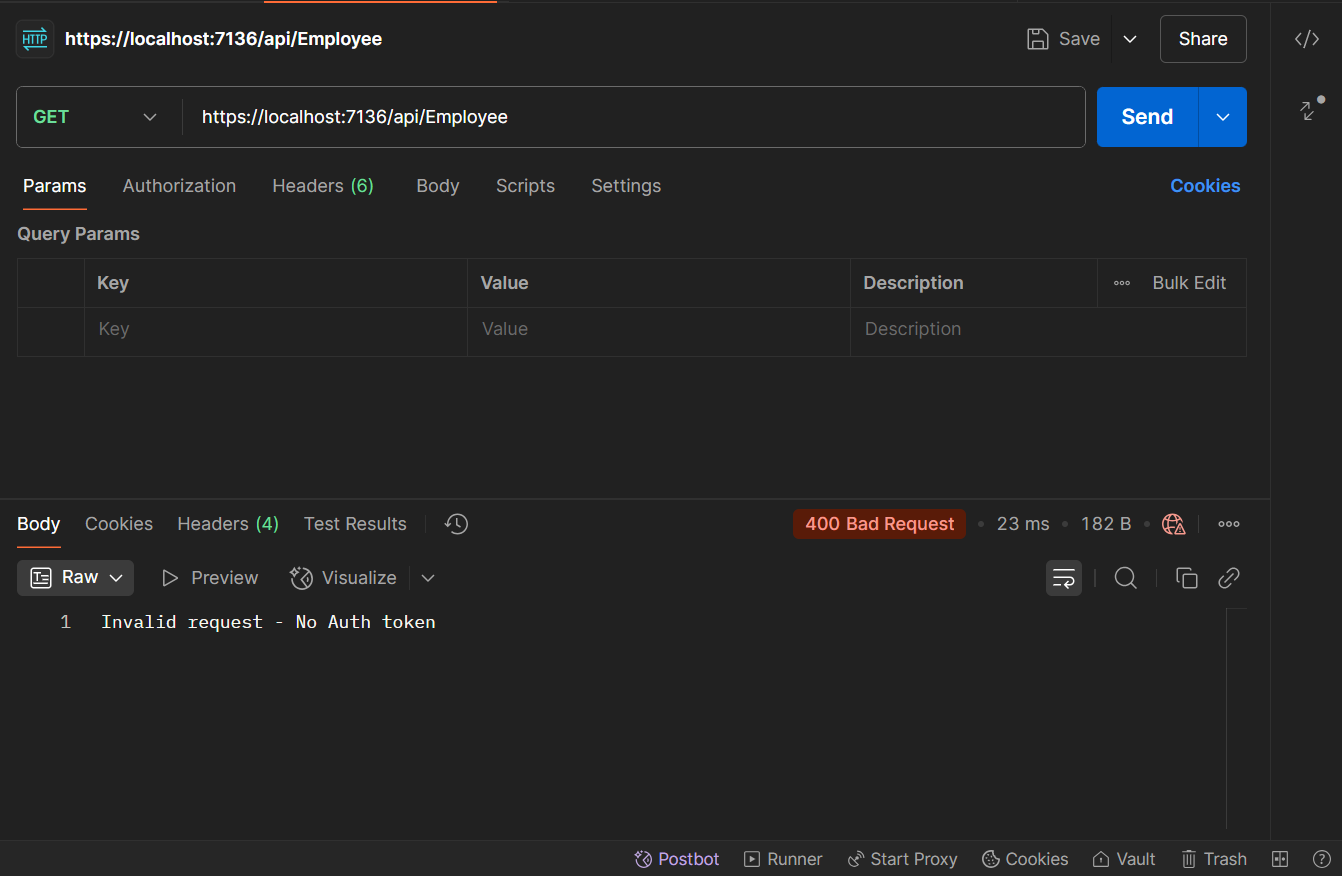
\_employees.Add(employee);

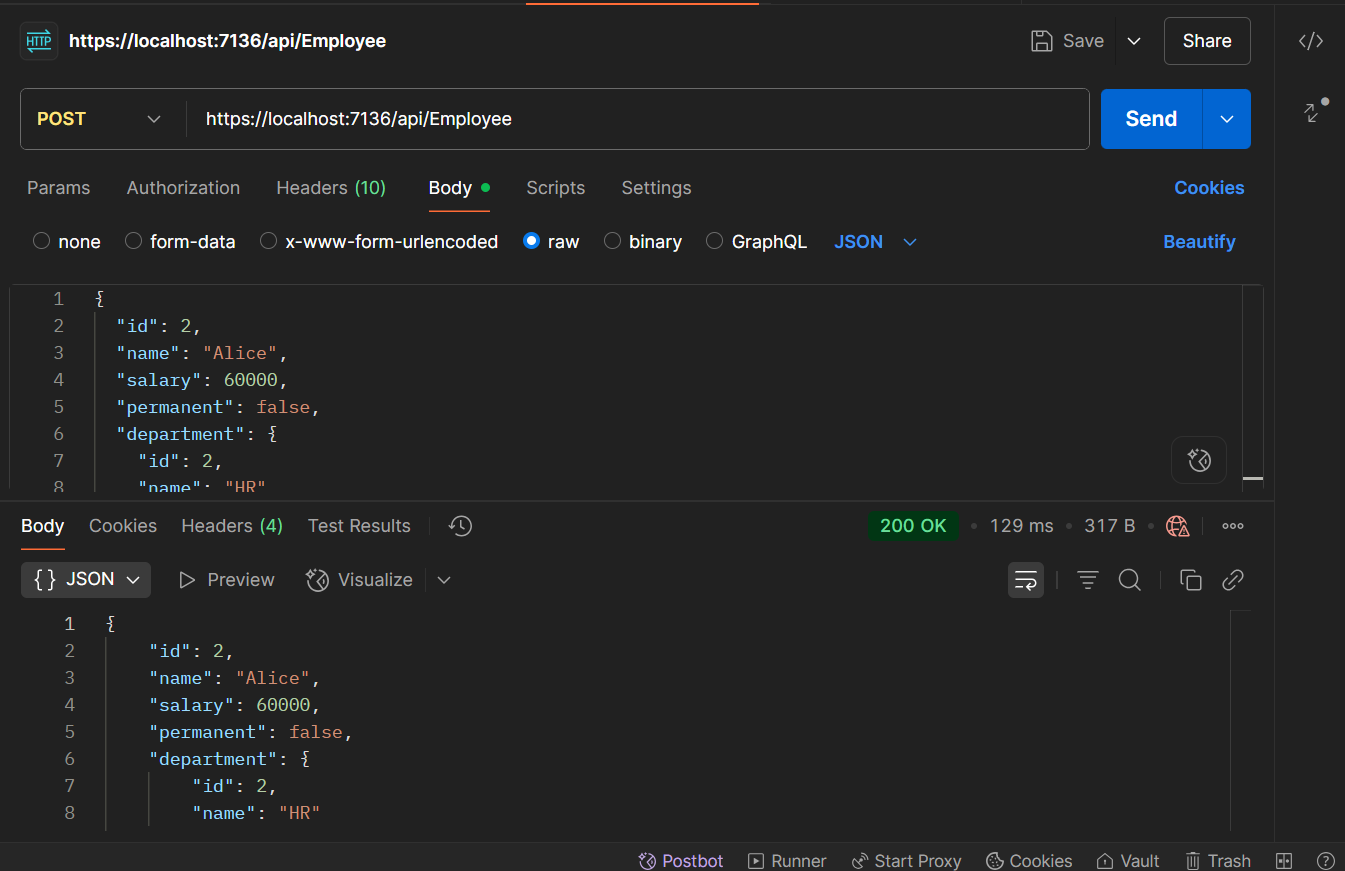
return Ok(employee);

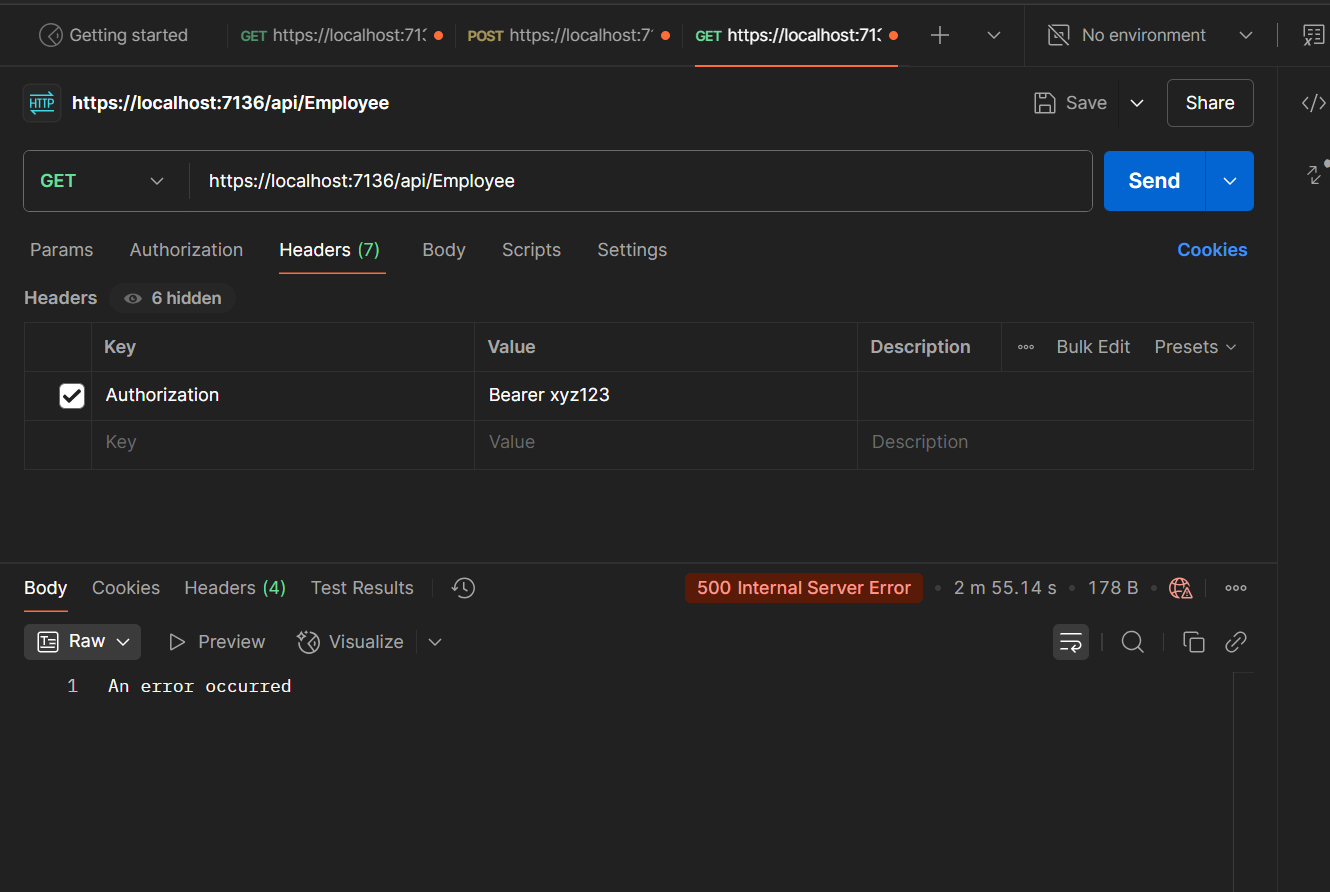
}

}

**Output :**

****

****

****