**3. WebApi\_Handson**

**Objectives:**1. Demonstrate creation of an Action method to return list of custom class entity

To return a list of custom class entities, first define a model class representing the data structure. Then, in the controller, use an action method with the [HttpGet] and [AllowAnonymous] attributes to expose the data. The method should return a list of the custom class using an IEnumerable<T> or List<T>. This allows public access to the endpoint without requiring authentication.

2. Explain the usage of FromBody attribute

The [FromBody] attribute is used in Web API to bind complex model types from the request body instead of query strings or route parameters. When an HTTP POST or PUT request is sent with JSON data, [FromBody] helps deserialize that JSON into a C# object. This is particularly useful for handling form submissions, login data, or any structured data passed in the body of the request.

3. Demonstrate Custom filter

A custom filter in Web API can be implemented by creating a class that inherits from ActionFilterAttribute. Override the OnActionExecuting method to intercept and manipulate the request before it reaches the action method. This can be used for logging, validation, or injecting headers. For exception handling, you can create a custom exception filter. To support compatibility with older MVC/Web API components, install the Microsoft.AspNetCore.Mvc.WebApiCompatShim package.

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

**Models Folder**  
**Department.cs**

namespace EmployeeWebApi.Models

{

public class Department

{

public int Id { get; set; }

public string Name { get; set; }

}

}  
**Skill.cs**  
namespace EmployeeWebApi.Models

{

public class Skill

{

public int Id { get; set; }

public string Name { get; set; }

}

}  
**Employee.cs**  
using System;

using System.Collections.Generic;

namespace EmployeeWebApi.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 Microsoft.AspNetCore.Authorization;

using EmployeeWebApi.Models;

using EmployeeWebApi.Filters;

using System;

using System.Collections.Generic;

namespace EmployeeWebApi.Controllers

{

[ApiController]

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

[AllowAnonymous] // Allows access without authentication

[TypeFilter(typeof(CustomAuthFilter))] // Custom Authorization Filter

public class EmployeeController : ControllerBase

{

// Private method to create sample data

private List<Employee> GetStandardEmployeeList()

{

return new List<Employee>

{

new Employee

{

Id = 1,

Name = "Varsha",

Salary = 55000,

Permanent = true,

Department = new Department

{

Id = 101,

Name = "Software"

},

Skills = new List<Skill>

{

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

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

},

DateOfBirth = new DateTime(2002, 10, 12)

},

new Employee

{

Id = 2,

Name = "John",

Salary = 47000,

Permanent = false,

Department = new Department

{

Id = 102,

Name = "Testing"

},

Skills = new List<Skill>

{

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

},

DateOfBirth = new DateTime(1998, 4, 25)

}

};

}

[HttpGet]

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

[ProducesResponseType(500)]

public ActionResult<List<Employee>> GetStandard()

{

var employeeList = GetStandardEmployeeList();

return Ok(employeeList);

}

[HttpPost]

public IActionResult Post([FromBody] Employee emp)

{

if (emp == null)

{

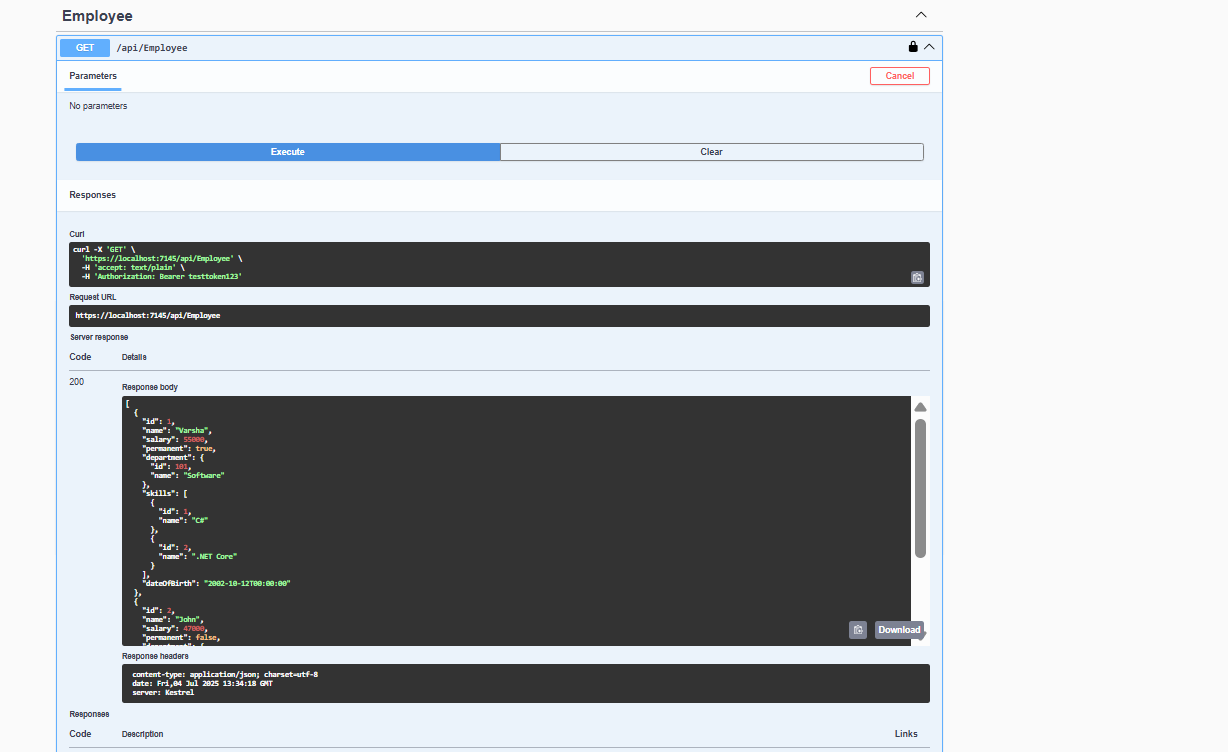
return BadRequest("Employee object is null.");

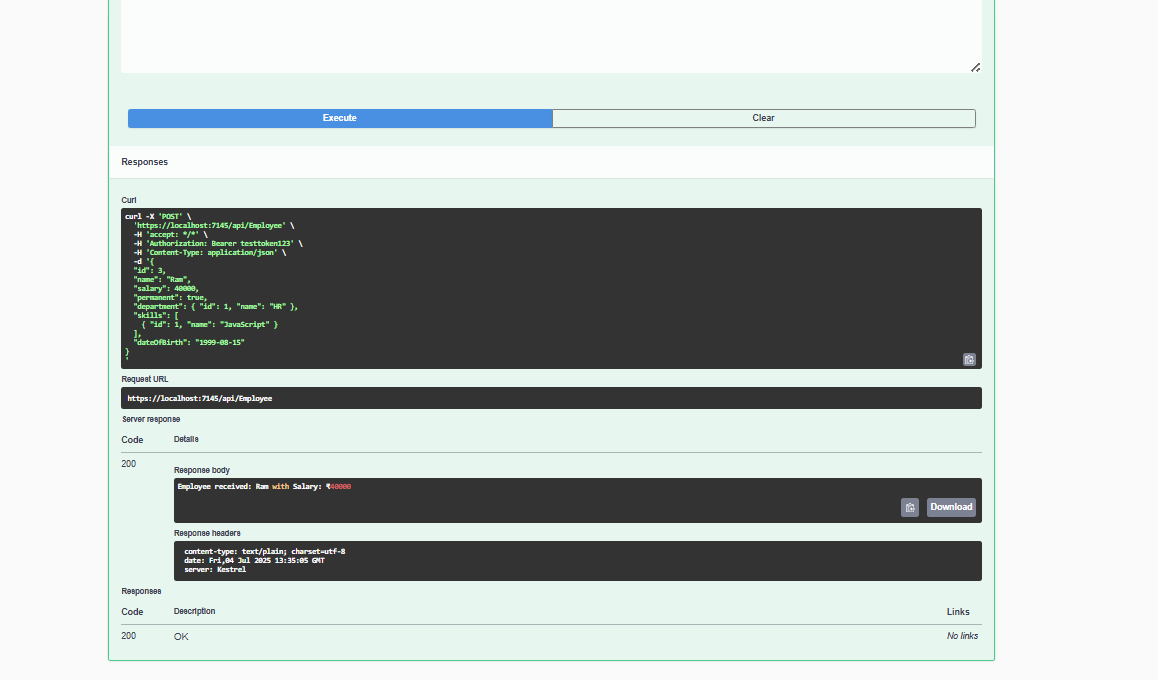
}

return Ok($"Employee received: {emp.Name} with Salary: ₹{emp.Salary}");

}

}

}  
  




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

**Filters folder**  
**CustomAuthFilter.cs**  
using Microsoft.AspNetCore.Mvc;

using Microsoft.AspNetCore.Mvc.Filters;

using EmployeeWebApi.Filters;

namespace EmployeeWebApi.Filters

{

public class CustomAuthFilter : ActionFilterAttribute

{

public override void OnActionExecuting(ActionExecutingContext context)

{

var headers = context.HttpContext.Request.Headers;

if (!headers.ContainsKey("Authorization"))

{

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

return;

}

var token = headers["Authorization"].ToString();

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

{

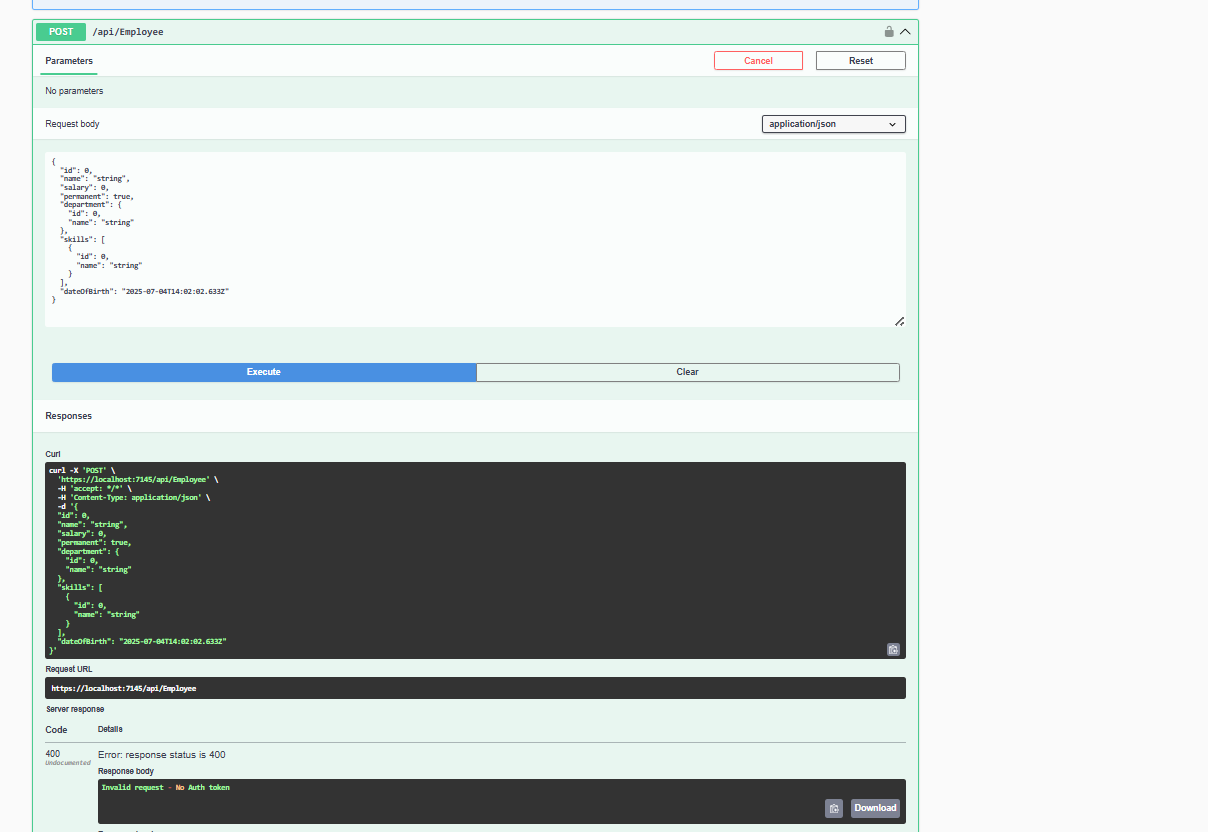
context.Result = new BadRequestObjectResult("Invalid request - Token present but Bearer unavailable");

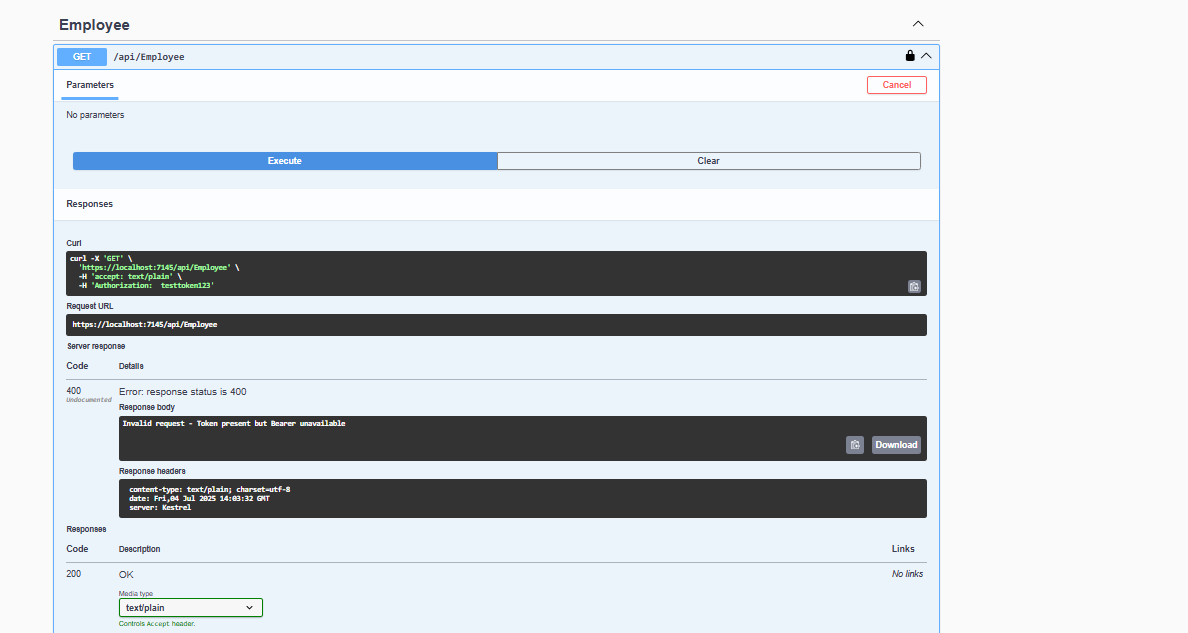
return;

}

base.OnActionExecuting(context);

}  
}

****



1. **Custom Exception filter**

**CustomExceptionFilter.cs**  
using Microsoft.AspNetCore.Mvc;

using Microsoft.AspNetCore.Mvc.Filters;

using System.IO;

namespace EmployeeWebApi.Filters

{

public class CustomExceptionFilter : IExceptionFilter

{

public void OnException(ExceptionContext context)

{

var ex = context.Exception;

File.WriteAllText("error\_log.txt", $"Exception: {ex.Message} at {DateTime.Now}");

context.Result = new ObjectResult("Something went wrong.")

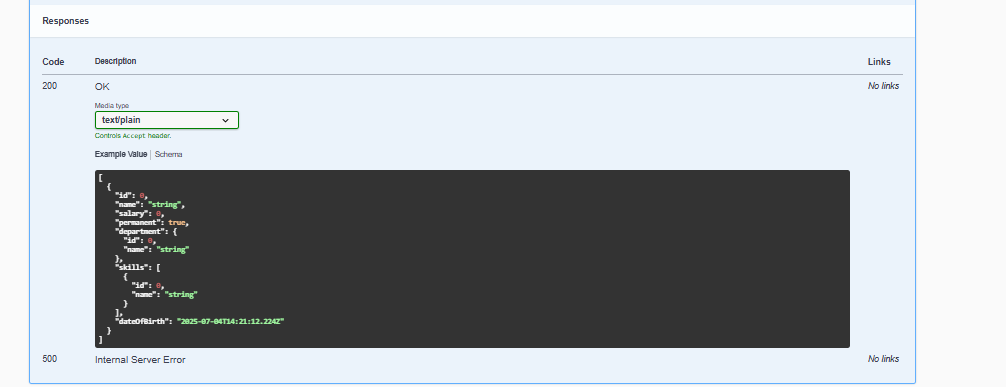
{

StatusCode = 500

};

}

}

}  
  


**Program.cs (common for all )**using EmployeeWebApi.Filters;

using Microsoft.OpenApi.Models;

var builder = WebApplication.CreateBuilder(args);

// Add services to the container

builder.Services.AddControllers(options =>

{

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

});

// Swagger configuration with Bearer token support

builder.Services.AddSwaggerGen(c =>

{

c.SwaggerDoc("v1", new() { Title = "EmployeeWebApi", Version = "v1" });

// Add Bearer token support

c.AddSecurityDefinition("Bearer", new OpenApiSecurityScheme

{

Name = "Authorization",

Type = SecuritySchemeType.ApiKey,

Scheme = "Bearer",

BearerFormat = "JWT",

In = ParameterLocation.Header,

Description = "Enter 'Bearer' [space] and then your token.\nExample: Bearer testtoken123"

});

c.AddSecurityRequirement(new OpenApiSecurityRequirement

{

{

new OpenApiSecurityScheme

{

Reference = new OpenApiReference

{

Type = ReferenceType.SecurityScheme,

Id = "Bearer"

}

},

Array.Empty<string>()

}

});

});

builder.Services.AddEndpointsApiExplorer();

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();