**3a)Web Api using custom model class**

**b) Create a Custom action filter for Authorization**

**c) Custom Exception filter**

**Department.cs:**

**namespace FirstWebApi.Models**

**{**

**public class Department**

**{**

**public int Id { get; set; }**

**public string Name { get; set; }**

**}**

**}**

**Skill.cs:**

**namespace FirstWebApi.Models**

**{**

**public class Skill**

**{**

**public int Id { get; set; }**

**public string Name { get; set; }**

**}**

**}**

**Employee.cs:**

**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 FirstWebApi.Filters;**

**using Microsoft.AspNetCore.Http;**

**namespace FirstWebApi.Controllers**

**{**

**[ApiController]**

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

**[CustomAuthFilter] // custom authorization**

**public class EmployeeController : ControllerBase**

**{**

**private static List<Employee> \_employees;**

**static EmployeeController()**

**{**

**\_employees = GetStandardEmployeeList();**

**}**

**[HttpGet]**

**[ProducesResponseType(typeof(List<Employee>), StatusCodes.Status200OK)]**

**[ProducesResponseType(StatusCodes.Status500InternalServerError)]**

**public ActionResult<List<Employee>> Get()**

**{**

**// Uncomment to test exception filter**

**// throw new Exception("Test exception from GET");**

**return Ok(\_employees);**

**}**

**[HttpGet("standard")]**

**public ActionResult<Employee> GetStandard()**

**{**

**return Ok(\_employees.FirstOrDefault());**

**}**

**[HttpPost]**

**public IActionResult Post([FromBody] Employee emp)**

**{**

**\_employees.Add(emp);**

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

**}**

**[HttpPut("{id}")]**

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

**{**

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

**if (existing == null) return NotFound();**

**existing.Name = emp.Name;**

**existing.Salary = emp.Salary;**

**existing.Permanent = emp.Permanent;**

**existing.Department = emp.Department;**

**existing.Skills = emp.Skills;**

**existing.DateOfBirth = emp.DateOfBirth;**

**return NoContent();**

**}**

**private static List<Employee> GetStandardEmployeeList()**

**{**

**return new List<Employee>**

**{**

**new Employee**

**{**

**Id = 1,**

**Name = "John",**

**Salary = 60000,**

**Permanent = true,**

**Department = new Department { Id = 1, Name = "HR" },**

**Skills = new List<Skill>**

**{**

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

**new Skill { Id = 2, Name = "SQL" }**

**},**

**DateOfBirth = new DateTime(1990, 5, 20)**

**}**

**};**

**}**

**}**

**}**

**CustomAuthFilter.cs:**

**using Microsoft.AspNetCore.Mvc;**

**using Microsoft.AspNetCore.Mvc.Filters;**

**namespace FirstWebApi.Filters**

**{**

**public class CustomAuthFilter : ActionFilterAttribute**

**{**

**public override void OnActionExecuting(ActionExecutingContext context)**

**{**

**if (!context.HttpContext.Request.Headers.TryGetValue("Authorization", out var token))**

**{**

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

**return;**

**}**

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

**{**

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

**}**

**}**

**}**

**}**

**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 exception = context.Exception;**

**File.AppendAllText("logs.txt", $"{DateTime.Now} - {exception.Message}\n");**

**context.Result = new ObjectResult("Internal Server Error")**

**{**

**StatusCode = 500**

**};**

**}**

**}**

**}**

**Program.cs:**

**using FirstWebApi.Filters;**

**using Microsoft.OpenApi.Models;**

**var builder = WebApplication.CreateBuilder(args);**

**// Register services + exception filter**

**builder.Services.AddControllers(options =>**

**{**

**options.Filters.Add<CustomExceptionFilter>();**

**});**

**// Add Swagger + Bearer Auth support**

**builder.Services.AddEndpointsApiExplorer();**

**builder.Services.AddSwaggerGen(c =>**

**{**

**c.SwaggerDoc("v1", new OpenApiInfo**

**{**

**Title = "Swagger Demo",**

**Version = "v1",**

**Description = "Custom Model, Authorization, Exception Filter"**

**});**

**// 🔐 Add support for Authorization header**

**c.AddSecurityDefinition("Bearer", new OpenApiSecurityScheme**

**{**

**Description = "Enter 'Bearer' followed by your token.\nExample: Bearer abc123",**

**Name = "Authorization",**

**In = ParameterLocation.Header,**

**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();**

**// Use Swagger**

**if (app.Environment.IsDevelopment())**

**{**

**app.UseSwagger();**

**app.UseSwaggerUI();**

**}**

**app.UseHttpsRedirection();**

**app.UseAuthorization();**

**app.MapControllers();**

**app.Run();**

