**Question 4: Validate JWT Token Expiry and Handle Unauthorized Access Scenario:**

**You want to handle expired or invalid tokens gracefully.**

**Steps: 1. Configure JWT bearer events.**

**2. Return custom messages for unauthorized access.**

**Program.cs:**

using Microsoft.AspNetCore.Authentication.JwtBearer;

using Microsoft.IdentityModel.Tokens;

using Microsoft.OpenApi.Models;

using System.IdentityModel.Tokens.Jwt;

using System.Text;

var builder = WebApplication.CreateBuilder(args);

// JWT Authentication with custom error handling

builder.Services.AddAuthentication(options =>

{

    options.DefaultAuthenticateScheme = JwtBearerDefaults.AuthenticationScheme;

    options.DefaultChallengeScheme = JwtBearerDefaults.AuthenticationScheme;

})

.AddJwtBearer(options =>

{

    options.TokenValidationParameters = new TokenValidationParameters

    {

        ValidateIssuer = true,

        ValidateAudience = true,

        ValidateLifetime = true,

        ValidateIssuerSigningKey = true,

        ValidIssuer = builder.Configuration["Jwt:Issuer"],

        ValidAudience = builder.Configuration["Jwt:Audience"],

        IssuerSigningKey = new SymmetricSecurityKey(

            Encoding.UTF8.GetBytes(builder.Configuration["Jwt:Key"]!))

    };

    // Custom error response

    options.Events = new JwtBearerEvents

    {

        OnAuthenticationFailed = async context =>

        {

            context.Response.StatusCode = 401;

            context.Response.ContentType = "application/json";

            var message = context.Exception is SecurityTokenExpiredException

                ? "{\"error\": \"Token has expired.\"}"

                : "{\"error\": \"Invalid token.\"}";

            await context.Response.WriteAsync(message);

        }

    };

});

builder.Services.AddAuthorization();

builder.Services.AddControllers();

// Swagger with JWT Support

builder.Services.AddEndpointsApiExplorer();

builder.Services.AddSwaggerGen(options =>

{

    options.SwaggerDoc("v1", new OpenApiInfo { Title = "JwtAuthDemo", Version = "v1" });

    options.AddSecurityDefinition("Bearer", new OpenApiSecurityScheme

    {

        Name = "Authorization",

        Type = SecuritySchemeType.ApiKey,

        Scheme = "Bearer",

        BearerFormat = "JWT",

        In = ParameterLocation.Header,

        Description = "Enter 'Bearer' followed by your token. Example: `Bearer abc123...`"

    });

    options.AddSecurityRequirement(new OpenApiSecurityRequirement

    {

        {

            new OpenApiSecurityScheme

            {

                Reference = new OpenApiReference

                {

                    Type = ReferenceType.SecurityScheme,

                    Id = "Bearer"

                }

            },

            Array.Empty<string>()

        }

    });

});

var app = builder.Build();

// Always enable Swagger (even in production for now)

app.UseSwagger();

app.UseSwaggerUI();

app.UseHttpsRedirection();

app.UseAuthentication();

app.UseAuthorization();

app.MapControllers();

app.Run();

**Output:**

