using HustleHub.BusinessArea.Interface;

using HustleHub.BusinessArea.Repository;

using HustleHub\_API.Data;

using HustleHub\_API.Middleware;

using Microsoft.EntityFrameworkCore;

using Microsoft.Extensions.FileProviders;

using Microsoft.OpenApi.Models;

using Microsoft.Extensions.Configuration;

using Microsoft.AspNetCore.Mvc;

var builder = WebApplication.CreateBuilder(args);

// --- Services ---

// Add controllers

builder.Services.AddControllers();

// Database Connection

builder.Services.AddDbContext<ApplicationDbContext>(options =>

{

options.UseSqlServer(

builder.Configuration.GetConnectionString("DefaultConnection"),

sqlOptions =>

{

sqlOptions.EnableRetryOnFailure(

maxRetryCount: 5,

maxRetryDelay: TimeSpan.FromSeconds(30),

errorNumbersToAdd: null

);

});

});

// Dependency Injection for your Repository

builder.Services.AddTransient<IRepository, Repository>();

// Swagger Configuration

builder.Services.AddEndpointsApiExplorer();

builder.Services.AddSwaggerGen(c =>

{

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

c.AddSecurityDefinition("ApiKey", new OpenApiSecurityScheme

{

Description = "ApiKey must appear in header",

Type = SecuritySchemeType.ApiKey,

Name = "XApiKey",

In = ParameterLocation.Header,

Scheme = "ApiKeyScheme"

});

var key = new OpenApiSecurityScheme()

{

Reference = new OpenApiReference

{

Type = ReferenceType.SecurityScheme,

Id = "ApiKey"

},

In = ParameterLocation.Header

};

var requirement = new OpenApiSecurityRequirement

{

{ key, new List<string>() }

};

c.AddSecurityRequirement(requirement);

});

// CORS Policy (Optional)

builder.Services.AddCors(options =>

{

options.AddPolicy("MyCorsPolicy", builder =>

{

builder.AllowAnyOrigin()

.AllowAnyHeader()

.AllowAnyMethod();

});

});

var app = builder.Build();

// --- Middlewares ---

// Swagger

if (app.Environment.IsDevelopment())

{

app.UseSwagger();

app.UseSwaggerUI();

}

// Create Upload folders if not exist

var uploadsPath = Path.Combine(Directory.GetCurrentDirectory(), "Uploads");

var imagesPath = Path.Combine(uploadsPath, "Images");

var documentsPath = Path.Combine(uploadsPath, "Documents");

if (!Directory.Exists(uploadsPath)) Directory.CreateDirectory(uploadsPath);

if (!Directory.Exists(imagesPath)) Directory.CreateDirectory(imagesPath);

if (!Directory.Exists(documentsPath)) Directory.CreateDirectory(documentsPath);

// Serve static files (Uploads)

app.UseStaticFiles(new StaticFileOptions

{

FileProvider = new PhysicalFileProvider(uploadsPath),

RequestPath = "/Uploads"

});

// CORS

app.UseCors("MyCorsPolicy");

// API Key Middleware (Check API key for all routes except public ones)

app.UseMiddleware<ApiKeyMiddleware>();

app.UseHttpsRedirection();

app.UseAuthorization();

// Controller Mapping

app.MapControllers();

app.Run();