Ready for PE.

1. Tools cần cài cho BussinessObject

* Design
* SqlServer
* Tools
* Configuration.Json

1. Lệnh Scaffold

* Mở tool -> Nuget package Manager -> Package Manager Console

Scaffold-DbContext "Server=DESKTOP-9C660FQ\NGUYENVI;Database=PE\_PRN\_23FallB5;uid=sa;pwd=12345;TrustServerCertificate=True;Trusted\_Connection=False;" Microsoft.EntityFrameworkCore.SqlServer -OutputDir Models

1. Sửa file default của Appsetting trong WebAPI

"ConnectionStrings": {

"DefaultConnection": "Server=DESKTOP-9C660FQ\\NGUYENVI;Database=PE\_PRN231\_Sum23B5;uid=sa;pwd=12345;TrustServerCertificate=True;Trusted\_Connection=False" },

* Nhớ vào Propeties của Appsetting chuyển sang Copy Always

1. Vào trong DbContext được tạo ra trong BussinessObject sửa OnConfiguring

protected override void OnConfiguring(DbContextOptionsBuilder optionsBuilder)

{

if (!optionsBuilder.IsConfigured)

{

optionsBuilder.UseSqlServer(GetConnectionString());

}

}

private string GetConnectionString()

{

var builder = new ConfigurationBuilder().SetBasePath(Directory.GetCurrentDirectory())

.AddJsonFile("appsettings.json", optional: true, reloadOnChange: true);

IConfiguration configuration = builder.Build();

return configuration.GetConnectionString("DefaultConnection");

}

1. Bắt đầu chuẩn bị JWT

"JWTSettings": {

"SecretKey": "PE\_SP24\_SE160956\_NguyenNgocThaiVi"

},

Thêm đoạn code trên vào trong appsetting để chbi làm JWT

Install Package JwtBearer 7.0.7

2. Thêm lẹ cái chỗ using cho đỡ tốn thời gian

using Microsoft.AspNetCore.Authentication.JwtBearer;

\*\*\* Thêm config swagger gen

builder.Services.AddEndpointsApiExplorer();

builder.Services.AddSwaggerGen(c =>

{

c.AddSecurityDefinition("Bearer", new OpenApiSecurityScheme

{

Description = "Please Enter The Token To Authenticate The Role",

Name = "Authorization",

BearerFormat = "JWTSettings",

In = ParameterLocation.Header,

Type = SecuritySchemeType.Http,

Scheme = JwtBearerDefaults.AuthenticationScheme

});

c.AddSecurityRequirement(new OpenApiSecurityRequirement

{

{

new OpenApiSecurityScheme

{

Reference = new OpenApiReference

{

Type = ReferenceType.SecurityScheme,

Id = JwtBearerDefaults.AuthenticationScheme

}

},

new string[] { }

}

});

});

Sau đó thêm

// Token

var serect = builder.Configuration["JWTSettings:SecretKey"];

var key = Encoding.ASCII.GetBytes(serect);

builder.Services.AddAuthentication(x =>

{

x.DefaultAuthenticateScheme = JwtBearerDefaults.AuthenticationScheme;

x.DefaultChallengeScheme = JwtBearerDefaults.AuthenticationScheme;

}).AddJwtBearer(x =>

{

x.RequireHttpsMetadata = false;

x.SaveToken = true;

x.TokenValidationParameters = new TokenValidationParameters

{

ValidateIssuerSigningKey = true,

IssuerSigningKey = new SymmetricSecurityKey(key),

ValidateIssuer = false,

ValidateAudience = false

};

});

Thêm DI Endpoit

// DI

builder.Services.AddEndpointsApiExplorer();

builder.Services.AddSwaggerGen();

builder.Services.AddDbContext<SilverJewelry2024DbContext>(option =>

option.UseSqlServer(builder.Configuration.GetConnectionString("DefaultConnection")));

//buider.Service.AddScoped…

Add block circle

// Seriable

builder.Services.AddControllers()

.AddJsonOptions(options => {

options.JsonSerializerOptions.ReferenceHandler = ReferenceHandler.IgnoreCycles;

});

1. Allow all host

var MyAllowSpecificOrigins = "\_myAllowSpecificOrigins";

builder.Services.AddCors(options =>

{

options.AddPolicy(name: MyAllowSpecificOrigins,

policy =>

{

policy.WithOrigins("\*").AllowAnyHeader().AllowAnyMethod();

});

});

Thêm đoạn code trên vào trong Program.cs

1. Add Odata

* Đầu tiên install Package AspNetCore.Odata

builder.Services.AddControllers().AddOData(options =>

options.AddRouteComponents(routePrefix: "odata", model: GetEdmModel())

.Select()

.Expand()

.OrderBy()

.SetMaxTop(100)

.Count()

.Filter());

// Odata

app.UseODataBatching();

app.UseHttpsRedirection();

static IEdmModel GetEdmModel()

{

ODataConventionModelBuilder builder = new ODataConventionModelBuilder();

//builder.EntitySet<Artworks>("Artworks");

//builder.EntitySet<Users>("Users");

//builder.EntitySet<Museums>("Museums");

return builder.GetEdmModel();

}

1. GenericDao

public class GenericDao<T> where T : class

{

public RoseTattooShop2023DbContext \_context;

public GenericDao(RoseTattooShop2023DbContext context)

{

\_context = context;

}

public void Add(T entity)

{

\_context.Set<T>().Add(entity);

}

public void Delete(T entity)

{

\_context.Set<T>().Remove(entity);

}

public virtual List<T> GetAll()

{

return \_context.Set<T>().ToList();

}

public virtual T? GetById(int id)

{

return \_context.Set<T>().Find(id);

}

public int Save()

{

return \_context.SaveChanges();

}

public void Update(T entity)

{

\_context.Set<T>().Update(entity);

}

}

1. Generic Repository

public interface IGenericRepository<T> where T : class

{

List<T> GetAll();

T? GetById(int id);

void Delete(T entity);

void Update(T entity);

void Add(T entity);

int Save();

}

1. Đối với Client
2. Tạo 1 Foler Inheritance để chưa class ClientAbstract.cs