**LẬP TRÌNH BACKEND BẰNG C#**

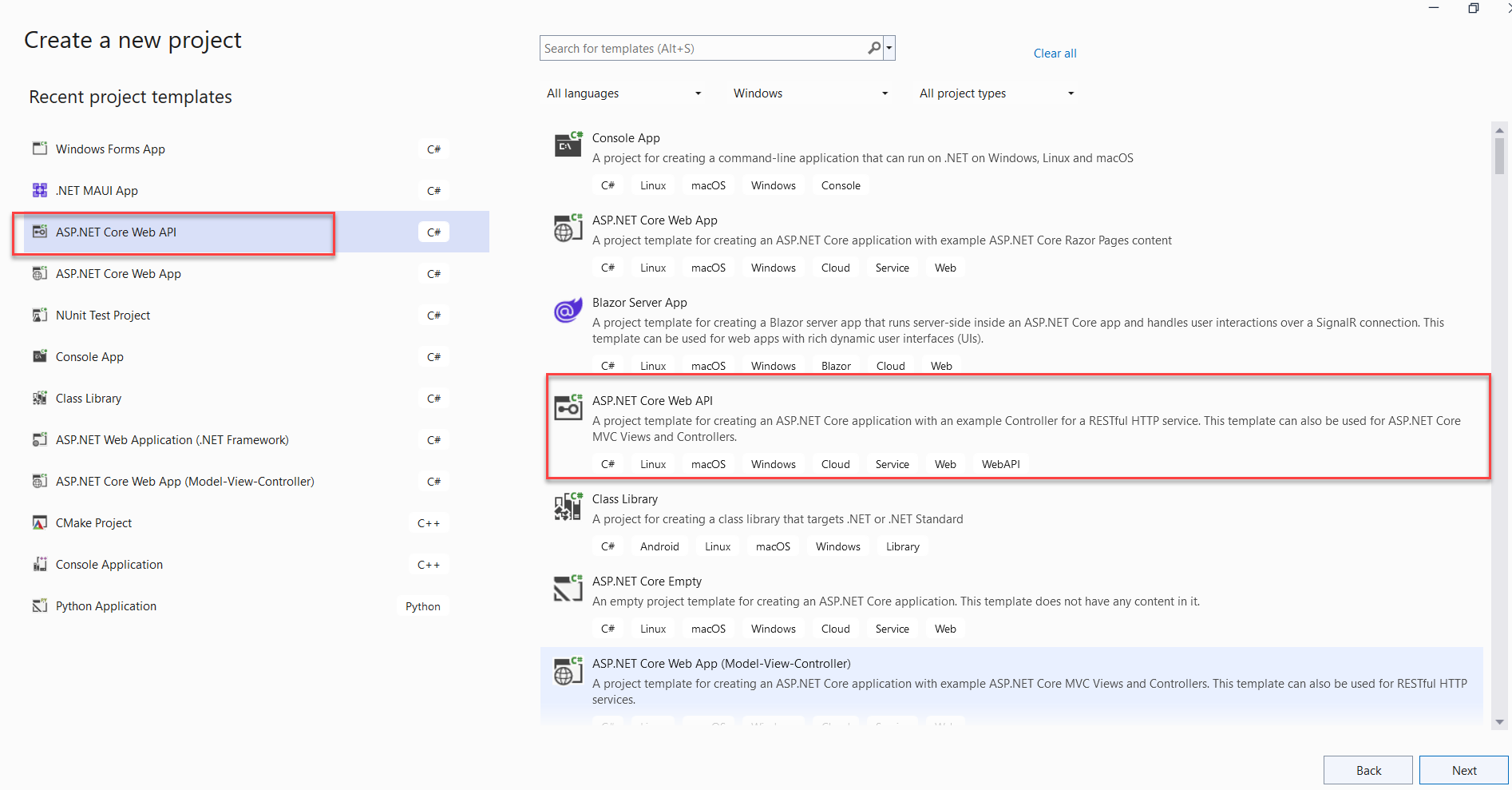
**1. Chuẩn bị CSDL**

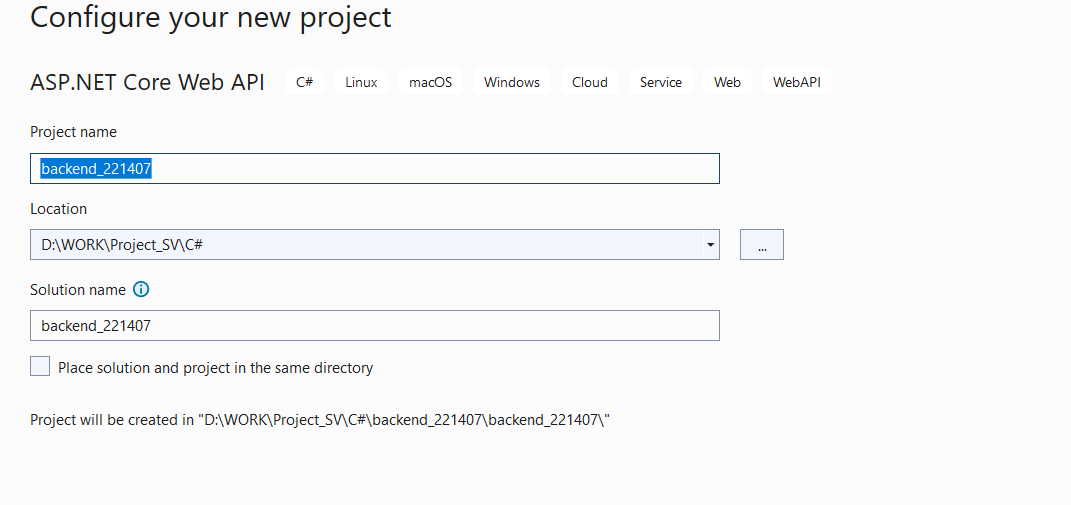
Chuẩn bị CSDL mẫu Northwind, là CSDL mẫu có từ những năm 2000. CSDL này có trên cả SQL Server lẫn MySQL. Link download Northwind cho SQL Server

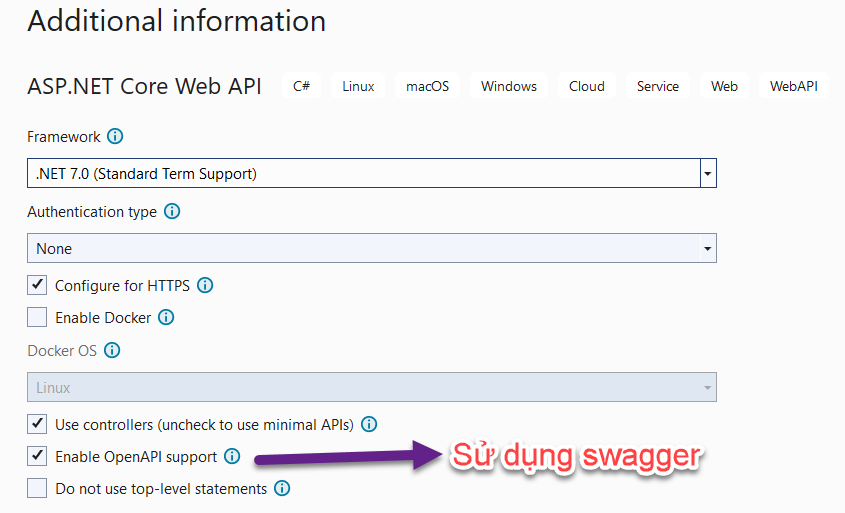
<https://github.com/Microsoft/sql-server-samples/tree/master/samples/databases/northwind-pubs>

**2. Tạo project**

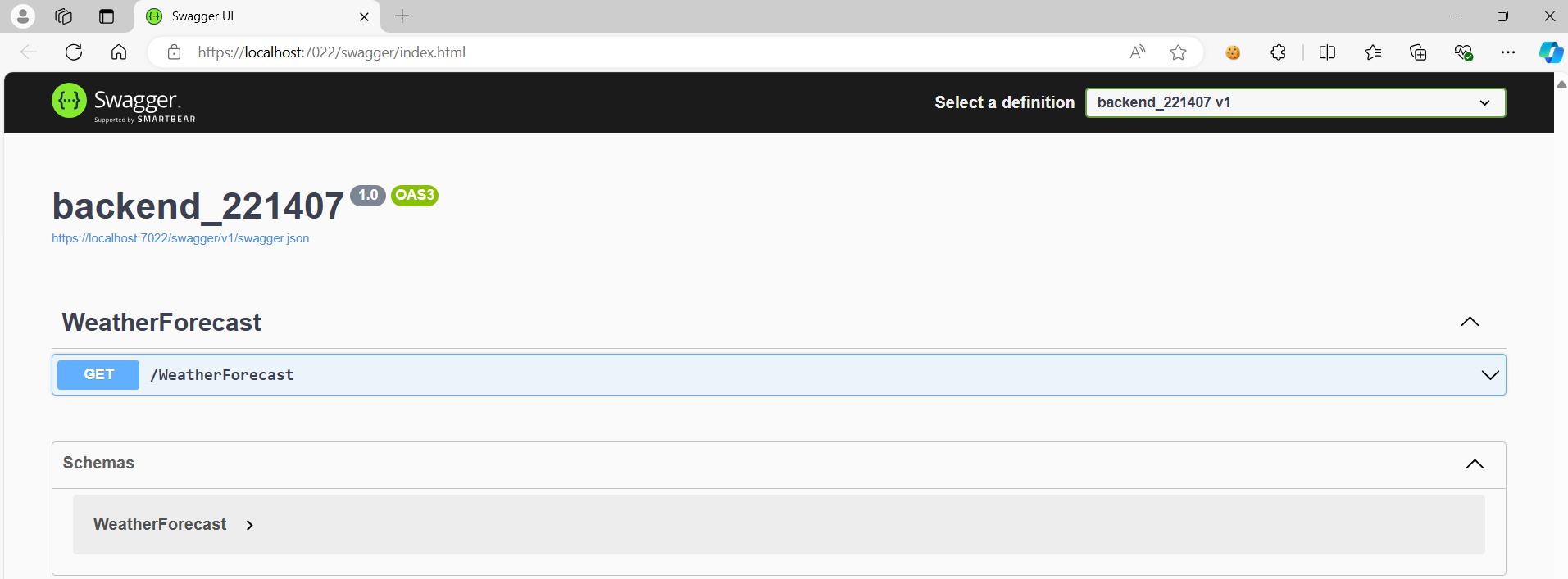
2.1. Tạo project loại ASP.NET.Core Web API, ngôn ngữ C#





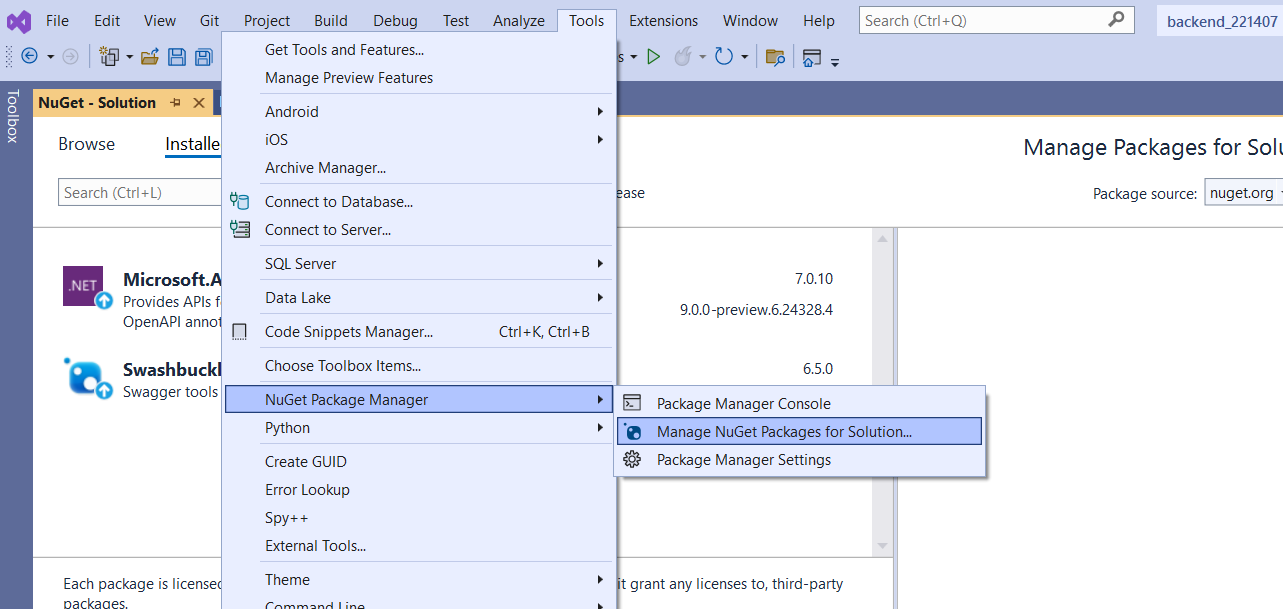


Chạy thử

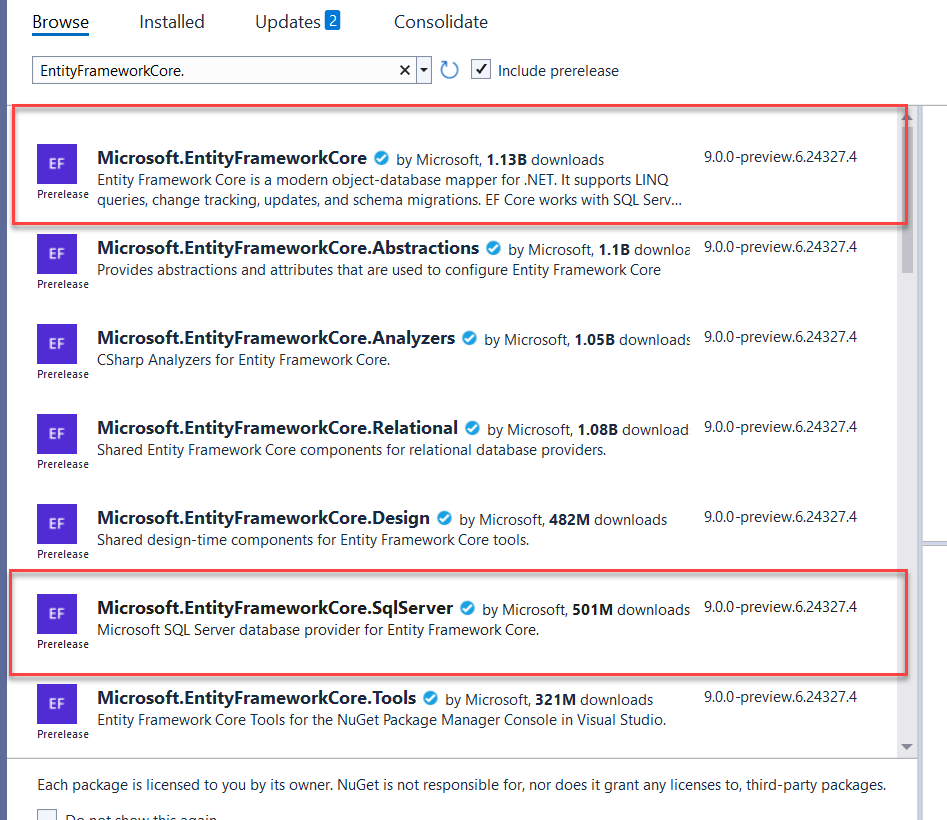


2.2. Add thêm gói thư viện Entity Frame work để lập trình

* Vào Tool\NuGet Packge Manager\Manager NuGet Packges for Solution…

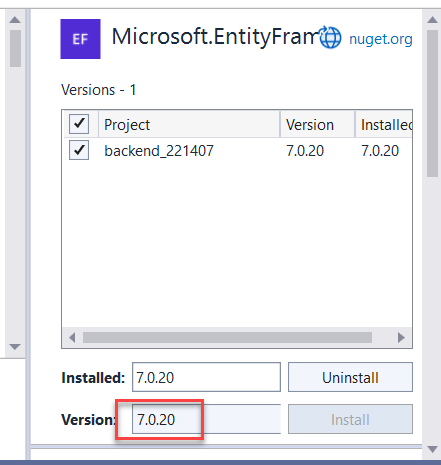


Gõ cụm từ “EntityFrameworkCore.”

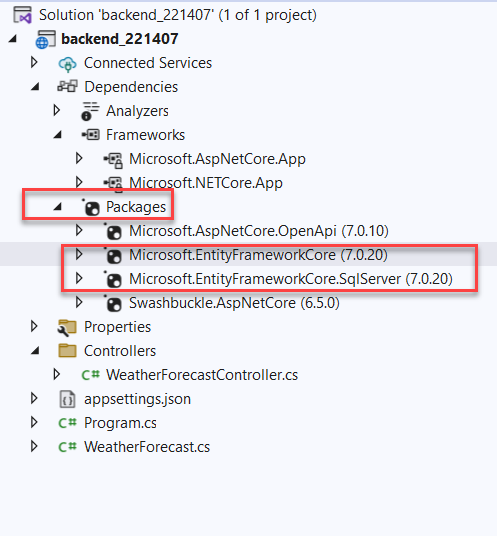


Cài Microsoft.EntityFrameworkCore và Microsoft.EntityFrameworkCore.SqlServer

Chú ý: Tùy theo phiên bản .NET để chọn phiên bản cho thư viện phù hợp nhé. Ví dụ: Thầy xài .NET 7 thì các phiên bản thư viện chỉ cần 7.xxx



Cài xong trong gói Packges sẽ có 2 thư viện như sau:



2.3. Tạo chuỗi kết nối SQL Server trong file cấu hình

Tìm đến file appsettings.json (Chú ý C# phiên bản mới xài file cấu hình .json thay cho .xml kiểu cũ). Thêm cụm cấu hình:

"ConnectionStrings": {

"Northwind\_conn": "Data Source=DESKTOP-BJVCGJB\\SQLEXPRESS;Initial Catalog=Northwind;User ID=sa;Password=12345678;TrustServerCertificate=True;Trusted\_Connection=true"

},

File cấu hình sẽ có dạng như sau:

{

"ConnectionStrings": {

"Northwind\_conn": "Data Source=DESKTOP-BJVCGJB\\SQLEXPRESS;Initial Catalog=Northwind;User ID=sa;Password=12345678;TrustServerCertificate=True;Trusted\_Connection=true"

},

"Logging": {

"LogLevel": {

"Default": "Information",

"Microsoft.AspNetCore": "Warning"

}

},

"AllowedHosts": "\*"

}

2.4. Tạo 1 class Global để chứa chuỗi kết nối lấy từ file cấu hình

namespace backend\_221407

{

public class Global

{

public static String getConnectString()

{

var builder = WebApplication.CreateBuilder();

string conStr = builder.Configuration.GetConnectionString("Northwind\_conn");

return conStr;

}

//Ứng dụng Windows Form thì lấy kết nối thế này

/\*

public static String getConnectionString()

{

String getstr =

ConfigurationManager.ConnectionStrings["Northwind\_conn"].ConnectionString;

return getstr;

}

\*/

public static int ToInt(object obj)

{

int i = 0;

int.TryParse(obj.ToString(), out i);

return i;

}

public static decimal ToDecimal(object obj)

{

decimal i = 0;

decimal.TryParse(obj.ToString(), out i);

return i;

}

}

}

2.5. Tạo các model

Tạo thư mục Models để chứa các model

+ Tạo class Product

using System.ComponentModel.DataAnnotations.Schema;

using System.ComponentModel.DataAnnotations;

namespace backend\_221407.Models

{

[Table("Products")]

public class Product

{

[Key]

public int ProductID { get; set; }

public String ProductName { get; set; }

public int? SupplierID { get; set; }

public int? CategoryID { get; set; }

public String? QuantityPerUnit { get; set; }

public decimal? UnitPrice { get; set; }

public Int16? UnitsInStock { get; set; }

public Int16? UnitsOnOrder { get; set; }

public Int16? ReorderLevel { get; set; }

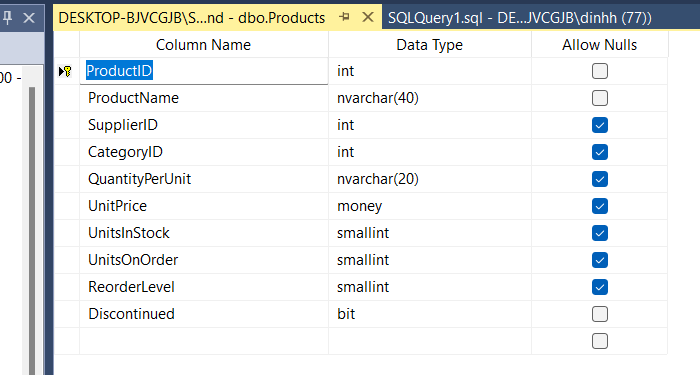
public bool Discontinued { get; set; }

}

}

Chú ý: Các thuộc tính ở class Product sẽ tương ứng với cấu trúc bảng Products trong CSDL, ví dụ kiểu Int16? sẽ tương đương với kiểu smallint cho phép NULL trong CSDL

Do tên bảng “Products” trong CSDL (có chữ “s” phía sau) trong khi class của chúng ta là “Product” nên có dòng map [Table("Products")]



+ Tạo class Category

using System.ComponentModel.DataAnnotations.Schema;

using System.ComponentModel.DataAnnotations;

namespace backend\_221407.Models

{

[Table("Categories")]

public class Category

{

[Key]

public int CategoryID { get; set; }

public String CategoryName { get; set; }

public String? Description { get; set; }

public byte[] Picture { get; set; }

}

}

+ Tạo class Supplier

using System.ComponentModel.DataAnnotations.Schema;

using System.ComponentModel.DataAnnotations;

namespace backend\_221407.Models

{

[Table("Suppliers")]

public class Supplier

{

[Key]

public int SupplierID { get; set; }

public String? CompanyName { get; set; }

public String? ContactTitle { get; set; }

public String? Address { get; set; }

public String? City { get; set; }

public String? Region { get; set; }

public String? PostalCode { get; set; }

public String? Country { get; set; }

public String? Phone { get; set; }

public String? Fax { get; set; }

public String? HomePage { get; set; }

}

}

2.5. Tạo DataContext kế thừa từ DbContext

using Microsoft.EntityFrameworkCore;

namespace backend\_221407.Models

{

public class ProductDBContext: DbContext

{

public DbSet<Category> Categories { get; set; }

public DbSet<Supplier> Suppliers { get; set; }

public DbSet<Product> Products { get; set; }

protected override void OnConfiguring(DbContextOptionsBuilder opt)

{

String conn = Global.getConnectString();

opt.UseSqlServer(conn);

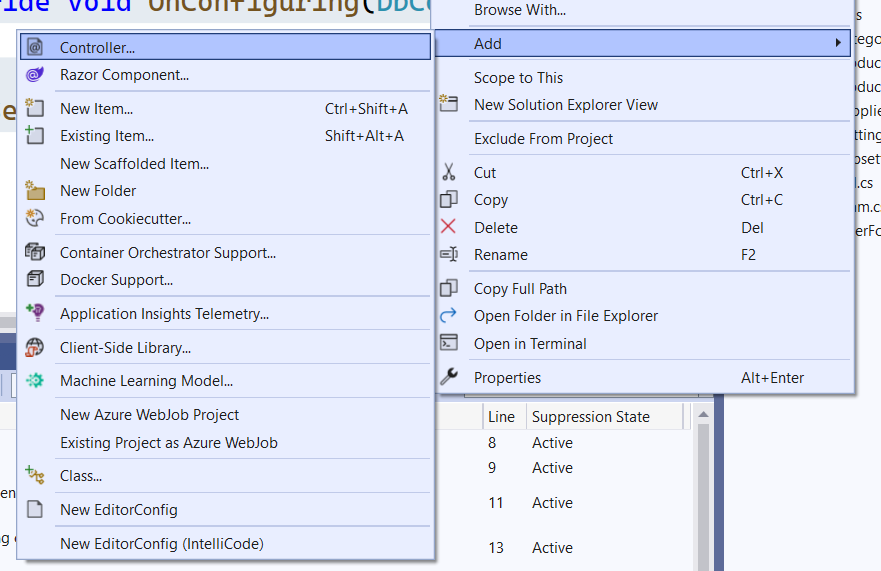
}

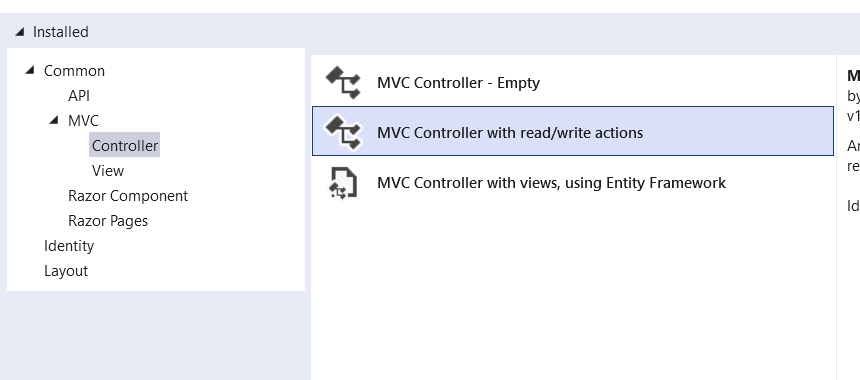
}

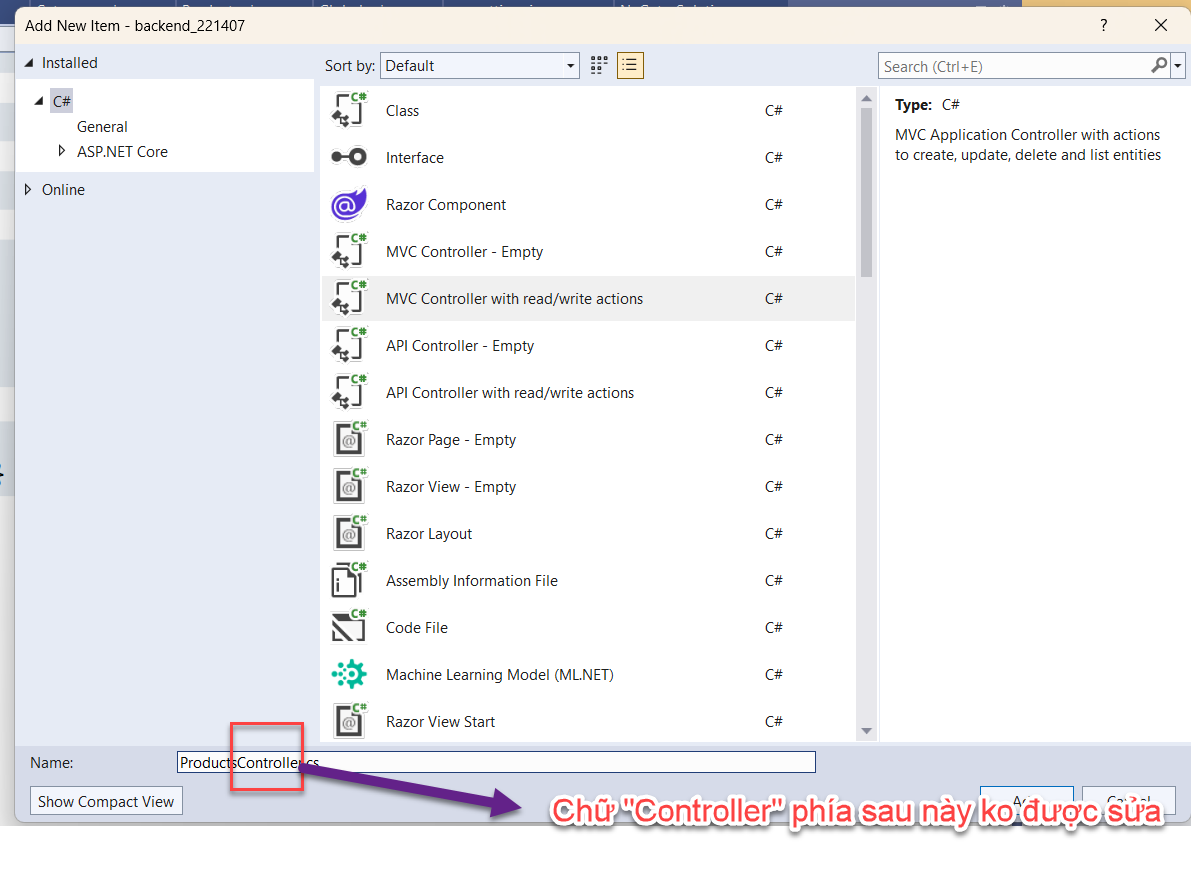
}

Trong ProductDBContext đã ánh xạ 3 bảng (DBSet) là Category, Supplier và Product

2.6. Tạo Controller ProductsController







Code của class ProductsController

using backend\_221407.Models;

using Microsoft.AspNetCore.Http;

using Microsoft.AspNetCore.Mvc;

namespace backend\_221407.Controllers

{

public class ProductsController : Controller

{

/\* Danh mục Categories \*/

[HttpGet("ListCategories")]

public IEnumerable<Category> ListCategories()

{

ProductDBContext dtx = new ProductDBContext();

return dtx.Categories;

}

/\* Danh mục Supplier \*/

[HttpGet("ListSuppliers")]

public IEnumerable<Supplier> ListSuppliers()

{

ProductDBContext dtx = new ProductDBContext();

return dtx.Suppliers;

}

[HttpGet("ListProducts")]

public IEnumerable<Product> ListProducts()

{

ProductDBContext dtx = new ProductDBContext();

return dtx.Products.Take(100).ToList();

}

[HttpGet("SearchProducts")]

public IEnumerable<object> SearchProducts(String? str\_search, int? pagenumber, int? pagesize)

{

if (str\_search == null)

str\_search = "";

if (!pagesize.HasValue)

pagesize = 20;

if (!pagenumber.HasValue)

pagenumber = 1;

int begin = (pagenumber.Value - 1) \* pagesize.Value;

ProductDBContext dtx = new ProductDBContext();

return (from x in dtx.Products

join cat in dtx.Categories on x.CategoryID equals cat.CategoryID

join sup in dtx.Suppliers on x.SupplierID equals sup.SupplierID

where x.ProductName.Contains(str\_search) || cat.CategoryName.Contains(str\_search) ||

sup.CompanyName.Contains(str\_search) || sup.Country.Contains(str\_search) || sup.Phone.Contains(str\_search)

select new

{

ProductID = x.ProductID,

ProductName = x.ProductName,

CategoryID = x.CategoryID,

CategoryName = cat.CategoryName,

SupplierID = x.SupplierID,

CompanyName = sup.CompanyName,

QuantityPerUnit = x.QuantityPerUnit,

UnitPrice = x.UnitPrice

}

)

.Skip(begin).Take(pagesize.Value).ToList();

}

[HttpGet("GetCount")]

public int GetCount(String? str\_search)

{

if (str\_search == null)

str\_search = "";

ProductDBContext dtx = new ProductDBContext();

int count = (int)(from x in dtx.Products

join cat in dtx.Categories on x.CategoryID equals cat.CategoryID

join sup in dtx.Suppliers on x.SupplierID equals sup.SupplierID

where x.ProductName.Contains(str\_search) || cat.CategoryName.Contains(str\_search) || sup.CompanyName.Contains(str\_search)

select x

).Count();

return count;

}

// GET api/<ProductsController>/5

[HttpGet("{id}")]

public Product Get(int id)

{

ProductDBContext dbx = new ProductDBContext();

Product product = (from x in dbx.Products where x.ProductID == id select x).FirstOrDefault();

return product;

}

// POST api/<ProductsController>

[HttpPost]

public void Post([FromBody] string value)

{

}

// PUT api/<ProductsController>/5

[HttpPut("{id}")]

public void Put(int id, [FromBody] string value)

{

}

// DELETE api/<ProductsController>/5

[HttpDelete("{id}")]

public void Delete(int id)

{

}

}

}

Kết quả chạy

