PRODUCT MANAGEMENT USING DATABASE FIRST APPROACH (Reverse Engineering)

Khung Solution - Tạo 3 Project Class Library, 1 Project Windows Form

- BusinessObjects
- DataAccessObjects - references BusinessObjects
- Repositories - references DataAccessObjects
- ProductManagementWindowsForms - references Repositories

Bước 1. Chạy Script MyStore.sql → tạo Database MyStore trong SQL Server

- Database=MyStore
- Uid= sa
- Pwd=123

→ Chuỗi kết nối tới CSDL "**Server=(local); Database=MyStore; Uid=sa; Pwd=123**, TrustServerCertificate=True;"

Bước 2. Khai báo các thư viện NuGet packages trong Project **BusinessObjects**

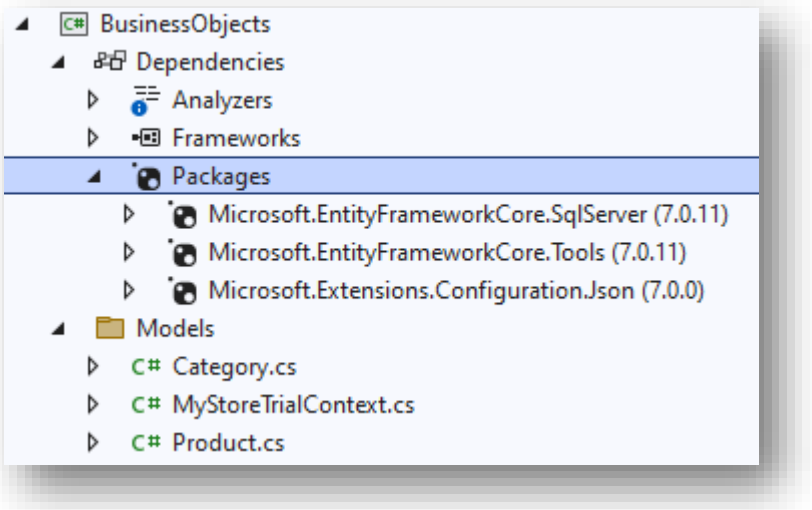
- Microsoft.EntityFrameworkCore.SqlServer
- Microsoft.EntityFrameworkCore.Tools
- Microsoft.Extensions.Configuration.Json

Kiểm tra tool EFCore và install tool

dotnet tool install --global dotnet-ef

Vào cửa sổ Developer PowerShell

- Thêm các packages dùng lệnh
  dotnet add package Microsoft.EntityFrameworkCore.SqlServer
  dotnet add package Microsoft.EntityFrameworkCore.Tools
  dotnet add package Microsoft.Extensions.Configuration.Json



- Thực hiện ORM
  dotnet ef dbcontext scaffold "**Server=(local); Database=MyStoreTrial; Uid=sa; Pwd=123**, TrustServerCertificate=True;" Microsoft.EntityFrameworkCore.SqlServer --output-dir ./

- Thay đổi chuỗi kết nối hardcode

```
// using System.IO;
// using Microsoft.Extensions.Configuration.Json;
private string GetConnectionString()
{
        IConfiguration configuration = new ConfigurationBuilder()
                .SetBasePath(Directory.GetCurrentDirectory())
                .AddJsonFile("appsettings.json", true, true).Build();
         return configuration["ConnectionStrings:DefaultConnectionString"];
}

protected override void OnConfiguring(DbContextOptionsBuilder optionsBuilder)
{
        optionsBuilder.UseSqlServer(GetConnectionString());
}
```

Bước 3. Làm việc Project DataAccessObjects

- Reference tới Project BusinessObjects
- Viết lớp ProductDAO.cs, CategoryDAO.cs

CategoryDAO.cs

```
public class CategoryDAO
{
    1 reference
    public static List<Category> GetCategories ()
    {
        List<Category> listCategories = new List<Category>();
        try
        {
            using (var context = new MyStoreContext())
            {
                listCategories = context.Categories.ToList();
            }
        }
        catch (Exception ex)
        {
            throw new Exception(ex.Message);
        }
        return listCategories;
    }
}
```

ProductDAO.cs

```
public class ProductDAO
{
    // Xử lý Singleton bổ sung
    1 reference
    public static List<Product> GetProducts()
    {
        List<Product> listProducts = new List<Product>();
        try
        {
            using (var context = new MyStoreContext())
            {
                listProducts = context.Products.Include(f => f.Category).ToList();
            }
        }
        catch(Exception ex)
        {
            throw new Exception(ex.Message);
        }
        return listProducts;
    }
    1 reference
    public static Product GetProductById(int id)...
    1 reference
    public static void SaveProduct(Product product)...
    1 reference
    public static void DeleteProduct(Product product)...
    1 reference
    public static void UpdateProduct(Product product)...
}
```

Bước 4. Làm việc với Project Repositories

- Reference tới DataAccessObjects
- Tạo Interface để giao tiếp: IProductRepository.cs

```
public interface IProductRepository
{
        //void SaveProduct(Product p);
        //void DeleteProduct(Product p);
        //void UpdateProduct(Product p);
        List<Category> GetCategories();
        List<Product> GetProducts();
        //Product GetProductById(int id);
 }
```

- Tạo lớp implements interface: ProductRepository.cs

```
public class ProductRepository : IProductRepository
 {
        //public void DeleteProduct(Product p)=>ProductDAO.DeleteProduct(p);

        public List<Category> GetCategories()=>CategoryDAO.GetCategories();

        //public Product GetProductById(int id)=>ProductDAO.GetProductById(id);
```

```
        public List<Product> GetProducts()=>ProductDAO.GetProducts();

        //public void SaveProduct(Product p)=>ProductDAO.SaveProduct(p);

        //public void UpdateProduct(Product p)=>ProductDAO.UpdateProduct(p);
    }
```

Bước 5. Làm với Project ProductManagementWindowsForms - references Repositories

- Tạo tập tin appsettings.json - khai báo chuỗi kết nối- thay đổi CopyToOutputDirectory= Copy Always

```
{
  "ConnectionStrings": {
    "DefaultConnectionString": "Server=(local);Database=MyStore;Uid=sa;Pwd=123;
TrustServerCertificate=True;"
  }
}
```

- Tạo form có một số controls như hình

Bước 6. Thực hiện các chức năng Update, Create, Delete

- Viết các hàm xử lý trong ProductDAO.cs
- Tạo giao diện