Demo web application mvc

1. Introduction

- IDE : VSCode or Visual Studio 2022
- Database: SQL server, MySQL ...
- Install Package: Entity Framework Core
- CRUD object (EF)
- Code first

2. Objectives

- Use the Visual Studio.NET to create ASP.NET Core Web App (MVC) and Class Library (.dll) projects.
- Create a SQL Server database named MyStoreDB that has a Product, Category, AccountMember tables.
- Apply Repository pattern in a project.
- Add CRUD action methods to ASP.NET Core Web App (MVC).
- Run the project and test the application actions.

3. Database Design

- Entity: AccountMember

Field	Type	Key	Description
MemberID	PK	✔	Mã thành viên
MemberPassword	Text		Mật khẩu
FullName	Text		Họ và tên
EmailAddress	Text		Địa chỉ email
MemberRole	Text		Vai trò thành viên

- Entity: Categories

Field	Type	Key	Description
CategoryID	PK	✔	Mã danh mục
CategoryName	Text		Tên danh mục

- Entity: Products

Field	Type	Key	Description
ProductID	PK	✔	Mã sản phẩm
ProductName	Text		Tên sản phẩm
CategoryID	FK	→Categories(CategoryID)	Liên kết đến danh mục
UnitsInStock	Integer		Số lượng còn lại
UnitPrice	Decimal		Đơn giá

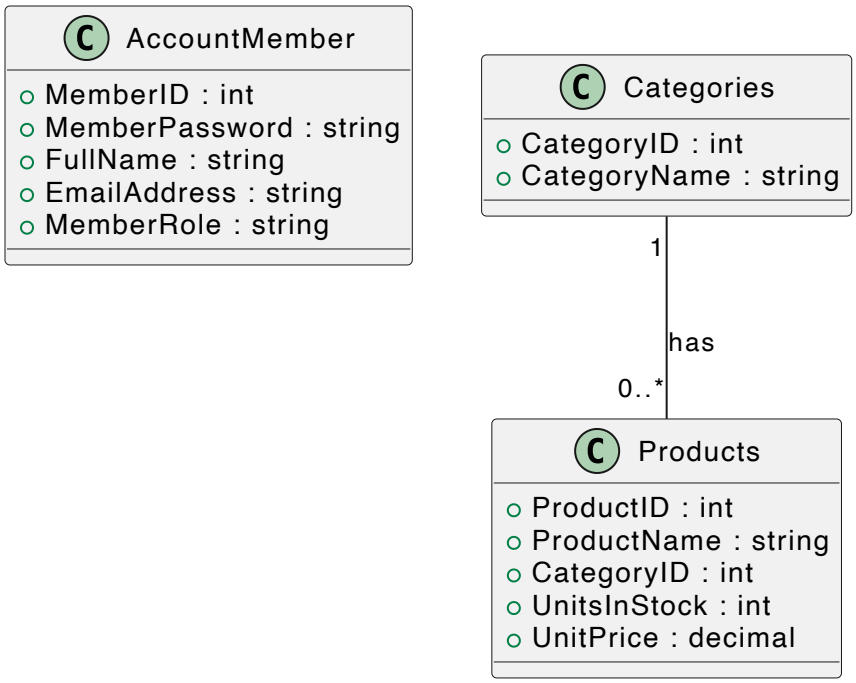
- Sql script

```sql
-- Bảng AccountMember
CREATE TABLE AccountMember (
    MemberID INT PRIMARY KEY,
    MemberPassword VARCHAR(255) NOT NULL,
    FullName VARCHAR(100),
    EmailAddress VARCHAR(100),
    MemberRole VARCHAR(50)
);

-- Bảng Categories
CREATE TABLE Categories (
    CategoryID INT PRIMARY KEY,
    CategoryName VARCHAR(100) NOT NULL
);

-- Bảng Products
CREATE TABLE Products (
    ProductID INT PRIMARY KEY,
    ProductName VARCHAR(100) NOT NULL,
    CategoryID INT,
    UnitsInStock INT,
    UnitPrice DECIMAL(10, 2),
    FOREIGN KEY (CategoryID) REFERENCES Categories(CategoryID)
);
```

- UML Class Diagram
- ORM



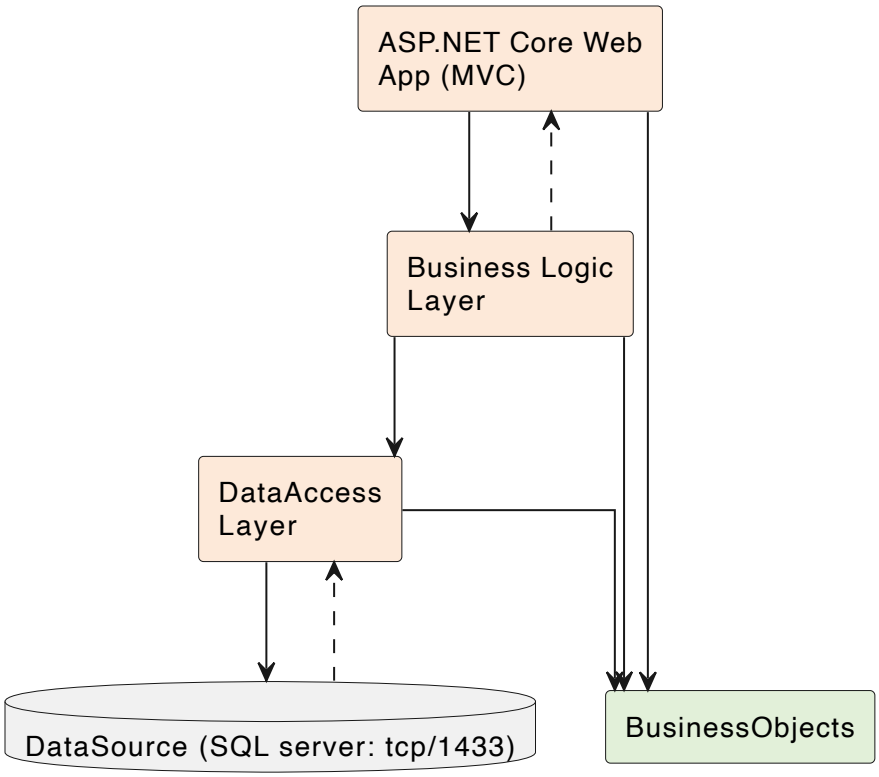
4. Create MVC with 3 layer

- Create a Blank Solution named MyStore then add new a Class Library project named:
- MyStore.Business
- MyStore.Repositories
- MyStore.Services
- ASP.NET Core Web App (MVC) project named MyStore.WebApp

  Step 01. Create a Blank solution.

  Step 02. Create 4 Class Library projects.

  Step 03. Create a project (ASP.NET Core Web App MVC).



Note

- Data Source in this case is the SQL Server Database
- Services Project: This project represents a layer or component responsible for implementing the business logic of an application.
- Repository Project: This project provides an abstraction layer between the application's business logic and the underlying data source.
- Data Access Layer Project: This project used to abstract and encapsulate the logic for accessing data from a data source, such as a database.

5. Write codes for the BusinessObjects project

Step 01. Install the following packages from NuGet:

- Microsoft.EntityFrameworkCore.SqlServer --version 8.0.2
- Microsoft.EntityFrameworkCore.Tools --version 8.0.2
- Microsoft.Extensions.Configuration.Json --version 8.0.0

Check the tool for EFCore (install/uninstall tool if needed) (dotnet SDK 8.0.202)

Install EF (versoin 8) = .Net (8)

```
dotnet tool install --global
dotnet-ef --version 8.0.2
dotnet tool uninstall --global
```

6. Implement ORM (Folder Business Logic Layer)

```
dotnet ef dbcontext scaffold "Data Source=127.0.0.1,1433;Initial Catalog=M
```

- Change the connection string in OnConfiguring() function of MyStoreContext.cs

```
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());
}


//Category.cs
public partial class Category
{
    public int CategoryId { get; set; }

    public string CategoryName { get; set; } = null!;

    public virtual ICollection<Product> Products { get; set; } = new List<Product>(
}
```

7. Structure mvc

- **MyStore.WebMVC**
  - Controllers
  - Models
  - Views
  - wwwroot *(Thư mục chứa các tài nguyên tĩnh như HTML, CSS, JavaScript, images, font
    - assets *(các tập tin tài nguyên như images hoặc các thành phần giao diện tùy chỉ
    - css *(các tập tin CSS)*
    - fonts *(font chữ được sử dụng trong dự án)*
    - html *(các file HTML tĩnh, nếu có)*
    - js *(các tập tin JavaScript)*
    - lib *(thư viện của bên thứ ba)*
    - libs *(có thể chứa thư viện hoặc mã thư viện bổ sung)*
    - scss *(các tập tin SCSS, mã nguồn để biên dịch ra CSS)*
    - tasks *(các script hoặc tập tin nhiệm vụ cho build hoặc automation)*
    - favicon.ico *(biểu tượng hiển thị trên tab trình duyệt)*
  - appsettings.Development.json
  - appsettings.json
  - MyStore.WebMVC.csproj
  - Program.cs

7.1. Structure View (_Layout)

- **Views**
  - **Shared**
    - **_Layout.cshtml** *(Template chung, định nghĩa cấu trúc chung của toàn bộ gia
    - **_ViewStart.cshtml** *(Chỉ định Layout mặc định)*
    - **_ViewImports.cshtml** *(Định nghĩa namespace và thư viện chung)*
    - **_PartialView.cshtml** *(Các thành phần nhỏ, dùng lại trong giao diện)*
  - **Home** *(Thư mục lưu View của Controller tương ứng)*
    - **Index.cshtml**
    - **About.cshtml**
    - **Contact.cshtml**
  - **Products**
    - **Index.cshtml**
    - **Create.cshtml**
    - **Edit.cshtml**
    - **Details.cshtml**
    - **Delete.cshtml**

7.2 Create Partial Views

```
Views
└ Shared
  ├ _Layout.cshtml
  ├ _Header.cshtml
  ├ _Nav.cshtml
  └ _Footer.cshtml
```

- ex: file _Header.cshtml

```
<header>
  <h1>MyStore</h1>
</header>
```

- ex: _Nav.cshtml

```
<nav>
  <ul>
    <li><a asp-controller="Home" asp-action="Index">Home</a></li>
    <li><a asp-controller="Products" asp-action="Index">Products</a></li>
  </ul>
</nav>
```

- ex: file _Footer.cshtml

```
<footer>
  <p>&copy; 2025 MyStore. All rights reserved.</p>
</footer>
```

7.3 3. Merge Partial Views to _Layout.cshtml

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="utf-8" />
    <title>@ViewData["Title"] - MyStore</title>
</head>
<body>
    @await Html.PartialAsync("_Header")

    @await Html.PartialAsync("_Nav")

    <div class="container">
        @RenderBody()
    </div>

    @await Html.PartialAsync("_Footer")

    @RenderSection("Scripts", required: false)
</body>
</html>
```