**Assignment 03**

**Building a Web Application with ASP.NET Core MVC**

# Introduction

Hãy tưởng tượng bạn là nhân viên của một nhà bán lẻ sản phẩm có tên **eStore**. Người quản lý của bạn đã yêu cầu bạn phát triển một ứng dụng Web để quản lý thành viên, quản lý sản phẩm và quản lý đơn hàng. Ứng dụng có một tài khoản mặc định có email là "**admin@estore.com**" và mật khẩu là "**admin@@**" được lưu trữ trong **appsettings.json**.

Ứng dụng phải hỗ trợ thêm, xem, sửa đổi và xóa sản phẩm — một động từ hành động sử dụng được tiêu chuẩn hóa hay còn được gọi là Tạo, Đọc, Cập nhật, Xóa (CRUD) và Tìm kiếm. Nhiệm vụ này khám phá việc tạo một ứng dụng bằng Windows Forms với .NET Core, C # và ADO.NET / Entity Framework. Một cơ sở dữ liệu MS SQL Server sẽ được tạo ra để duy trì dữ liệu và nó sẽ được sử dụng để đọc và quản lý dữ liệu.

# Assignment Objectives

In this assignment, you will:

* Use the Visual Studio.NET to create a Web application and Class Library (.dll) project.
* Perform CRUD actions using ADO.NET and Entity Framework Core
* Use LINQ to query and sort data
* Apply passing data in ASP.NET Core MVC application
* Apply 3-layers architecture to develop the application
* Apply Repository pattern and Singleton pattern in a project
* Add CRUD and searching actions to the Web application.
* Apply to validate data type for all fields
* Run the project and test the actions of the Web application.

# Database Design

# 

# Main Functions

* Member management, Product management, and Order management: Read, Create, Update and Delete actions.
* Search ProductName (keywork of ProductName) and UnitPrice
* Create a report statistics sales by the period from StartDate to EndDate, and sort sales in descending order
* Member authentication by Email and Password. If the user is “**Admin**” then allows to perform all actions, otherwise, the normal user is allowed to view/update the profile and view their orders history.

# Guidelines

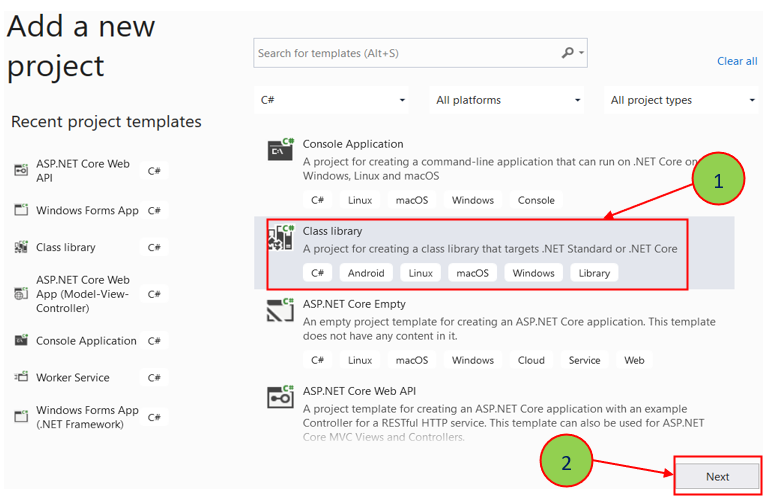
# Activity 01: Build a solution [01 mark]

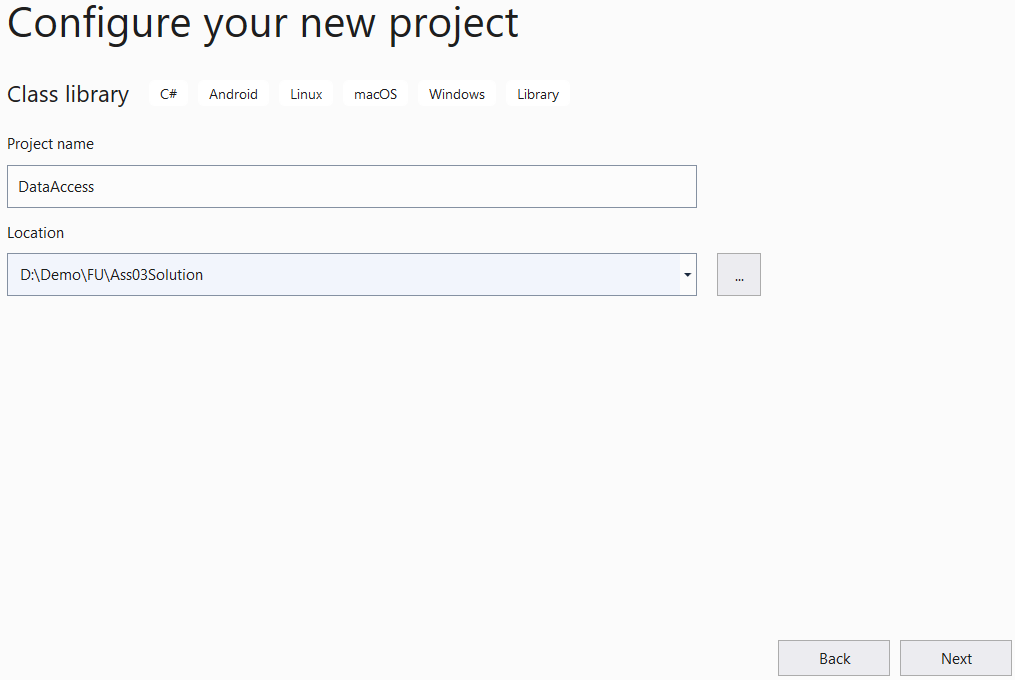
Create a Blank Solution named **Ass03Solution** that includes Class Library Project: **DataAccess, BusinessObject,** and an ASP.NET Core MVC project named **eStore**

**Step 01**. Open the Visual Studio .NET application and create a Blank solution named **Asm03Solution**

**Step 02.** Create a Class Library project named **DataAccess**

From the File menu | Add | New Project, on the Add New Project dialog, select “Class Library” and performs steps as follows:



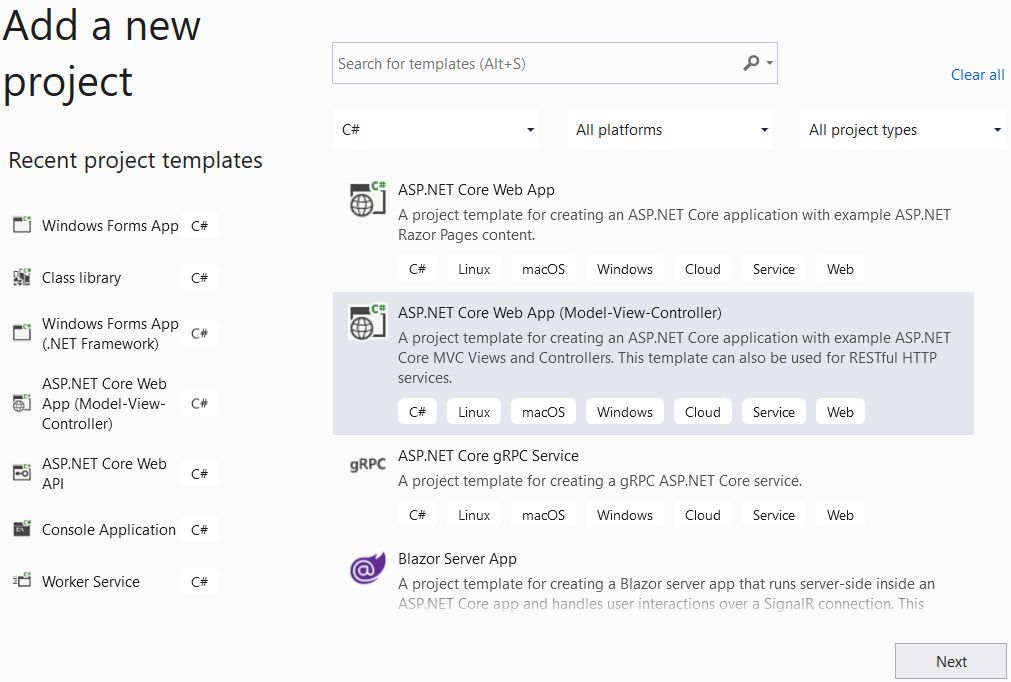
****

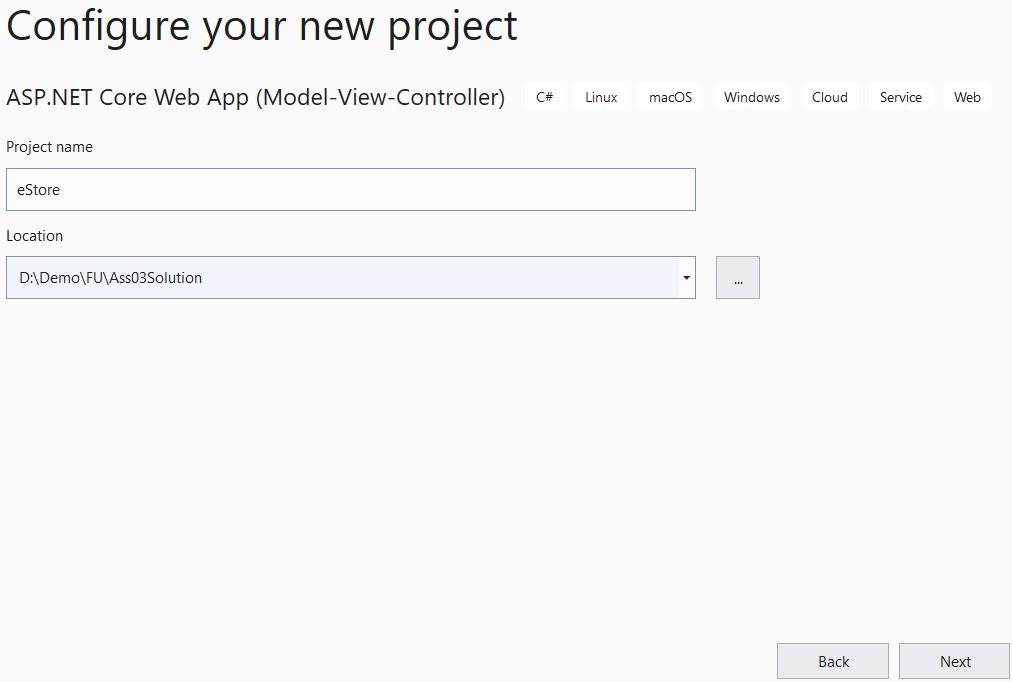
****

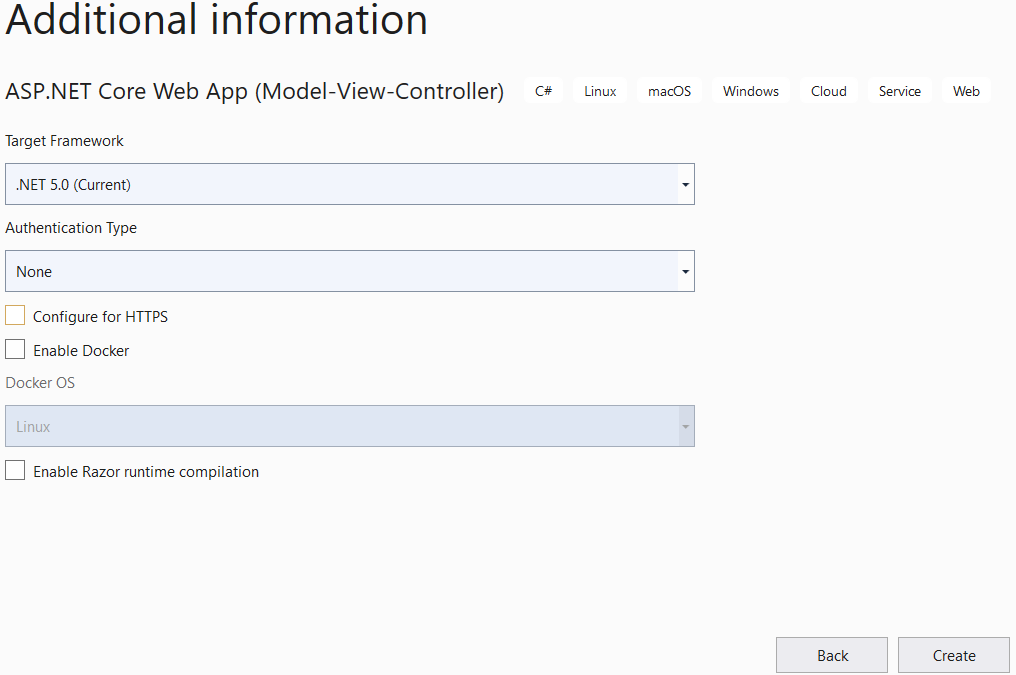
**Step 03.** Repeat **Step 02** to create a **BusinessObject** project.

**Step 04.** Create an ASP.NET Core MVC project named **eStore**

* From the File menu | Add | New Project, on the Add New Project dialog, select “ASP.NET Core Web App (Model-View-Controller)” and performs steps as follows:







**Step 05**. Create folders and add class to the projects as follows:

# 

# Activity 02: Develop BusinessObject project [02 marks]

**Step 01**. Write codes to create classes and definition all data members

**Step 02**. Write codes to perform business rules for data members

# Activity 03: Develop DataAccess project [03 marks]

***Hints*:** If using Entity Framework, you can install the AutoMapper package from Nuget to map Entity with Busines Object.

**Step 01**. Add a project reference to the **BusinessObject** project

**Step 02**. Write codes for **MemberDAO.cs, IMemberRepository.cs** and **MemberRepository.cs**

**Step 03**. Write codes for **ProductDAO.cs, IProductRepository.cs** and **ProductRepository.cs**

**Step 04**. Write codes for **OrderDAO.cs, IOrderRepository.cs** and **OrderRepository.cs**

**Step 05**. Write codes for **OrderDetailDAO.cs, IOrderDetailRepository.cs** and **OrderDetailRepository.cs**

# Activity 04: Develop MyStoreWinApp project [03 marks]

**Step 01**. Add a reference to **BusinessObject** and **DataAccess** project.

**Step 02**. Design UI for views and write codes for controllers to perform functions.

# Activity 05: Run the Web project and test all actions [01 mark]

# For example: Search products by ProductName and UnitPrice

# 