Assignment 03

Building a Web Application with ASP.NET Core MVC

Introduction

Imagine you're an employee of a product retailer named **eStore**. Your manager has asked you to develop a Web application for member management, product management, and order management. The application has a default account whose email is "**admin@estore.com**" and password is "**admin@@**" that stored in the **appsettings.json**.

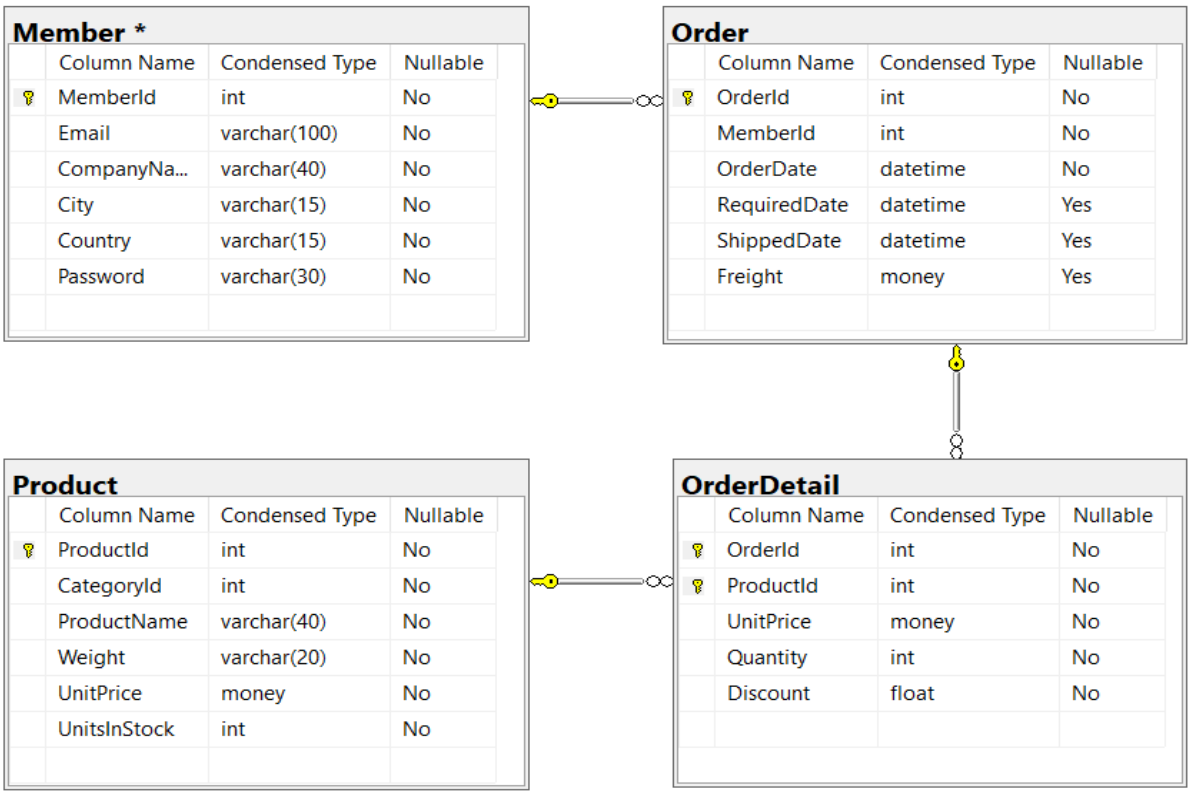
The application has to support adding, viewing, modifying, and removing products—a standardized usage action verbs better known as Create, Read, Update, Delete (CRUD) and Search. This assignment explores creating an application using Windows Forms with .NET Core, C#, and ADO.NET / Entity Framework. An MS SQL Server database will be created to persist the data and it will be used for reading and managing data.

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

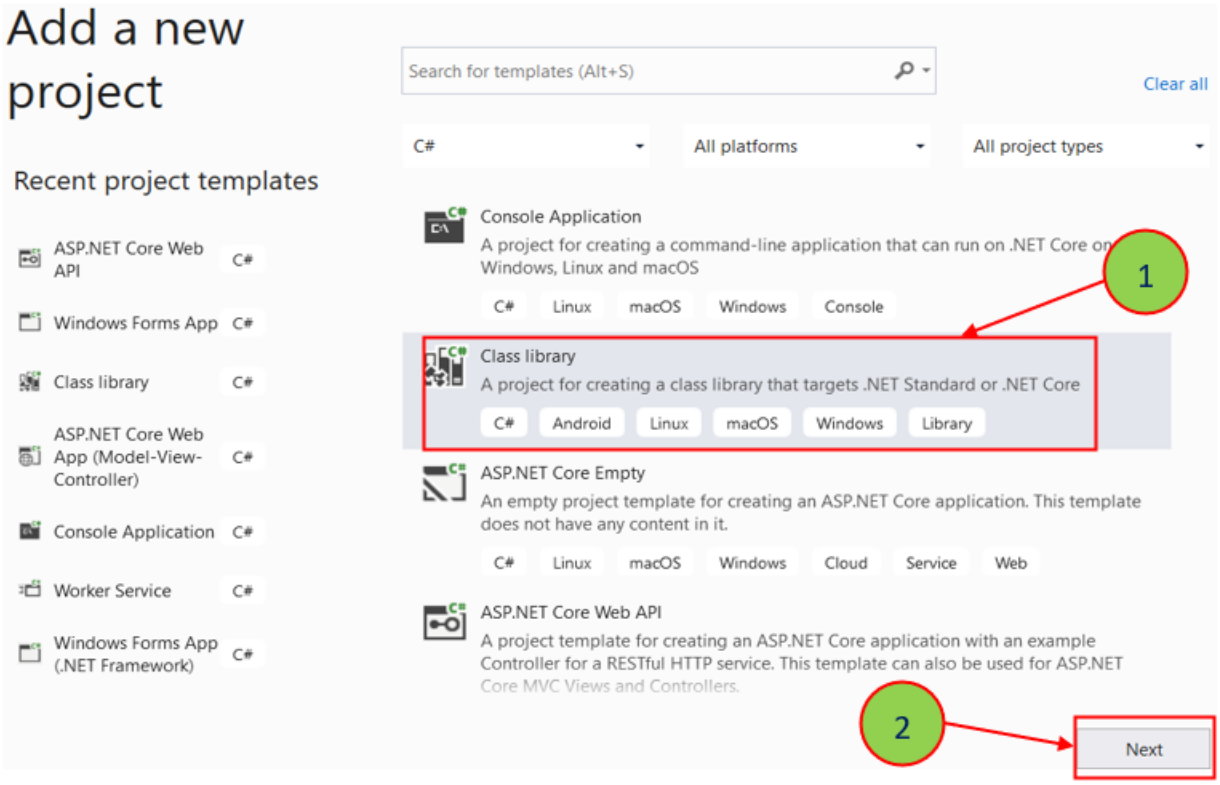
Activity 01: Build a solution

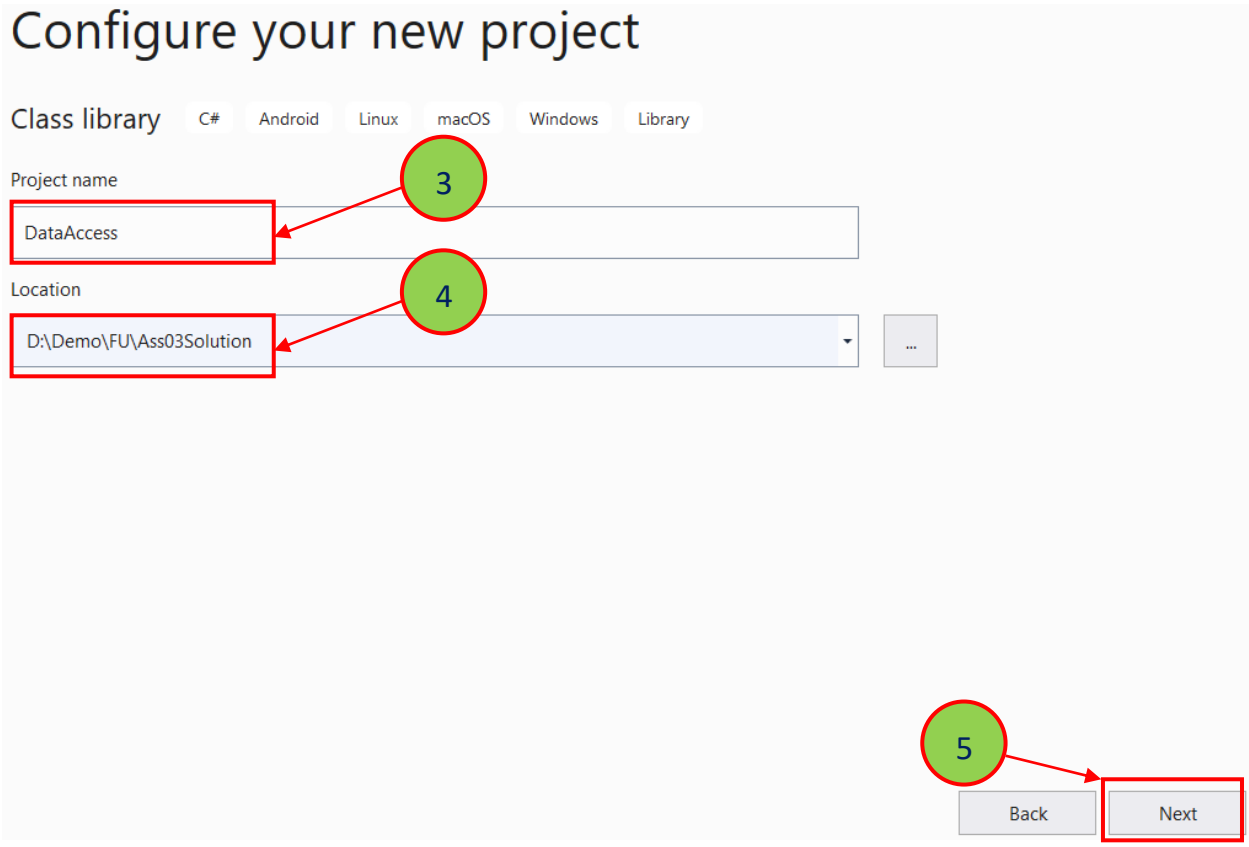
Create a Blank Solution named **Ass03Solution** that includes Class Library Project: **DataAccess, BusinessObject,** and an ASP.NET Core MVC project named **eStore**

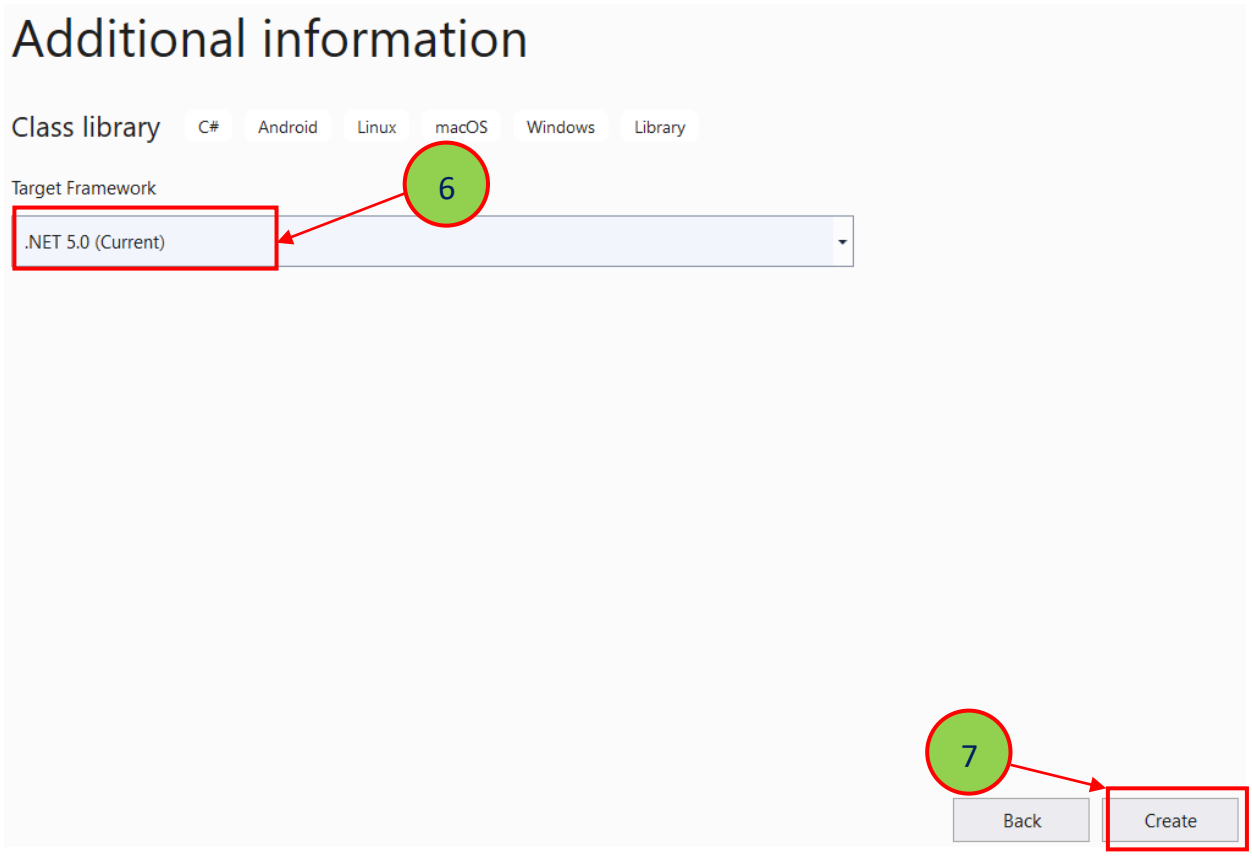
Step 01. Open the Visual Studio .NET application and create a Blank solution named **Asm03Solution**

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:


Add a new project
Search for templates (Alt+S)
Clear all
C#
All platforms
All project types
Recent project templates
ASP.NET Core Web API C#
Windows Forms App C#
Class library C#
ASP.NET Core Web App (Model-View-Controller) C#
Console Application C#
Worker Service C#
Windows Forms App (.NET Framework) C#
Console Application
A project for creating a command-line application that can run on .NET Core on Windows, Linux and macOS
C# Linux macOS Windows Console
Class library
A project for creating a class library that targets .NET Standard or .NET Core
C# Android Linux macOS Windows Library
ASP.NET Core Empty
An empty project template for creating an ASP.NET Core application. This template does not have any content in it.
C# Linux macOS Windows Cloud Service Web
ASP.NET Core Web API
A project template for creating an ASP.NET Core application with an example Controller for a RESTful HTTP service. This template can also be used for ASP.NET Core MVC Views and Controllers.
1
2
Next

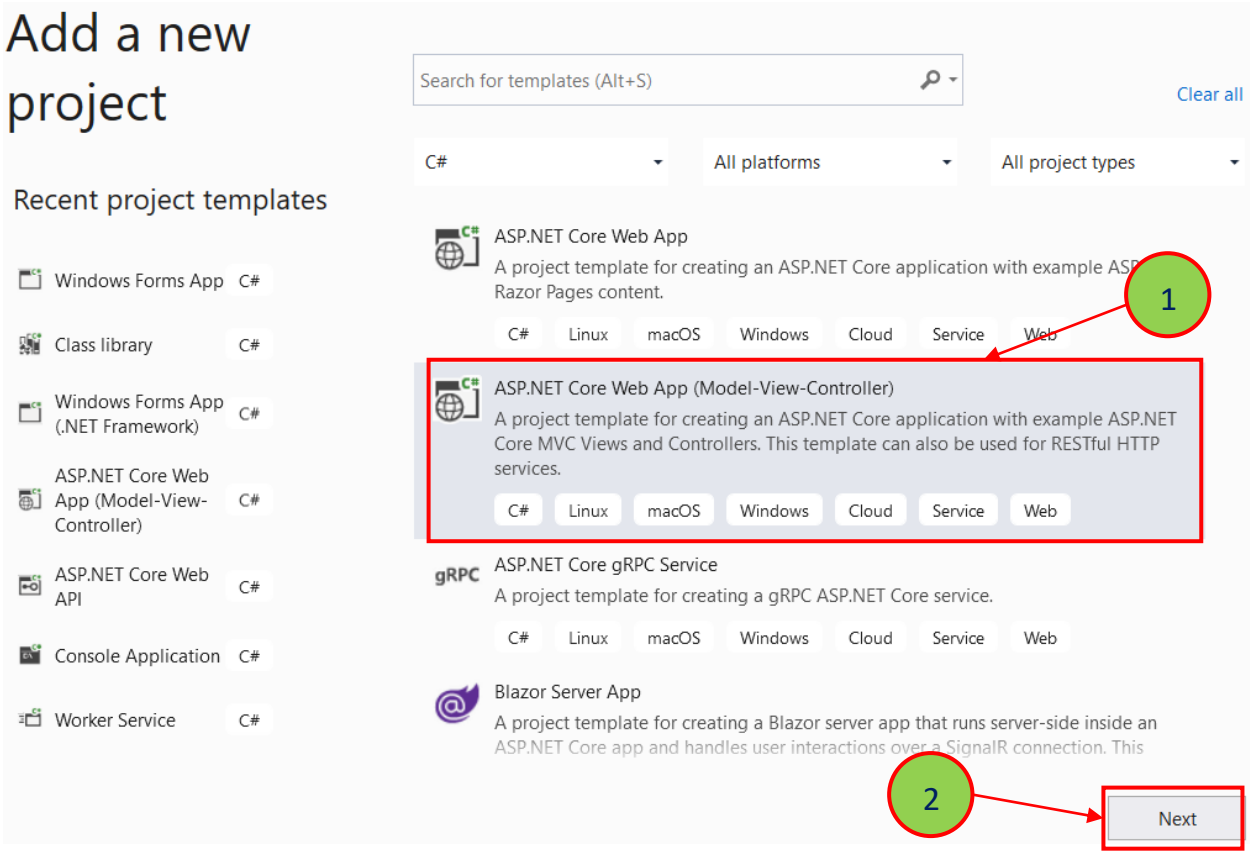

Configure your new project
Class library C# Android Linux macOS Windows Library
Project name
DataAccess
3
Location
D:\Demo\FU\Ass03Solution
4
...
5
Back
Next

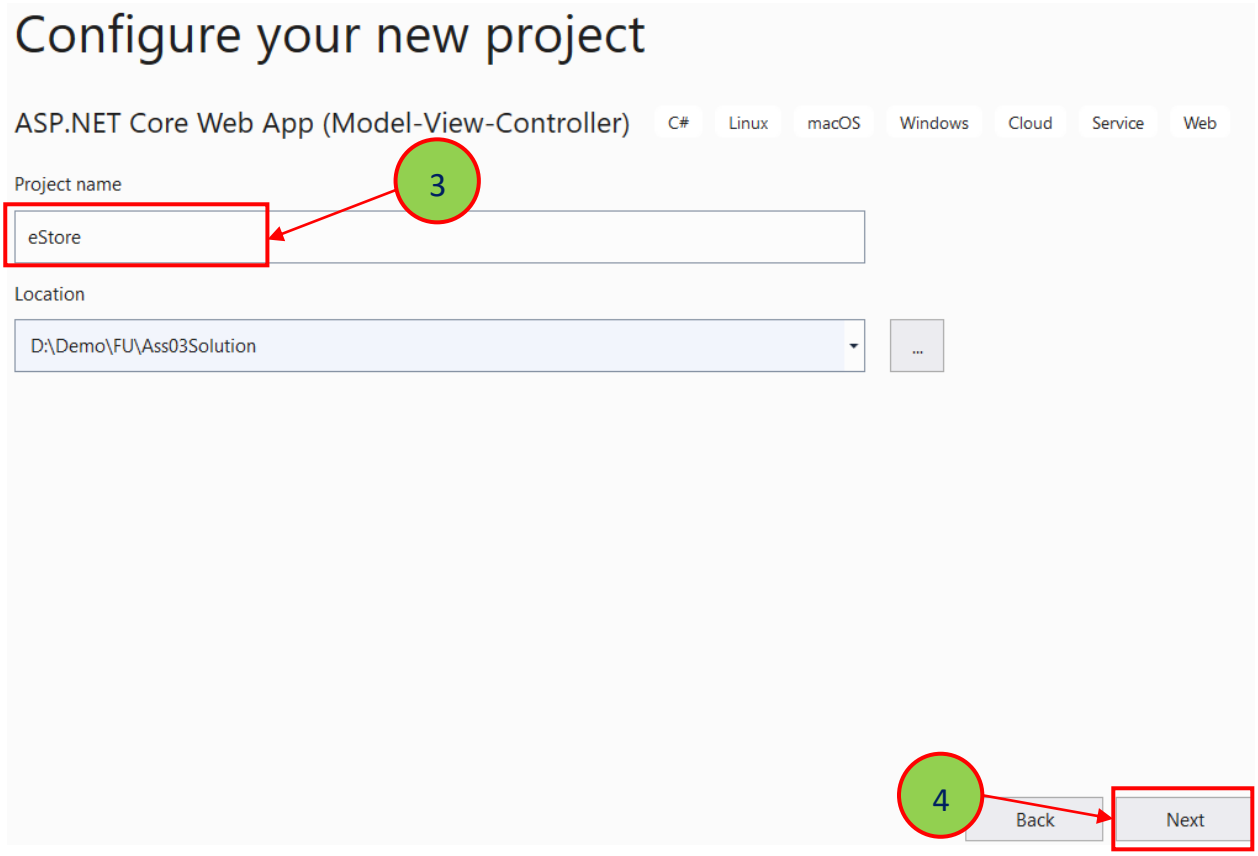


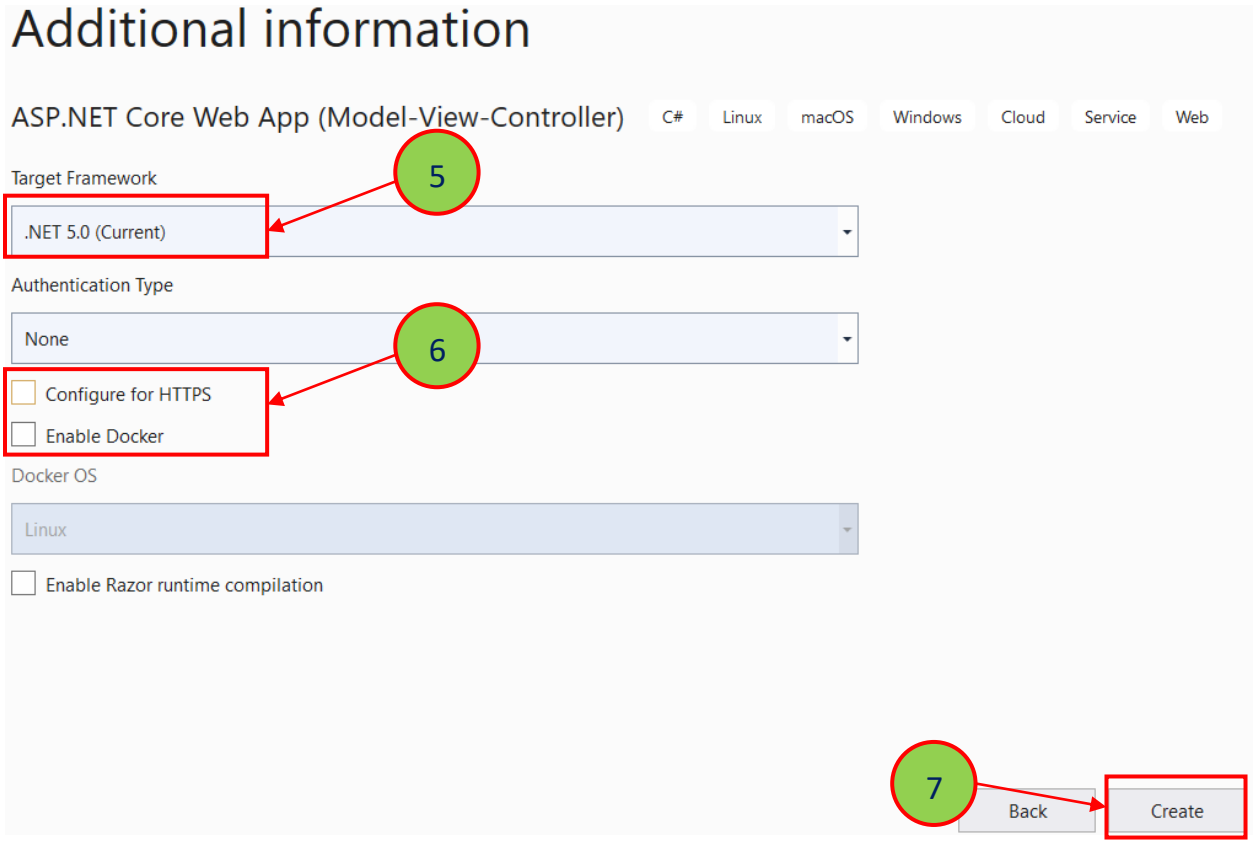
Step 03. Repeat **Step 02** to create a **BusinessObject** project.

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

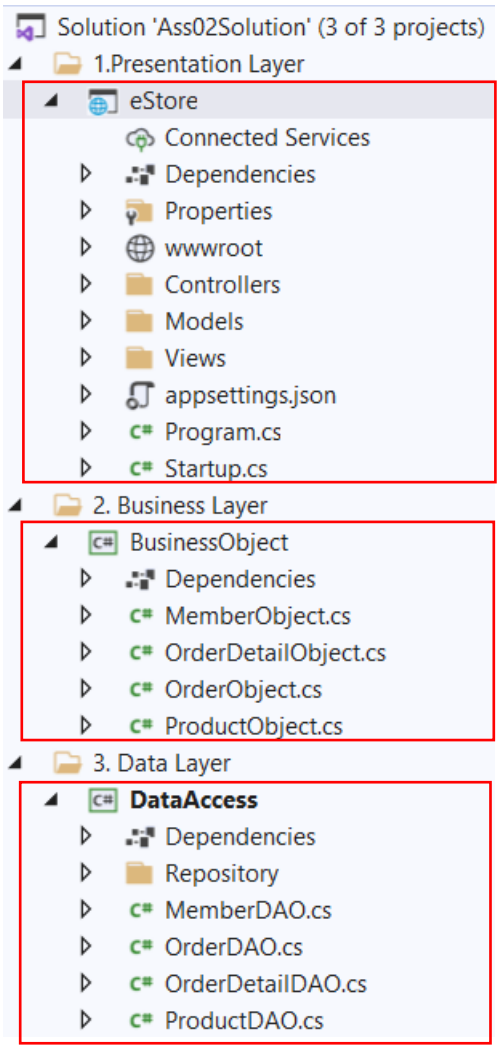
- From the File menu | Add | New Project, on the Add New Project dialog, select "ASP.NET Core Web App (Model-View-Controller)" and performs steps as follows:


Add a new project
Search for templates (Alt+S)
Clear all
C#
All platforms
All project types
Recent project templates
Windows Forms App C#
Class library C#
Windows Forms App (.NET Framework) C#
ASP.NET Core Web App (Model-View-Controller) C#
ASP.NET Core Web API C#
Console Application C#
Worker Service C#
ASP.NET Core Web App
A project template for creating an ASP.NET Core application with example ASP
Razor Pages content.
C# Linux macOS Windows Cloud Service Web
ASP.NET Core Web App (Model-View-Controller)
A project template for creating an ASP.NET Core application with example ASP.NET Core MVC Views and Controllers. This template can also be used for RESTful HTTP services.
C# Linux macOS Windows Cloud Service Web
ASP.NET Core gRPC Service
A project template for creating a gRPC ASP.NET Core service.
C# Linux macOS Windows Cloud Service Web
Blazor Server App
A project template for creating a Blazor server app that runs server-side inside an ASP.NET Core app and handles user interactions over a SignalR connection. This
1
2
Next


Configure your new project
ASP.NET Core Web App (Model-View-Controller)
C# Linux macOS Windows Cloud Service Web
Project name
eStore
3
Location
D:\Demo\FU\Ass03Solution
...
4
Back
Next



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



Activity 02: Develop BusinessObject project

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

***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

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

For example: Search products by ProductName and UnitPrice