**Assignment 03 - Fall 2023 HCM**

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

# 1. Introduction

Imagine you're an employee of a product retailer named **eFURentingCarSystem**. Your manager has asked you to develop a Web application for member management, car information management, and renting transaction management. The application has a default account whose email is “**admin@ eFURentingCarSystem.com**” and password is “**@@admin@@**” that stored in the **appsettings.json**.

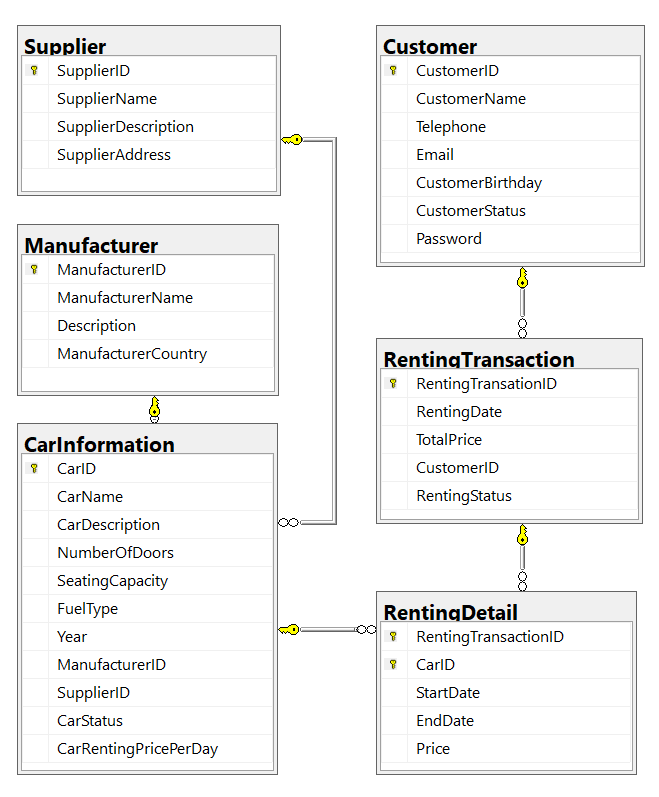
The application has to support adding, viewing, modifying, and removing items—a standardized usage action verb better known as Create, Read, Update, Delete (CRUD) and Search. This assignment explores creating an application using ASP.NET Core Web App with Model View Controller (MVC), C#, and ADO.NET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

# 2. 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.

# 3. Database Design



A car will belong to only one manufacturer and only one supplier.

A customer can make renting transaction in this system many times. A renting transaction will have one or many car information. A car information will belong to many renting transactions.

# 4. Main Functions

* Member (Admin/Customer) authentication by Email and Password. If the user is “**Admin**” *(get from appsettings.json file)* then allows to perform all actions, otherwise, the customer *(get from the Customer table in database)* is allowed to view/create/update the profile and view their renting transactions history.
* Customer management, Car management, and Renting management: Read, Create, Update and Delete actions. Creating and Updating actions must be performed by popup dialog. *One renting transaction will have many cars with difference time (start/end); one car will have in many renting transaction.*
* Search car by ID, name (by keyword of CarName), CarRentingPricePerDay.
* Create a report statistics sales by the period from StartDate to EndDate, and sort the results in descending order.

# 5. Guidelines

# Activity 01: Build a solution

Create a Blank Solution named **Ass03Solution\_ClassCode\_StudentName** that includes Class Library Project: **DataAccessObjects, BusinessObjects, Repositories** and an ASP.NET Core MVC project named **eFURentingManagement**

**Step 01**. Open the Visual Studio .NET application and create a Blank solution named **Ass03Solution\_ClassCode\_StudentName**

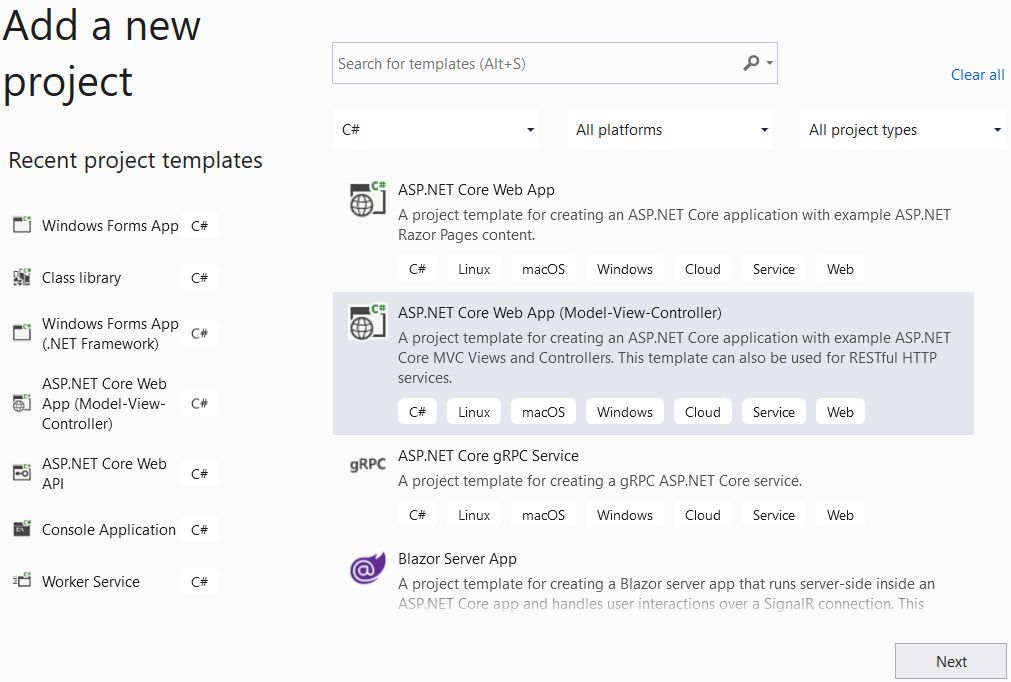
**Step 02.** Create a Class Library project named **DataAccessObjects**

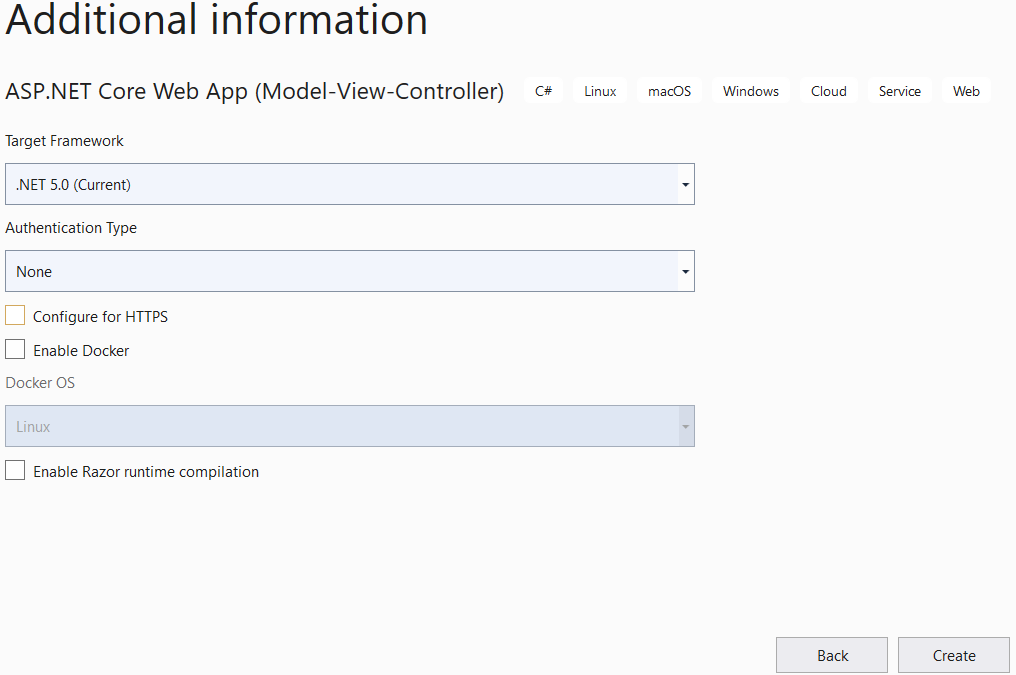
**Step 03.** Repeat **Step 02** to create a **BusinessObjects** project.

**Step 04.** Repeat **Step 02** to create a **Repositories** project.

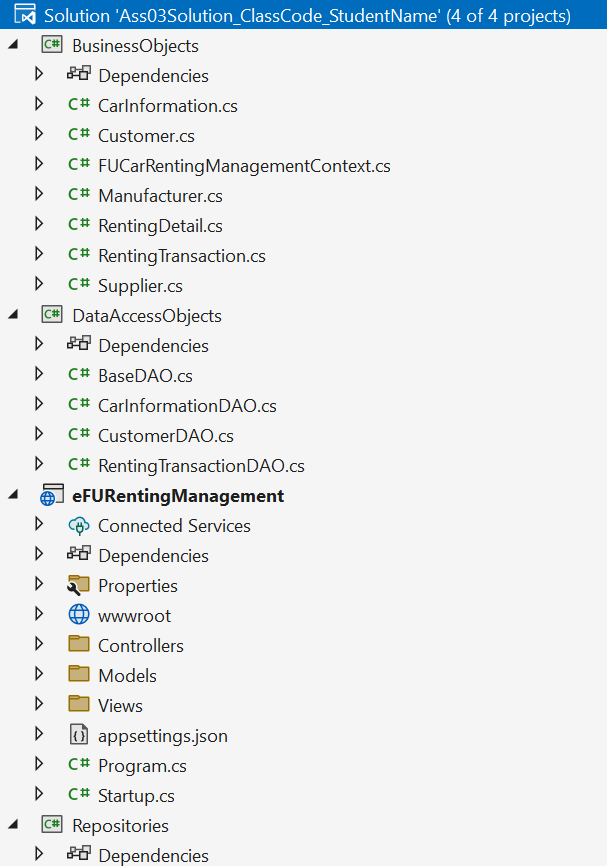
**Step 05.** Create an ASP.NET Core MVC project named **eFURentingManagement**

* 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 BusinessObjects project

# Activity 03: Develop DataAccessObjects project

# Activity 04: Develop Repositories project

# Activity 05: Develop eFURentingManagement project

**Step 01**. Add a reference to **Repositories** project.

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

# Activity 06: Run the Web project and test all actions