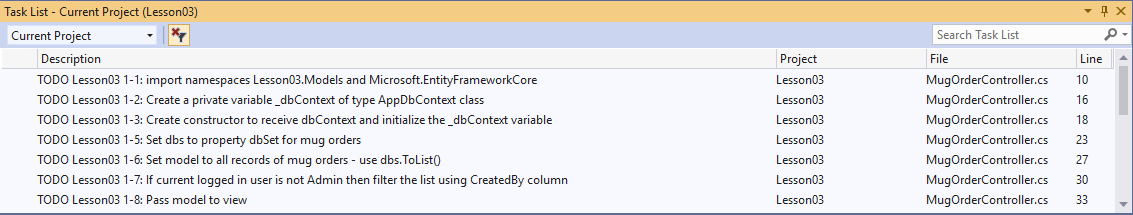
Lesson 03 – Introduction to Entity Framework Core

**Using Task List**

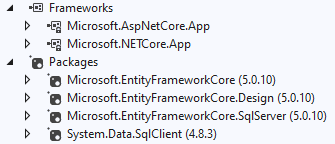
1. Open **Task List** window from the menu **VIEW**, **Task List**. Use the **Filter** function in the Project heading to display only the Task for current project which is Lesson03.



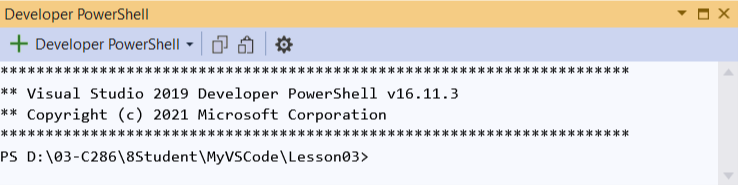
1. **Activity 1:** This activity requires you to setup Entity Framework service, generate entity classes and DbContext class before performing those tasks in the task list
   * + 1. Include the THREE package references (highlighted in yellow) needed to use EF Core with SQL Server in your project by adding the following lines of code in **Lesson03.csproj:**

|  |
| --- |
| <Project Sdk="Microsoft.NET.Sdk.Web">  <PropertyGroup>  <TargetFramework>net5.0</TargetFramework>  </PropertyGroup>  <ItemGroup>  <PackageReference Include="Microsoft.EntityFrameworkCore" Version="5.0.10" />  <PackageReference Include="Microsoft.EntityFrameworkCore.SqlServer" Version="5.0.10" />  <PackageReference Include="Microsoft.EntityFrameworkCore.Design" Version="5.0.10" />  <PackageReference Include="System.Data.SqlClient" Version="4.8.3" />  </ItemGroup>  </Project> |

Press [Ctrl]+[S] to save and check the packages have been downloaded in Solution Explorer.

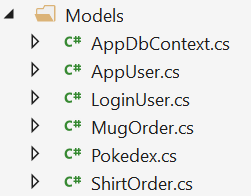


* + - 1. In the Solution Explorer, right-click on **Lesson03** and click Open in Terminal. You should see the **Developer Powershell** window appears in your Visual Studio.



* + - 1. Copy the following one-liner command and paste into the **Developer Powershell** window. To paste is just to right-click within **Developer Powershell**. Press [Enter] to execute the command.

dotnet ef dbcontext scaffold "Data Source=(localdb)\ProjectsV13;Initial Catalog=C286Lesson03;Integrated Security=True" Microsoft.EntityFrameworkCore.SQLServer -o Models -c AppDbContext -f --no-onconfiguring --no-pluralize



Generated through Entity Framework Scaffold-dbcontext command

* + - 1. Dependency Injection step 1 and 2: Use the slides on Dependency Injection to help you configure EF service in **Startup.cs**. The following yellow markers will tell you where to paste the code. The required code is provided below:

|  |
| --- |
| // Add more Using Statements After Here  using Microsoft.EntityFrameworkCore;  using Lesson03.Models;  // Add Dependency Injection Code After Here  services.AddDbContext<AppDbContext>(  options =>  options.UseSqlServer(Configuration.GetConnectionString("DefaultConnection"))); |

* + - 1. Dependency Injection step 3 to 4: Perform **Tasks 1-1 and 1-3**. Use **C286 L03 Student**, the slides on Dependency Injection, as reference.
      2. Perform **Tasks 1-4 to 1-8** to complete activity 1. **Task 1-4** is in Views/MugOrders/Index.cshtml.
      3. Run the application and click **Cadeau Mug Ordering** link to view the MugOrders/Index action. You should observe that the data is displayed properly.

1. **Activity 2:**
   * + 1. Perform **Tasks 2-1 to 2-8**.
       2. **Task 2-1** is in Views/MugOrder/Create.cshtml.
       3. Run the application and test insert a new mug order to verify the order can be successfully inserted to database.
2. **Activity 3:**
   * + 1. Perform **Tasks 3-1 to 3-2**.
       2. All these tasks are in Views/Sales/Index.html
       3. Run the application and view each step to ensure correct result is obtained.

# Solving the Problem

1. You are suggested to complete the solution in the following order:
   * + 1. Dependency injection – create \_dbContext private variable and constructor in ShirtOrderController.
       2. Implement ShirtOrderController/Index action
       3. Implement ShirtOrderController/Create get and post actions
       4. Implement the requirements for Sales Report.

# Testing Your Solution

1. Test Cases:
   * + 1. Add a new shirt order. Verify that the new shirt order is displayed in the index page.

Graphical user interface, application

Description automatically generated

Graphical user interface, application, Teams

Description automatically generated

* + - 1. Run Sales Report and review the result.

ss

Graphical user interface, text, application, email

Description automatically generated

Graphical user interface, application

Description automatically generated

Provide necessary screenshots here to prove you have successfully tested your web app.

*— End of Worksheet —*