Lesson 6 – Modifying Data Part 2

The demonstration web app for this lesson can be found at

[C236 Lesson 6 (c236dotnet.azurewebsites.net)](http://c236dotnet.azurewebsites.net/lesson06)

# Setup DB

1. Open the window **SQL Server Object Explorer** by clicking **View** > **SQL Server Object Explorer**. The following window will appear

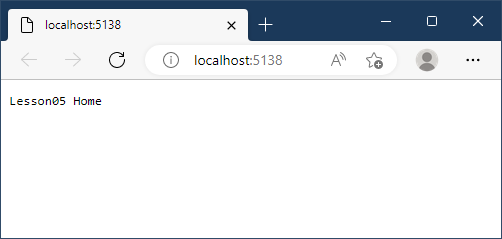
Graphical user interface, text, application, email

Description automatically generated

1. Expand **(localdb)\ProjectsModels**. Do **not** expand MSSQLLocalDB.
2. If you cannot find it, click on the icon **Add SQL Server…**  Expand **Local** and select **ProjectModels**. Click on the [Connect] button.
3. Right-click **Databases** and click **Add New Database**. The **Create Database** window will appear.
4. In the **Database Name** enter **DB06**
5. In the **Database Location** , select the folder **Databases** in your C236 folder which you have created in Lesson 01.
6. Right-click on **DB06**, click **New Query.** The **SQLQuery1.sql** window will appear.
7. Copy the SQL statements given in **DBSetup.txt** and paste inside SQLQuery1.sql.
8. Click the **green** execute button at the top left corner of the SQLQuery1.sql window to execute the SQL statements.
9. Right-click on **DB06**, click **Refresh** . Expand the **Tables**. You will see the four tables: **BwBook, BwPublisher**, **OrgProduct** and **OrgSubscription.**
10. Close the **SQLQuery1.sql** window. You do not need to save any changes.

# Setup Application

1. In the Solution Explorer, right-click on the Project **Lesson06** > Set as Startup Project.
2. Right-click **Lesson06** and select **Build**
3. Run the project.



1. From the **Models** folder inthe lesson materials, **copy all files** into the **Models** folder in VS2022 Lesson06. Do not drag in the whole folder, or you will create Lesson06/Models/Models, which will not work.
2. From the **Views** folder in the lesson materials, **copy all the sub-folders and contents** into the **Views** folder in VS2022 Lesson06. Do not drag in the whole folder.
3. From the **Controllers** folder in the lesson materials, copy **all the files** into the **Controllers** folder in VS2022 Lesson06.
4. Drag the **images folder** and the **photos folder** from the lesson materials **onto the** **wwwroot** folder in VS2022 Lesson 06.
5. Drag the **utils folder** from the lesson materials **onto** **Lesson06** in VS2022 Lesson 06.
6. Drag the **libman.json file** from the lesson materials **onto** **Lesson06** in VS2022. Right-click **libman.json > Restore Client-Side Libraries**. This step creates **wwwroot/lib** folder structure containing the necessary bootstrap libraries.
7. Modify the **Program.cs** file by adding the following line to the very top of the file, then save the file.

global using Lesson06.Models;

global using System.Data;

global using RP.SOI.DotNet.Utils;

1. Double click Lesson06 to open the Lesson06.csproj file. Amend the file as shown.

|  |
| --- |
| <Project Sdk="Microsoft.NET.Sdk.Web">  <PropertyGroup>  <TargetFramework>net6.0</TargetFramework>  <Nullable>enable</Nullable>  <ImplicitUsings>enable</ImplicitUsings>  <AspNetCoreHostingModel>OutOfProcess</AspNetCoreHostingModel>  <AspNetCoreModuleName>AspNetCoreModule</AspNetCoreModuleName>  <TreatWarningsAsErrors>true</TreatWarningsAsErrors>  <RootNamespace>Lesson06</RootNamespace>  </PropertyGroup>  <ItemGroup>  <PackageReference Include="System.Data.SqlClient" Version="4.8.3" />  </ItemGroup>  </Project> |

1. Open **Appsettings.Development.json** (expand **Appsettings.json** to see the file). Amend the file as shown.

|  |
| --- |
| {  "Logging": {  "LogLevel": {  "Default": "Information",  "Microsoft.AspNetCore": "Warning"  }  },  "ConnectionStrings": {  "DefaultConnection": "Data Source=(localdb)\\ProjectModels;Initial Catalog=DB06;Integrated Security=True"  }  } |

1. Save all files (Ctrl-Shift-S).
2. Right-click the project **Lesson06** and select **Build**. Ensure there are no errors.
3. Run the project with debugging.

Graphical user interface, text, application, chat or text message

Description automatically generated

CONTINUE ONLY IF YOUR PROJECT IS ERROR-FREE.

# HTML attributes affecting input

1. Run your project, add the controller **demo** and action **textfields** to your Browser's URL. Click the [Submit] button.

Graphical user interface

Description automatically generated

Examine the code for the **Views/Demo/**TextFields.cshtml and DemoController's actions Textfields() and TextFieldsPost(). Reference the HTML types.pdf document included in the student package.

|  |  |  |
| --- | --- | --- |
| a. | What is the Name for each field | Text1, Text2, Text3 |
| b. | What is the attribute inside the element that makes it mandatory? | required |
| c. | What is the attribute inside the element that makes it non-editable? | read-only |
| d. | What is the attribute inside the element that hides it from view? | hidden |
| e. | Which attribues are used to ensure that only positive integers can be entered? | type=”number” min=”0” |
| f. | Which attributes are used to ensure that only integers or decimal values can be entered? | type=”number” min=”0” and step=”any” |
| g. | Is there any difference in the controller to retrieve a value of a normal, readonly hidden text field? | No |
| h. | If all fields are required, why are not all marked with the required attribute? | Hidden and read only fields does not require required attribute. |
| i. | In which order are the fields validated? | Top to bottom |
| j. | With reference to the **HTML Types pdf**, list three other input types which you believe may be of use in the future. | “number”, “date” and “file” |

# Organic Fruits

Double click on the file **launchSettings.json** in L06/Properties folder. Add the following line after the line the contains "IISExpress" (Line 19). Save the file. Run your project.

|  |
| --- |
| "launchUrl": "organic/products", |

Try to create a new product (e.g. Grapes)

|  |  |  |
| --- | --- | --- |
| a. | What is the controller/action for the Web page to create a new product? | Organic/ ProductAdd |
| b. | After clicking the [Add] button, which controller/action you are redirected to? | Organic /Products |
| c. | Is there any message to tell you that the product has been successfully created? | Product Added |

Try to edit the product you created in the previous step.

|  |  |  |
| --- | --- | --- |
| a. | What is the controller/action for the Web page to edit the product? | Organic /ProductEdit |
| b. | Are you able to change the entries for all fields? | No |
| c. | Which field you cannot change and why? | Description, readonly attribute |

Try to delete the product which you have edited in the previous step.

|  |  |  |
| --- | --- | --- |
| a. | What is the controller/action to delete the product? | Organics/ ProductDelete |
| b. | Does the Web app confirm with you before the deletion? | Yes |
| c. | After the deletion, which controller/action you are redirected to? | Organics /Products |
| d. | Is there any message to tell you that the product has been successfully deleted? | Yes |

Open Products.cshtml in Views/Organic folder.

|  |  |  |
| --- | --- | --- |
| a. | Find the code that displays the price of the product in currency format ($0.00) | <td style="text-align:right">@String.Format("{0:c}", row["Price"])</td> |
| b. | Find the code that displays the "gms" after the weight of the product | <td style="text-align:right">@String.Format("{0} gms", row["Gram"])</td> |
| c. | Copy the hyperlinks for the **Edit** and **Delete** actions. | <a href="/Organic/ProductEdit/@row["OrgCode"]"  style="text-decoration:none; color:dodgerblue">  <text class="bi bi-pencil"></text>  </a> |  <a href="/Organic/ProductDelete/@row["OrgCode"]"  style="text-decoration:none; color:indianred"  onclick="return confirm('Delete product @row["OrgDesc"]?')">  <i class="bi bi-trash3"></i> |
| d. | What does text-decoration:none do? | Remove underline for text based hyperlinks. |

Open ProductEdit.cshtml in Views/Organic folder.

|  |  |  |
| --- | --- | --- |
| a. | What is the **Model** for this **View**? | OrgProduct |
| b. | What **attribute** inside the <input> element is used to display the **property values** of the model on the Web page? | value |
| c. | Which **attribute** inside the <input> element is responsible to make the **Description** uneditable? | readonly |
| d. | What is the purpose of **step** in the price control? | To allow end-user to type in decimal places when updating price of item |
| e. | Which attribute prevents negative numbers? | min=”0” |

Open **OrganicController.cs** in **Controller** folder. The regions "Product Add", "Product Edit" and "Product Delete" may prove useful when you are solving the problem.

# Solving the Problem

## Travel/Index

Run your project, add the controller (**travel**) and action (**index**) to your Browser's URL. Compare this Web page with [Where We Were (c236dotnet.azurewebsites.net)](http://c236dotnet.azurewebsites.net/lesson06/travel). Write down the differences here. The first difference has been done for you.

|  |
| --- |
| The **Period** column. LocalHost displays the full date/time. Azure displays only year and month.  Picture canot be reader  The story column is cut off |

Open the files **Main.cshtml** and **Add.cshtml** in the folder **Views** > **Travel**. Open the **Task List** window from **View** menu and then click **Task List**. In the drop down list at the top, select **Current Project**.

Graphical user interface

Description automatically generated with low confidence



Perform Task 1. Complete the extension method Abbreviate. The code can be found in the file **C236 L06 Student.pdf**.

Run your project, add the controller (**travel**) and action (**index**) to your Browser's URL. You should see the abbreviated story appears in the column.

Open the file **main.cshtml** in Views/Travel folder. Perform Task 2a-e. When you have completed these tasks. Repeat the earlier comparison. There are to be no differences in output format.

## Travel/Add

Perform Task 3a-c to complete the code for the **AddPost** action portion in the **TravelController.cs**.

Open **Add.cshtml** in **Views/Travel** folder. Perform Task 4a-c to complete the code for the view portion.

Run your project, add the controller (**travel**) and action (**add**) to your Browser's URL. Inspect that the view has all the fields and test whether a record can be inserted into the database.

## Travel/Edit

Perform Task 5a-b to complete the code for the **EditPost** action portion in the **TravelController.cs**.

Run your project, add the controller (**travel**) and action (**index**) to your Browser's URL. In the Title column, click on any hyperlink to edit the story. Test whether the story can be edited and saved successfully to the database.

## Travel/Delete

Perform Task 6a-b to complete the code for the **Delete** action portion in the **TravelController.cs**.

Run your project, add the controller (**travel**) and action (**index**) to your Browser's URL. In the Action column, click on any bin icon to delete the corresponding trip record.

*— End of Worksheet —*