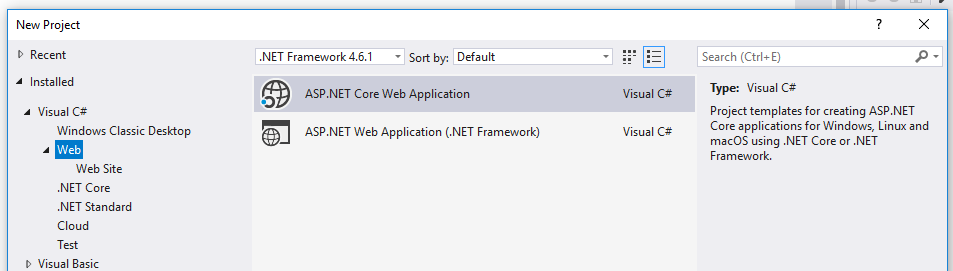
**Activity: ASP.NET CORE 2.1 MVC Web App with a supplied "Northwind" database.**

Create a container folder for this web app in your working folder for this course.  
Folder name: **MVCNorthwindTest01**

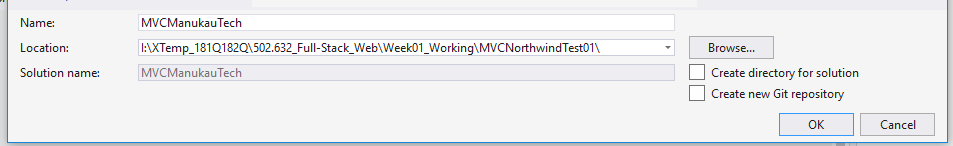
Start Visual Studio

Menu Bar -- File -- New -- Project

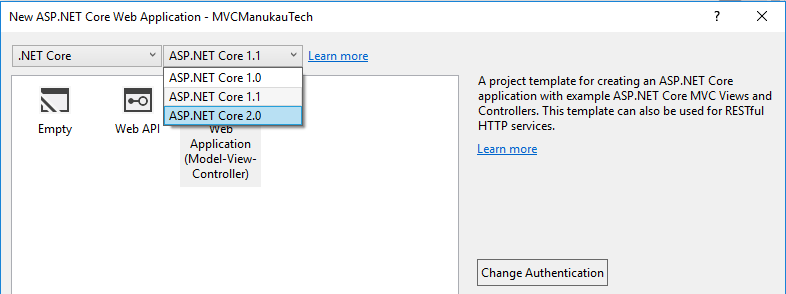
In the "New Project" dialog (popup window), select "Visual C#" -- "Web" -- "ASP.NET Core Web Application".



At the bottom of the dialog:  
For "Location", click the "Browse" button and navigate to the folder "MVCNorthwindTest01" that you have created.  
Do NOT tick (check) the checkbox "Create directory for solution" because we have already created our container.  
For Name: enter **MVCManukauTech** -- this will be the standard name and "namespace" for all our applications.  
Click the "OK" button.

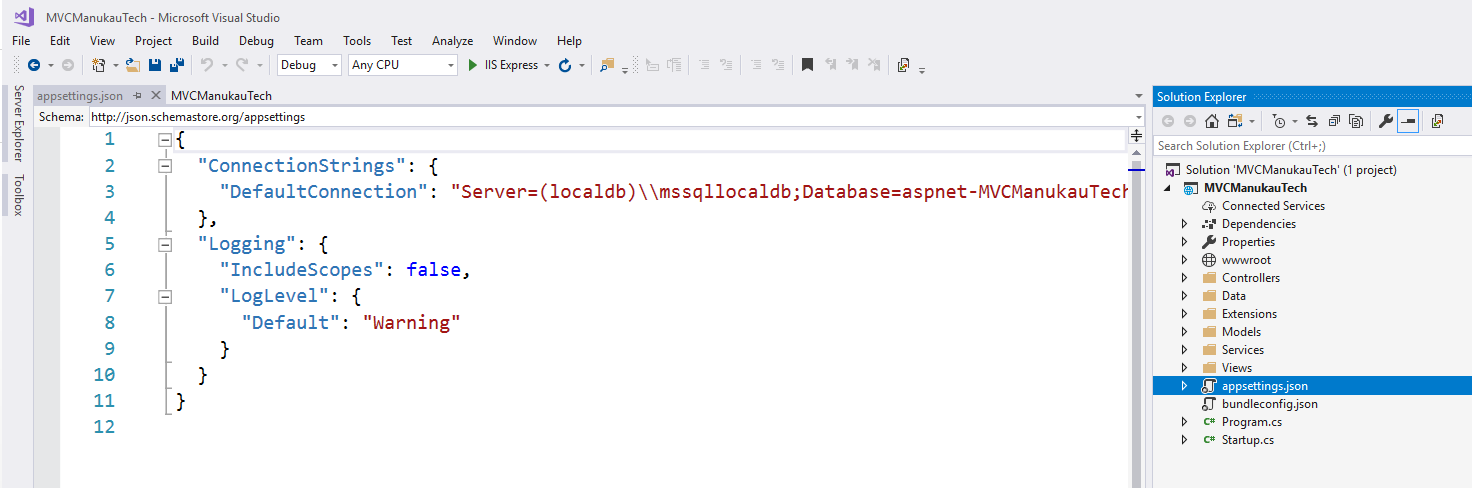


Dialog: New ASP.NET Core Web Application - MVCManukauTech  
Select "**ASP.NET Core 2.1"** if you have it, otherwise "**ASP.NET Core 2.0"** is OK for the early part of this course.  
Select **"Web Application (Model-View-Controller)"**  
Click on the **"Change Authentication"** button and in its popup, select **"Individual User Accounts"**   
-- OK Button -- OK Button



In the "Solution Explorer" panel on the right hand side of Visual Studio, select "appsettings.json".

This gives us a configuration file used to start this web app.  
It includes a "Connection String" which is code to connect to a database.  
We are going to change this to work with our database server.



Replace the Connection String with the one the lecturer gives you. This example is for Q3 2018:

Server=citizen.manukautech.info,6303;Database=Northwind;UID=tron01;PWD=inVoice[22];Encrypt=true;TrustServerCertificate=true;

To give a result like this:

{

"ConnectionStrings": {

"DefaultConnection": "Server=citizen.manukautech.info,6303;Database=Northwind;UID=tron01;PWD=inVoice[22];Encrypt=true;TrustServerCertificate=true"

},

"Logging": {

"IncludeScopes": false,

"LogLevel": {

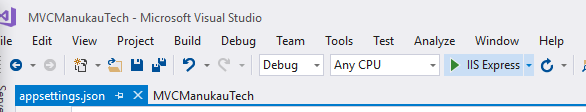
"Default": "Warning"

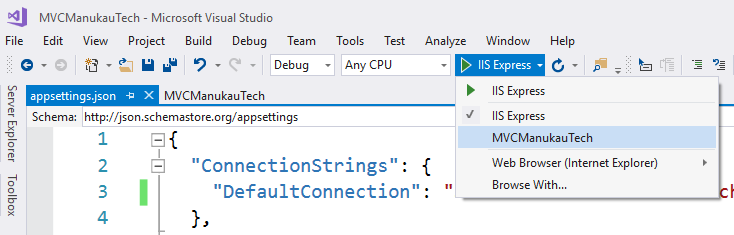
}

}

}

Run by clicking on the "IIS Express" button in the top toolbar.

  
A browser should start showing a test website. If that does not happen, use the dropdown control  
to change from "IIS Express" to the project name, in this case "MVCManukauTech".  
Then try that run again. My lecturer laptop needs 2 or 3 attempts but most student laptops are better.

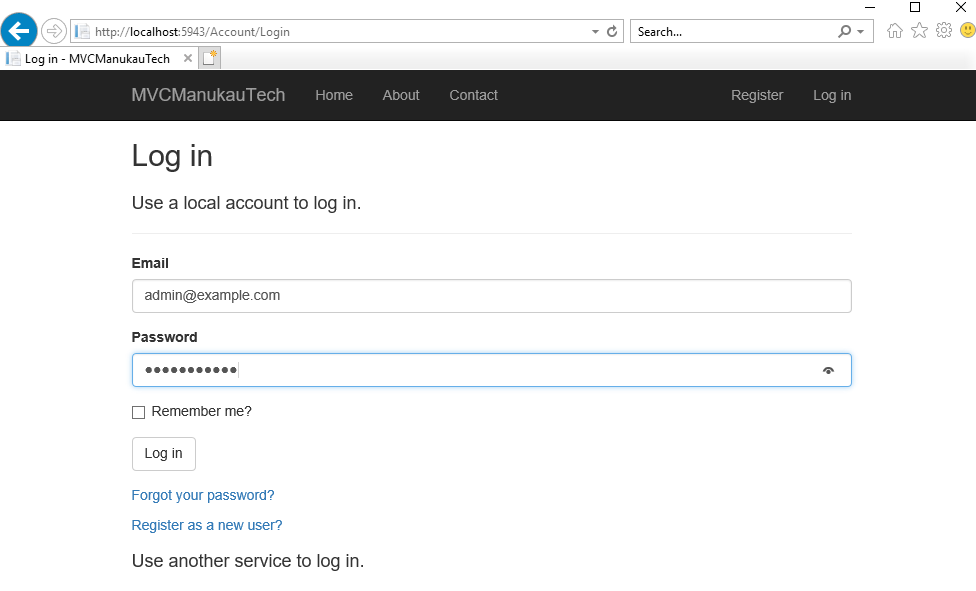


If you still have trouble, dropdown again and select "Web Browser (Internet Explorer)" or "Browse With … Google Chrome". These options switch off some Visual Studio debugging to make the session simpler.

Run the app. Test that login works. Test admin user is:

Username: admin@example.com

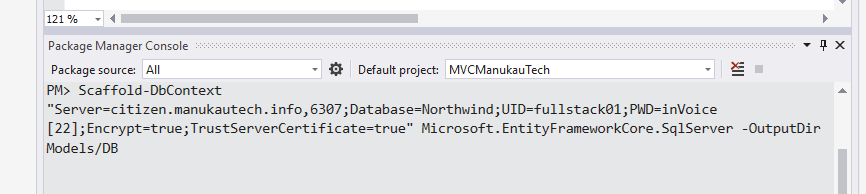
Password: inVoice[22]



Generating Classes for working with our Database Tables.

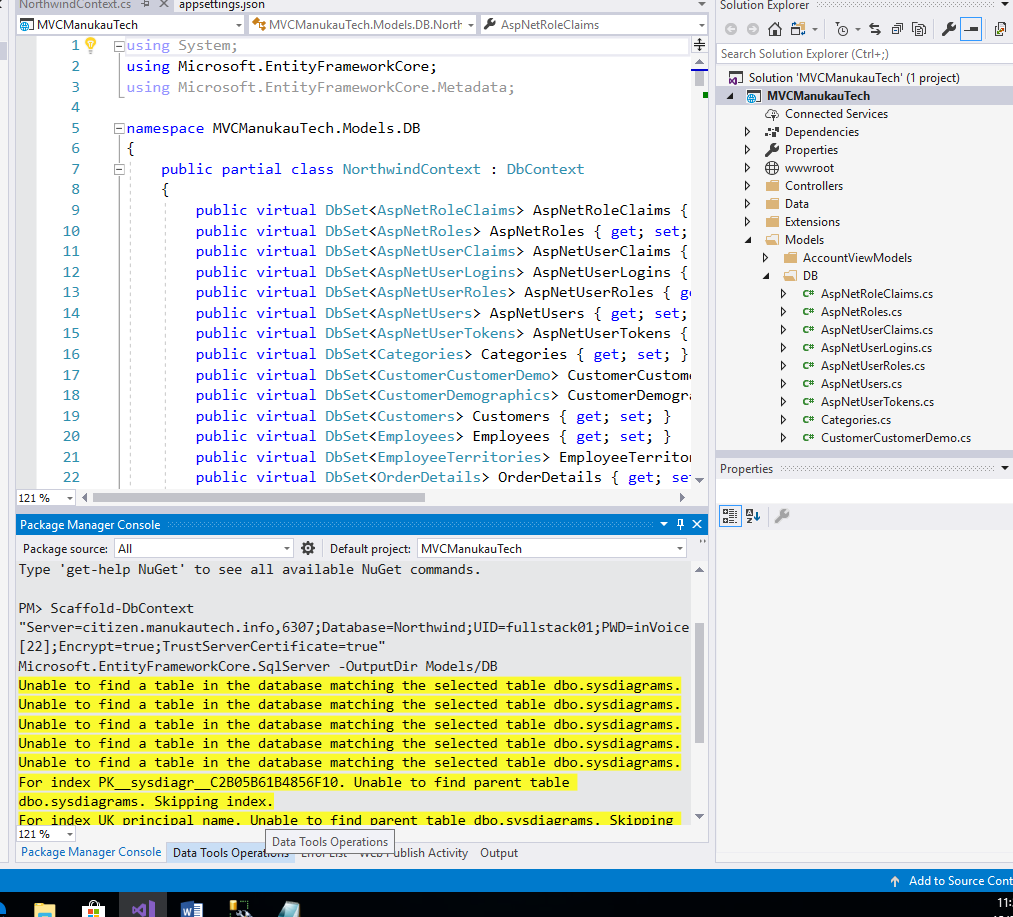
ASP.NET CORE 2 has a Scaffold-DbContext command that can scan our database tables and "generate" = automatically write code to create Classes for them.  
We now need to run this as a command from a console panel in Visual Studio.   
Open "Package Manager Console" from Menu:View -- Other Windows.  
  
Copy and paste this template command into the console after the prompt "PM >"  
Scaffold-DbContext "Server=citizen.manukautech.info,6303;Database=CDQ3\_Michael\_Music;UID=CDQ3\_Michael;PWD=fBit$80726;Encrypt=true;TrustServerCertificate=true" Microsoft.EntityFrameworkCore.SqlServer -OutputDir Models/DB

In this command, "<ConnectionString>" is a placeholder. We need to delete it and replace it with the connection string for our app that we setup before. We did this before in file "appsettings.json" so we can now copy and paste our connection string from there. This will look like this screen shot - but note that the connection string changes for every class so this screen shot is only a guide.



|  |
| --- |
| Note for future reference only. This is in the "Powershell" language which has some reserved characters like "$".  If we have any of these causing trouble then we need to "escape" them with a "backquote" character. |

We see some yellow-highlighted warning notes. Don't worry, these refer to system tables that we do not need.  
The main result is a set of data table classes in sub-folder Models\DB



Like many generators this result, although impressive, needs some cleaning up to work.

**Cleanups needed for Scaffold-DbContext**

In the master class for all of the database, in this example "NorthwindContext.cs", we need to:

**DELETE this method "OnConfiguring":**

protected override void OnConfiguring(DbContextOptionsBuilder optionsBuilder)

{

if (!optionsBuilder.IsConfigured)

{

#warning To protect potentially sensitive information in your connection string, you should move it out of source code. See http://go.microsoft.com/fwlink/?LinkId=723263 for guidance on storing connection strings.

optionsBuilder.UseSqlServer(@"Server=citizen.manukautech.info,6303;Database=Northwind;UID=tron01;PWD=inVoice[22];Encrypt=true;TrustServerCertificate=true");

}

}

------------------------------------------------------------------------------

We also need to get the Northwind Database started in file "startup.cs"

We already have this code in there:

services.AddDbContext<ApplicationDbContext>(options =>

options.UseSqlServer(Configuration.GetConnectionString("DefaultConnection")));

services.AddDefaultIdentity<IdentityUser>()

.AddEntityFrameworkStores<ApplicationDbContext>();

This creates a wide scope "ApplicationDbContext" for connection to the database but this is in use by the Microsoft Identity Login system. It is difficult to share this object with our business data which connects in a different way. It is easier to create a separate connection object.

We can do that by copying and pasting the code for the first statement, then replace "ApplicationDbContext" with "NorthwindContext".  
This will give a compiler warning. Fix this by adding a new "using" statement up top.

using MVCManukauTech.Models.DB;

Giving a result like this:

services.AddDbContext<ApplicationDbContext>(options =>

options.UseSqlServer(Configuration.GetConnectionString("DefaultConnection")));

services.AddDefaultIdentity<IdentityUser>()

.AddEntityFrameworkStores<ApplicationDbContext>();

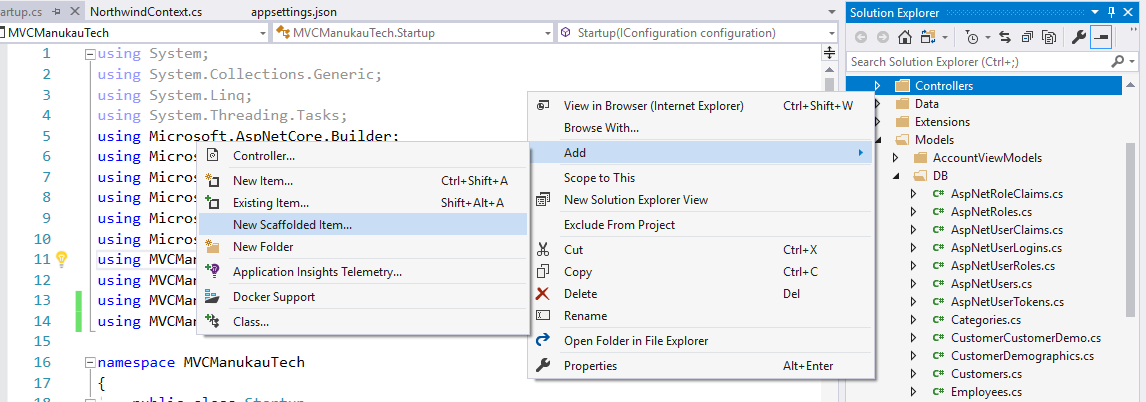
services.AddDbContext<NorthwindContext>(options =>

options.UseSqlServer(Configuration.GetConnectionString("DefaultConnection")));

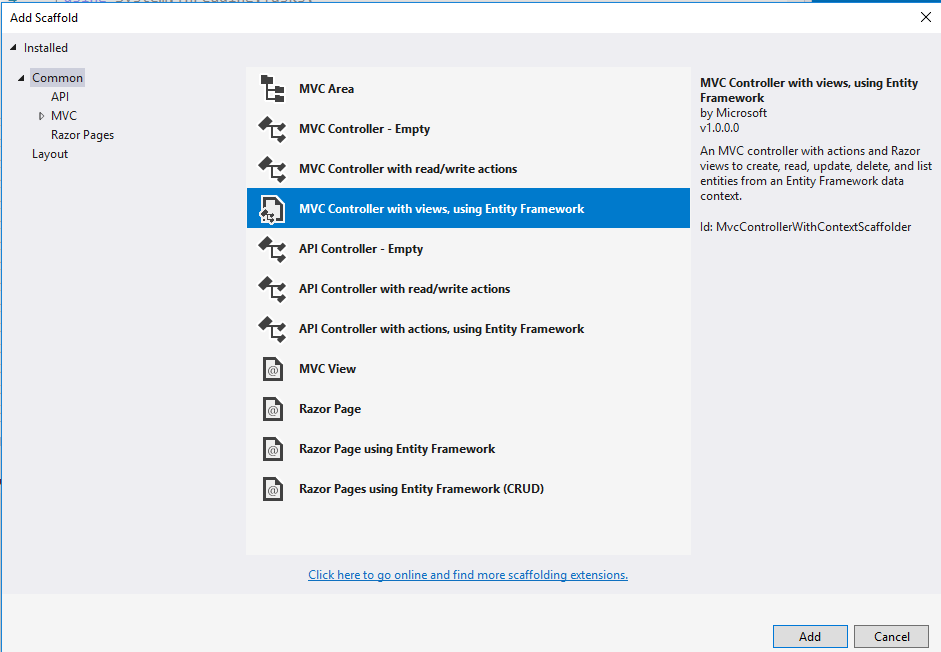
Test run the app which will "build" it.  
We do not expect to see anything new yet. That should appear after the next set of tasks.

**Products - Scaffolding the Controller and Views**

In the "Solution Explorer" panel, right-click on "Controllers" and Add -- New Scaffolded Item



In the "Add Scaffold" dialog, select "MVC Controller with views, using Entity Framework".   
Then click the "Add" button.

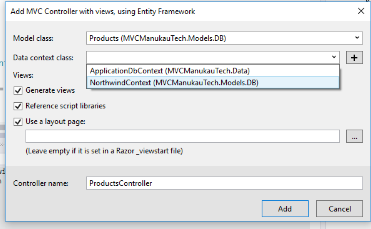


**MVC Controller with views, using Entity Framework**

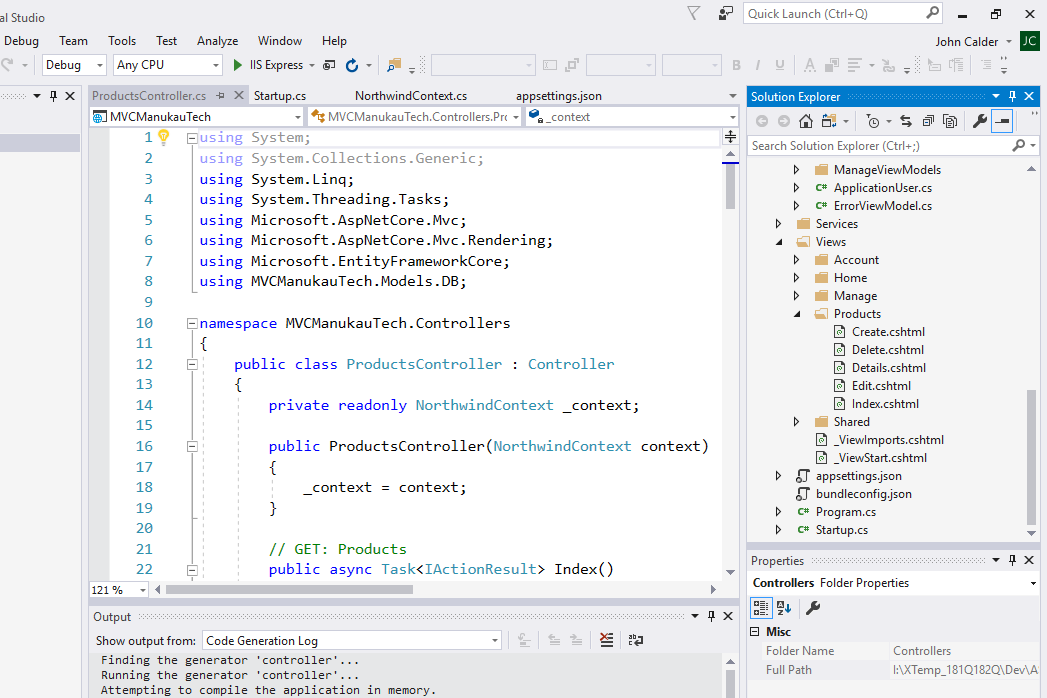
Select these options:  
Model class is "Products". Data context class is "NorthwindContext"

Tick (check) all the Views checkboxes.   
Leave the layout page empty because we are using the default provided by the app generator.

Controller name: RENAME the provided placeholder "DefaultController" to "ProductsController".

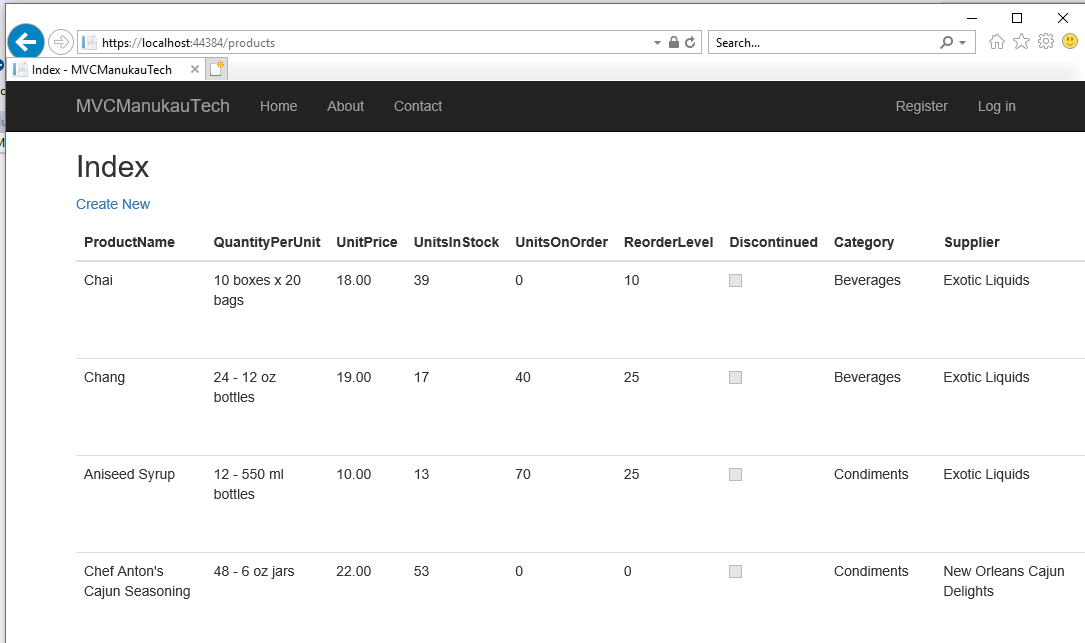


Code generates: Result should look something like this:



**Run the App**  
Manually type the string "Products/" on the end of the address in the browser address bar then press Enter.  
eg: <https://localhost:44384/> -- after editing is -- <https://localhost:44384/Products/>

Some browsers may not end the address with "/" - in that case add "/Products/"

We should see this result, a list of products with details from the database.  
Scroll left to see controls like "Details", "Edit", "Delete". Try "Details" and "Edit".  
Also try "Create New".  
  


**Add menu link.**  
It would be good to have "Products" in the menu bar.  
Under "Views" -- "Shared" open **"\_Layout.cshtml"**

Add code to the menu as a new <li> item to give the result below.  
For the comment, use today's date and your initials - not my old comment!

This is a "razor" page which mixes html and C# scripting and has file extension ".cshtml".   
It uses @\* comment goes here \*@ for comments.

<ul class="nav navbar-nav">

<li><**a** **asp-area**="" **asp-controller**="Home" **asp-action**="Index">Home</**a**></li>

<li><**a** **asp-area**="" **asp-controller**="Home" **asp-action**="About">About</**a**></li>

<li><**a** **asp-area**="" **asp-controller**="Home" **asp-action**="Contact">Contact</**a**></li>

@\*20180216 JPC add Products item to main menu\*@

<li><**a** **asp-area**="" **asp-controller**="Products" **asp-action**="Index">Products</**a**></li>

</ul>

Test Run

End of Activity.