Activating AspCoreGen 3.0 MVC Pro Plus

Note: You need internet connection to activate AspCoreGen 3.0 MVC. You can install one (1) license of AspCoreGen 3.0 MVC for up to 2 computers you own or use for work. You need one (1) license per developer and the license cannot be shared with other developers.

The first time you open AspCoreGen 3.0 MVC Professional Plus edition you will be presented with an activation form right after the Splash screen as shown in Figure 1. You will not be shown the activation form when using the Express edition.

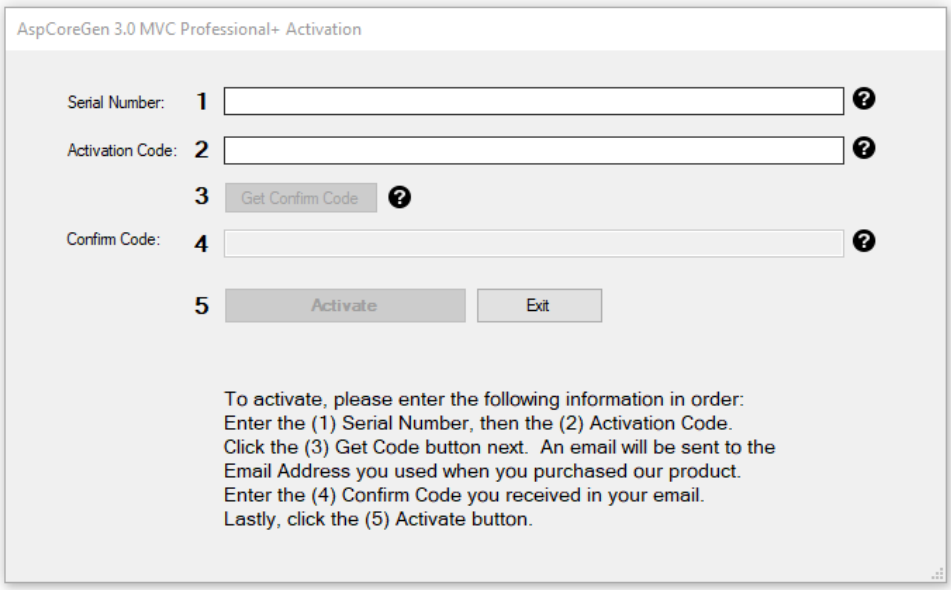

Figure 1 Activation Form

Enter the *Serial Number (1)* and *Activation Code (2)* we provided you in the respective text boxes. The *Serial Number* and *Activation Code* contains dashes "-", make sure to include these dashes when entering them. Then click the *Get Code* button to get the *Confirm Code*. See Figure 2.

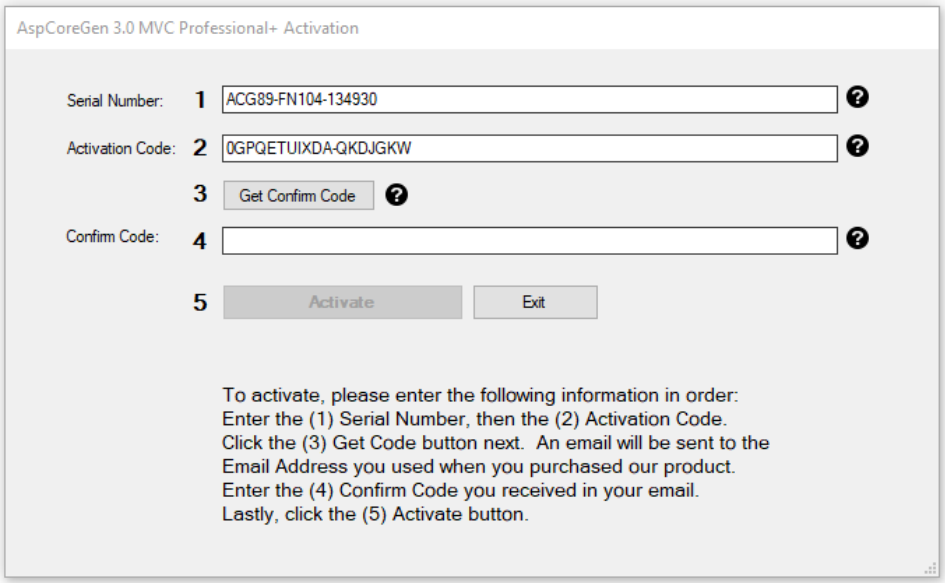

Figure 2

Click the *Get Confirm Code (3)* button next. An error will pop up when you enter an invalid *Serial Number* or *Activation Code*. See Figure 3.

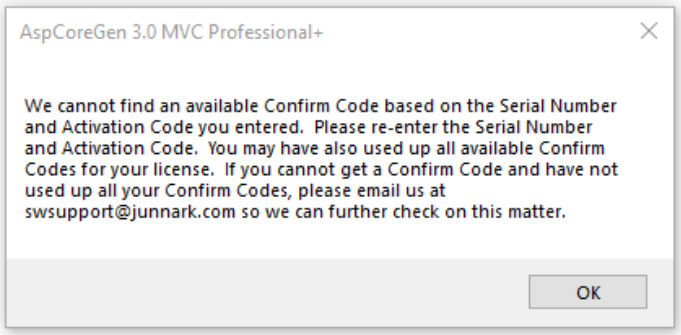

Figure 3 Invalid Serial Number or Activation Code

When this happens, click *OK* to close this message. Re-enter the correct *Serial Number* and *Activation Code* and then click the *Confirm Code* button. The *Confirm Code* will be sent via email to the original purchaser's email address we have on file. When the *Confirm Code* is sent, a pop up message will show as seen in Figure 4. Click the *OK* button.

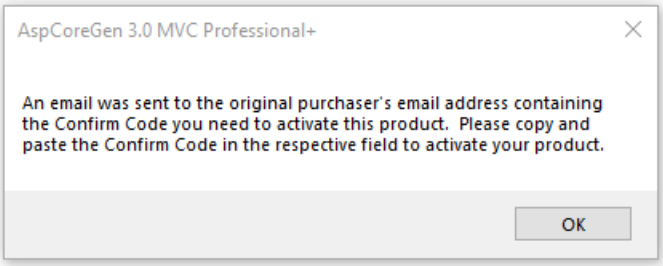

Figure 4 Confirm Code Sent

Enter the *Confirm Code (4)* you received from your email in the respective box and then click the *Activate* button. See Figure 5.

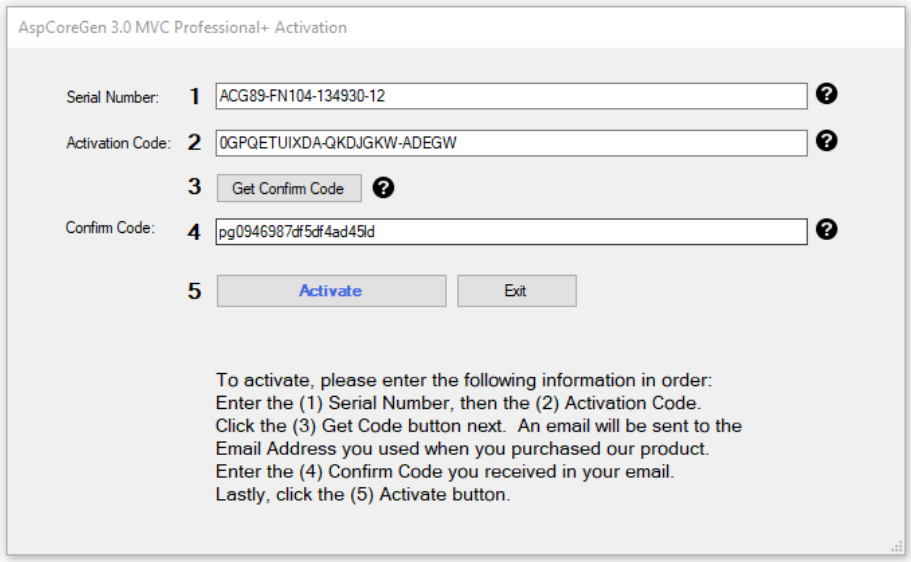

Figure 5 Enter Confirm Code

If you enter an invalid *Serial Number*, *Activation Code,* or *Confirm Code* a warning pops up. When this happens, click the *Ok* button (See Figure 6), then re-enter the *Serial Number*, *Activation Code,* and *Confirm Code* Again.

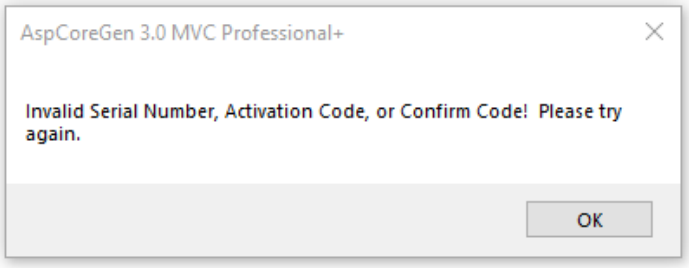

Figure 6 Invalid Serial Number, Activation Code, or Confirm Code

If you enter a valid *Serial Number*, *Activation Code* and *Confirm Code*, you will be presented with the main window of the application, see Figure 7. You will only see the Activation Form once. After that, you will always go straight to the main window.

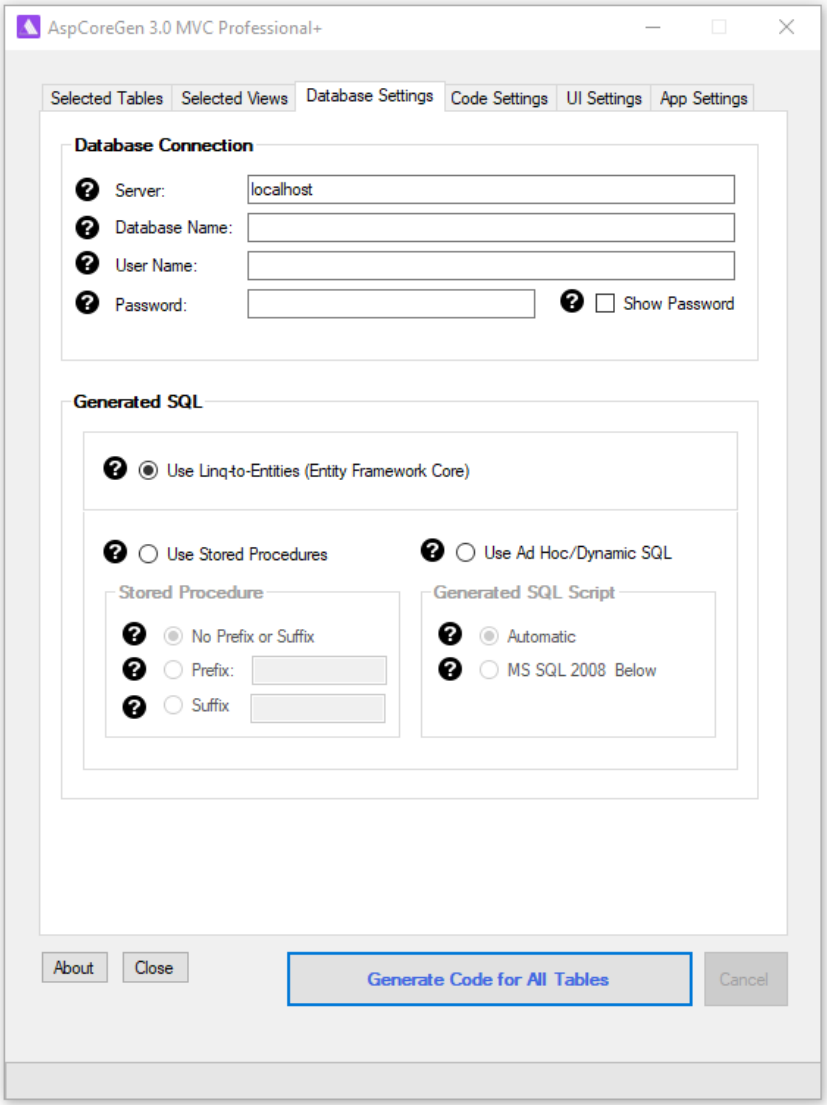

Figure 7 Main Window

A Simple Interface

To keep AspCoreGen 3.0 MVC simple, there's only one main interface as shown in Figure 7. The main window consists of six (6) tabs.

1. **Selected Tables**: AspCoreGen 3.0 MVC generates code from all the tables in your database by default. You can choose to generate from selected tables only from the Code Settings tab, and then select just the tables you need to generate from on this tab.

2. **Selected Views**: You can choose to generate from selected views only from the Code Settings tab, and then select just the views you need to generate from on this tab.

3. **Database Settings:** This is where you enter the database you want to generate code from and whether you want to generate linq-to-entities (entity framework core), stored procedures, or dynamic SQL. This is probably going to be your most used tab.

4. **Code Settings**: You'll find a selection here on where to generate your objects from: all tables, all views, selected tables, or selected views. The language the generated code will be in is C#.

   Three projects can be generated and are shown in this tab in 3 separate boxes:
   a. Web Application (ASP.NET Core 3.1 Project).
   b. Business Layer and Data Layer API (Core 3.1 Class Library Project).
   c. Web API (ASP.NET Core 3.1 Web API Project). This is optional.

   You can enter the Name of each project and the directory for the Web Application. Directories for the Business Layer and Data Layer API and Web API projects are filled automatically. You can choose to Generate Code Examples, and also a Web API project.

5. **UI Settings**: You can customize your own settings for the generated ASP.NET MVC Views here. You can choose themes for the JQuery UI controls. You can also select which MVC Views to generate and the MVC View's filename prefix to use for each MVC View.

6. **App Settings**: These are application settings. You can see a list of files that are overwritten every time you generate code for the same web project. You can also see a list of files that are only written once every time you generate code for the same web project. These (overwritten and written-once) files are listed per project (web project, class library, web api project).

   You can also reset all settings to its original default from here.

That seems like a lot of features, you're probably asking **"*where's the One Click feature?*"**

Since AspCoreGen 3.0 MVC remembers the last settings you used such as, e.g. server, database name, directory, namespace, etc., the next time you open AspCoreGen 3.0 MVC, and you're developing for the same web project, you can just click the Generate… button, that simple.

A Quick Tour

Let's learn how to generate an ASP.NET Core Web Application, a Class Library (our middle tier and data tier), the optional ASP.NET Core Web API project using AspCoreGen 3.0 MVC. **We're going to use *Microsoft's Northwind database* for this demo. Please Google and download it.** Or you can use your own database; just follow the steps shown below.

1. Open the AspCoreGen 3.0 MVC app.

2. Click the "*Generate Code for All Tables*" button at the bottom of the app. Notice an error message box pops up showing the required fields to fill. See Figure 8.

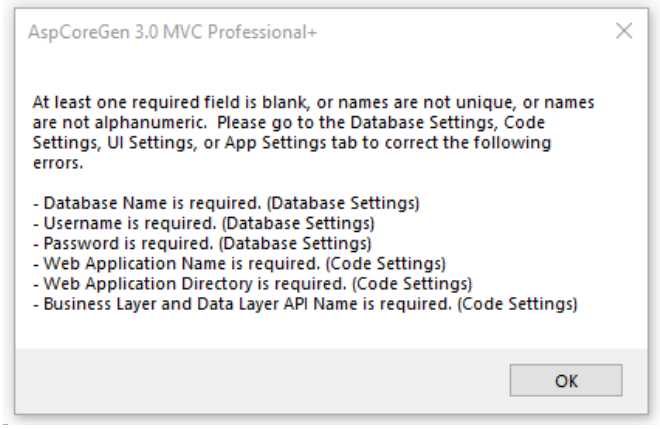

Figure 8 Error Message

2. Click the *OK* button to close the pop up message. Go to the *Database Settings* tab and start filling the required fields. Do the same in the *Code Settings* tab. See Figures 9 and 10.

   a. **Database Connection:** This is the MS SQL Server Database connection settings to the database. All of these items are required and cannot be blank.
      - **Server:** Server Name. If the MS SQL Server Database is local, you can just enter "*localhost*".
      - **Database Name:** The database you're trying to access.
      - **User Name:** An admin user name login that has access to the database.
      - **Password:** The password associated with the user name.

      Note: Checking the *Show Password* in Figure 9 will remember the password you enter here so you don't have to enter it again the next time you open AspCoreGen 3.0 MVC.

   b. **Generated SQL:** SQL script that will be generated.
      - **Use Linq-to-entities (Entity Framework Core):** Entity Framework.
      - **Use Stored Procedures:** Scripts generated straight in your MS SQL database.
        - **No Prefix or Suffix:** Names the generated script with no prefix or suffix. E.g. *Products_SelectAll*.
        - **Prefix:** Names the generated script with a prefix. For example, when you enter *"sp_"* on the Prefix text box, the generated stored procs will start with an *"sp_"*. E.g. ***sp_****Products_SelectAll*.
      - **Use Ad-Hoc/Dynamic SQL:** Scripts generated in your C# code.
        - **Automatic:** Automatically determines your MS SQL Server version before generating scripts.
        - **MS SQL 2008 Below:** Generates scripts for MS SQL Server versions 2008 and below. There's a slight difference with the generated script for older versions because some keywords are not available during this time.

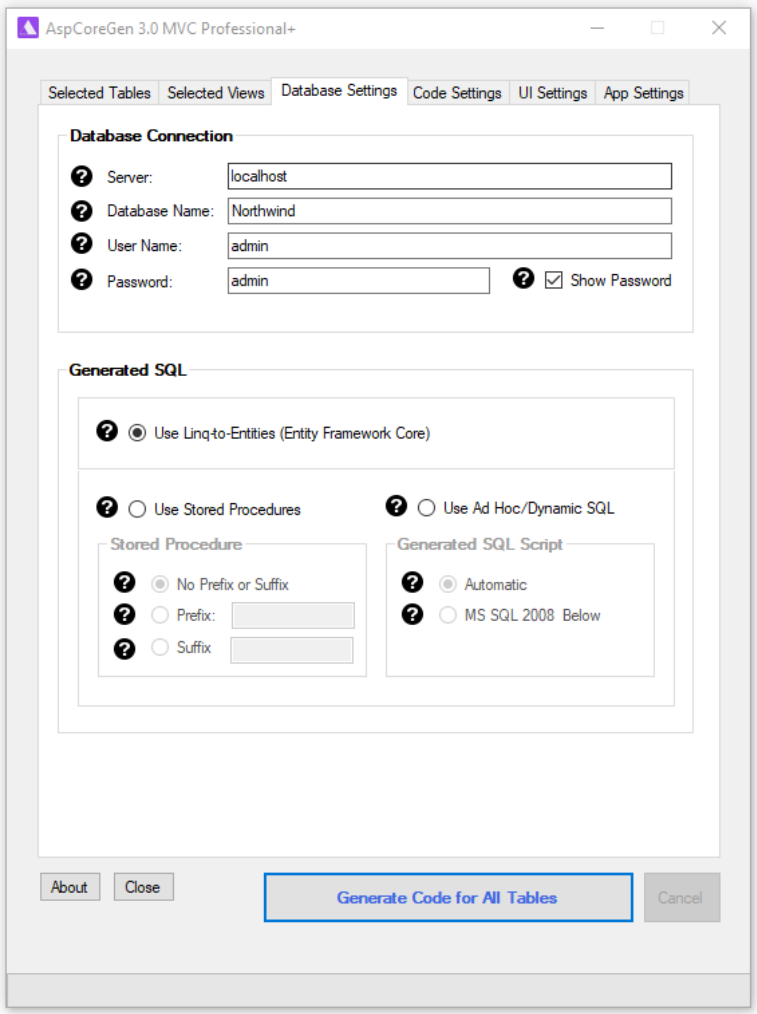

Figure 9 Database Settings Tab

a. **Database Object to Generate From:** Generates code from the Database Name you entered on the *Database Settings* tab.
   - **All Tables:** Generates code for all the database tables.
   - **All Views:** Generates code for all the database views. Not available for EF Core.
   - **Selected Tables Only:** Generates code for selected database tables. Clicking this will open the *Selected Tables* tab. You can load all the tables from your database here and select (check) the tables you want to generate code from.
   - **Selected Views Only:** Generates code for selected database views. Clicking this will open the *Selected Views* tab. You can load all the views from your database here and select (check) the views you want to generate code from. Not available for EF Core.

b. **Web Application:** There are 3 projects that can be generated by AspCoreGen 3.0 MVC. The web project is the user interface (front end). It houses the MVC views, javascript, css files, etc.
   - **Name:** Web application name.
   - **Directory:** Directory where the web app will be generated.
   - **Generate Code Examples:** Generates code examples placed in the web app.
   - **Dev Server Port:** The port number used when you run your web app while developing locally using Visual Studio. The web app will open up in a browser and at the address bar you'll see something like this: http://localhost:**27229**/index. 27229 here is the dev server port.

c. **Business Layer and Data Layer API:** A class library project. This houses middle-tier (business object) and data-tier (data layer) classes. These classes can be reusable in your other application. Middle-tier classes are referenced by the web app in CRUD operations by default unless the web API project is generated.
   - Name: Class library application name.
   - **Directory:** Directory where the class library app will be generated.

d. **Web API:** A web API project. This project is optional.
   - Use Web API: When checked, a web API project is generated and referenced by the front end (web application) in CRUD operations instead of the middle-tier.
   - Name: Web API application name.
   - **Directory:** Directory where the web API app will be generated.

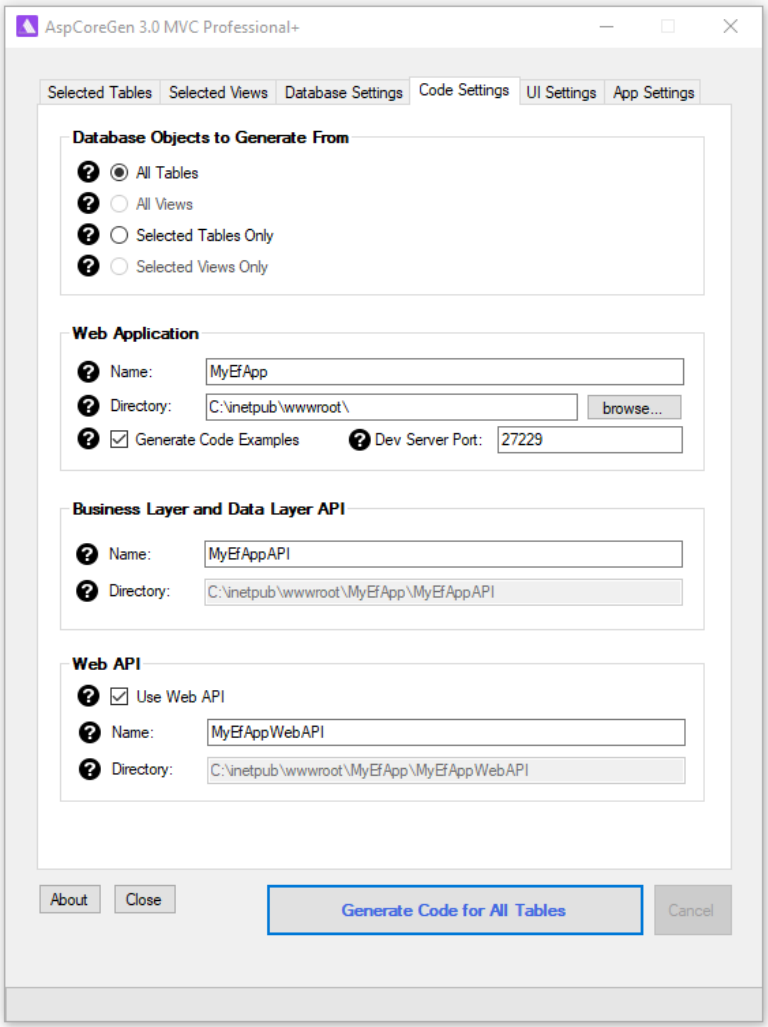

Figure 10 Code Settings Tab

3. Simply hover over any of the *Question Mark* images if you need information from the respective fields.

4. Click the "*Generate Code for All Tables*" button. AspCoreGen 3.0 MVC will start generating code. See Figure 11.

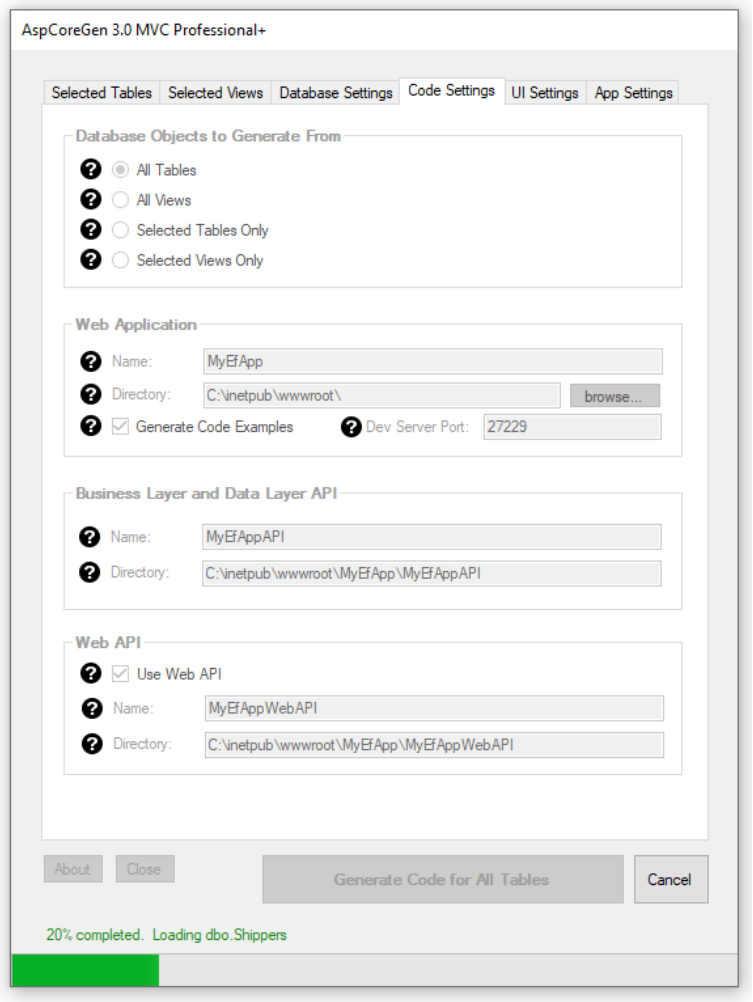

Figure 11 Generating Code

5. Wait for a few seconds. When AspCoreGen 3.0 MVC is done generating objects, a message pops up. See Figure 12.

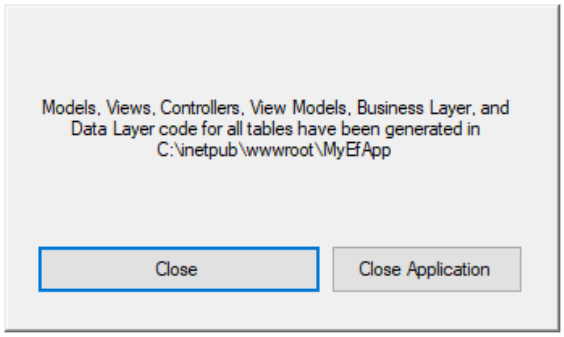

Figure 12 Done Generating Objects

6. Click the *Close Application* button to close the message box and exit the application.

7. To view the generated Web Application simply go to the directory you specified from the *Code Settings* tab of AspCoreGen 3.0 MVC. You will see three folders and a solution file like the one shown in Figure 13.

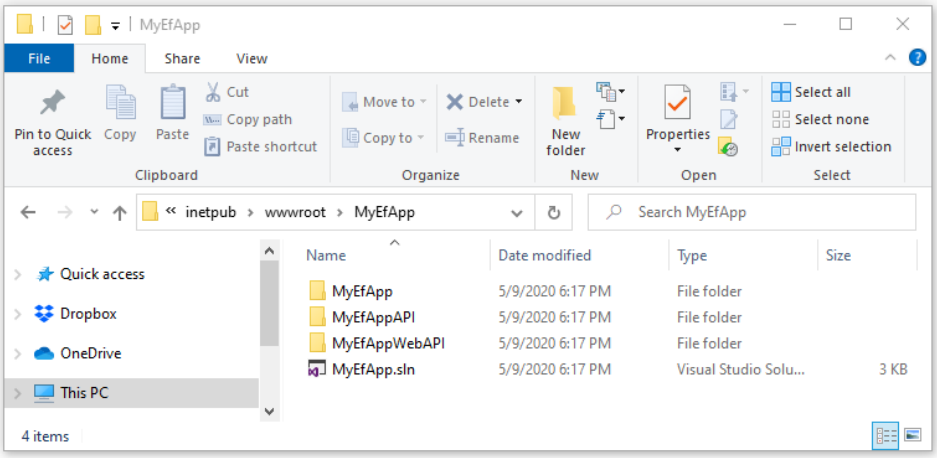

Figure 13 Generated Web Application

8. *Double-Click* the solution file (*MyEfApp.sln*). An *Elevated Permissions* dialog may appear. Click *Restart this application under different credentials*. See Figure 14.

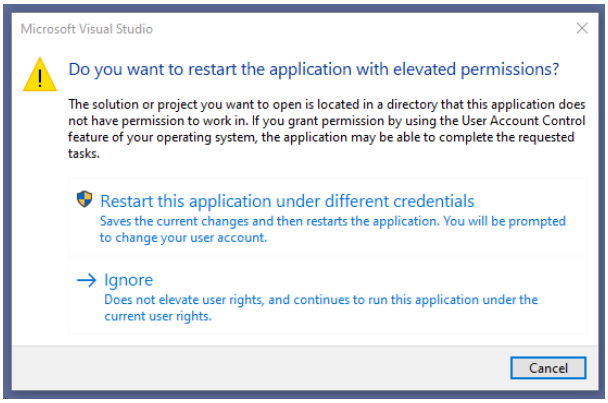

Figure 14 Elevated Permissions Dialog

9. *Double-Click* the solution file (*MyEfApp.sln*) in Visual Studio's recent apps list as seen in Figure 15.

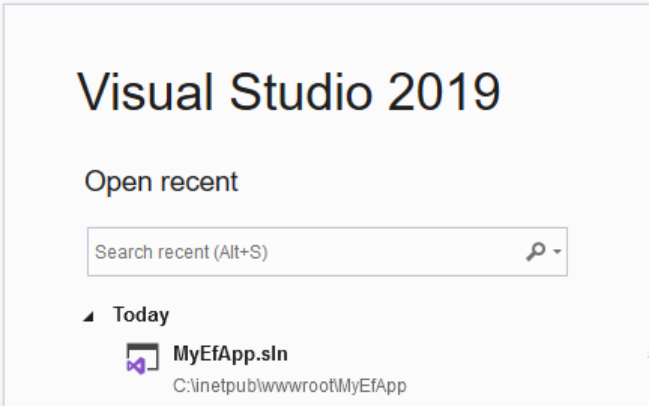

Figure 15 Visual Studio 2019 – Recent Apps

10. The generated *Web Application* will now be opened/loaded into Visual Studio 2019. See Figure 16.

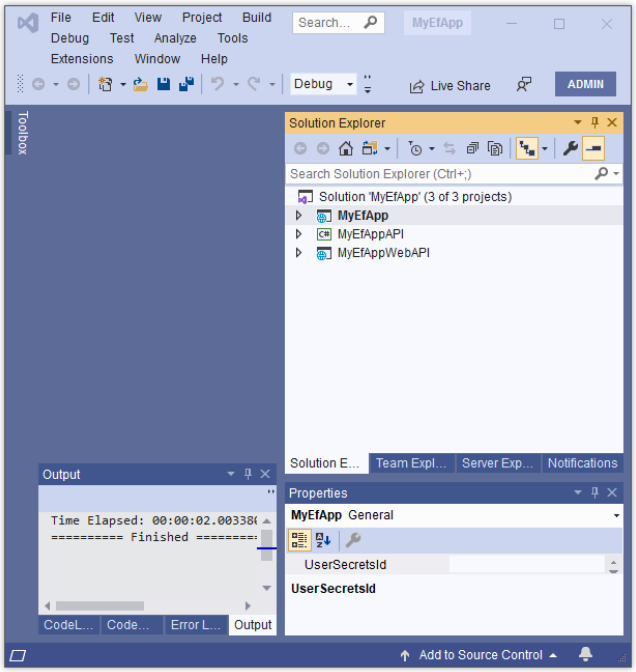

Figure 16 Generated Web Application in Visual Studio

11. Run the solution by pressing *F5*. You will see a list of all the generated ASP.NET Core Views in the *Home Page*. See Figure17. You can click any link to preview the functionality of each of the generated MVC View. However, we're not going to discuss this in this tutorial.

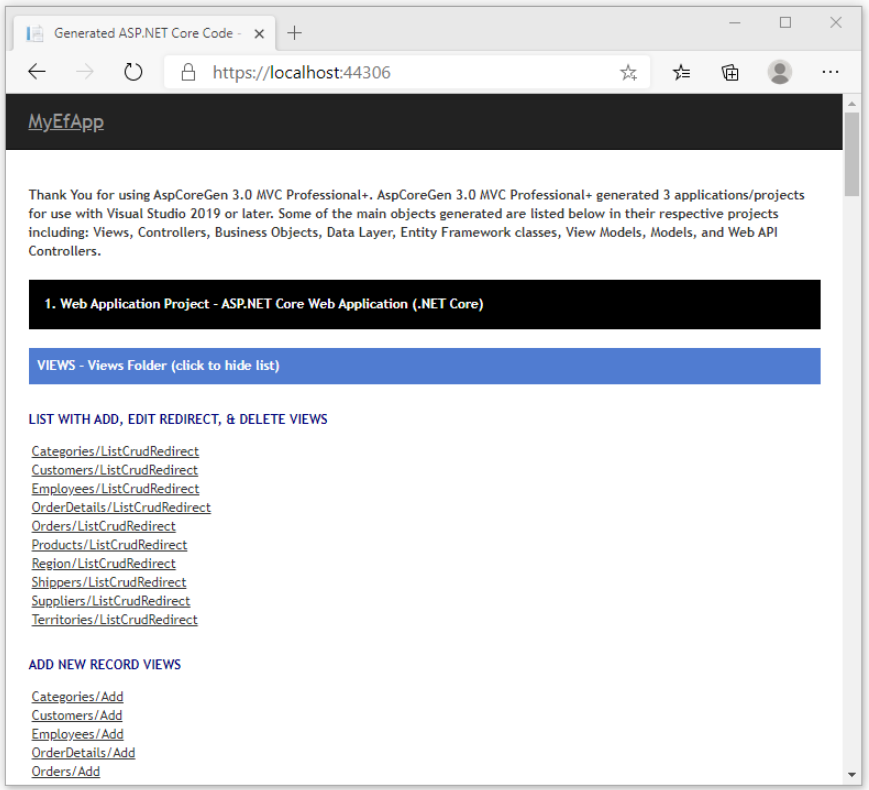

Figure 17 List of Generated MVC Objects

12. When you click any of the links you are redirected to that specific page. Play around to see the functionality of each web page. See Figure 18, this is an example of one of the pages.

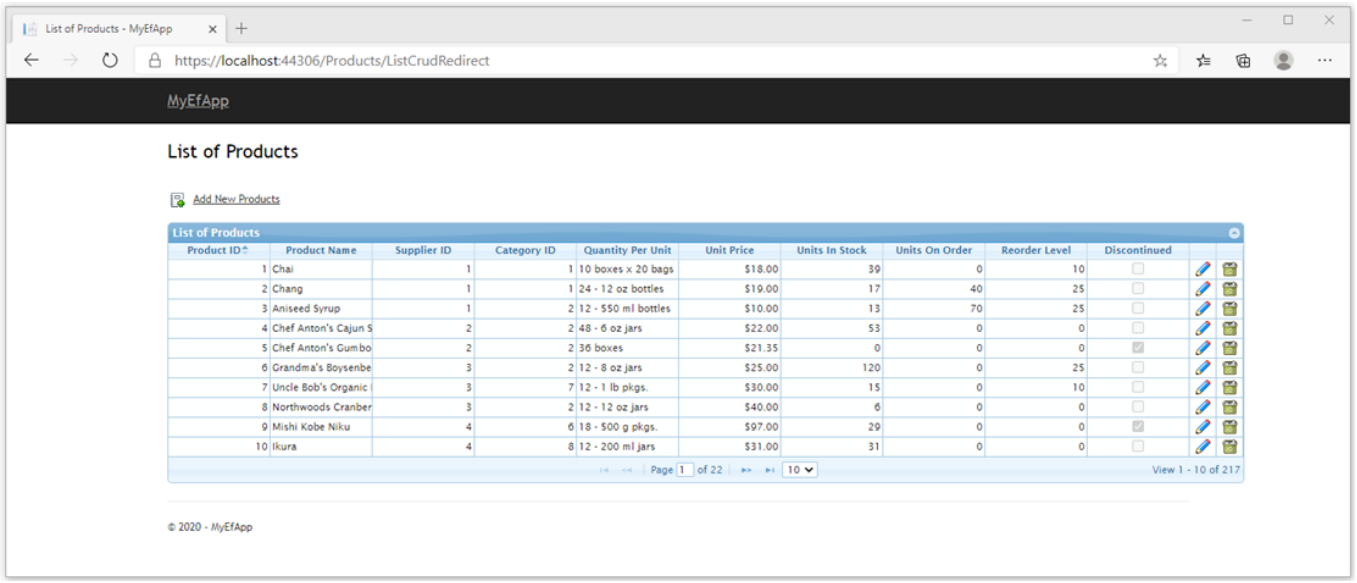

Figure 18 Example Page – Products/ListCrudRedirect

13. From the *Home Page*, Each black bar is the specific Project generated by AspCoreGen 3.0 MVC. Each black bar contains blue bars. Each blue bar on this page is clickable. Each blue bar is a section of the generated project. When clicked, each blue bar toggles which would either show or hide the list beneath them. See Figure 19.

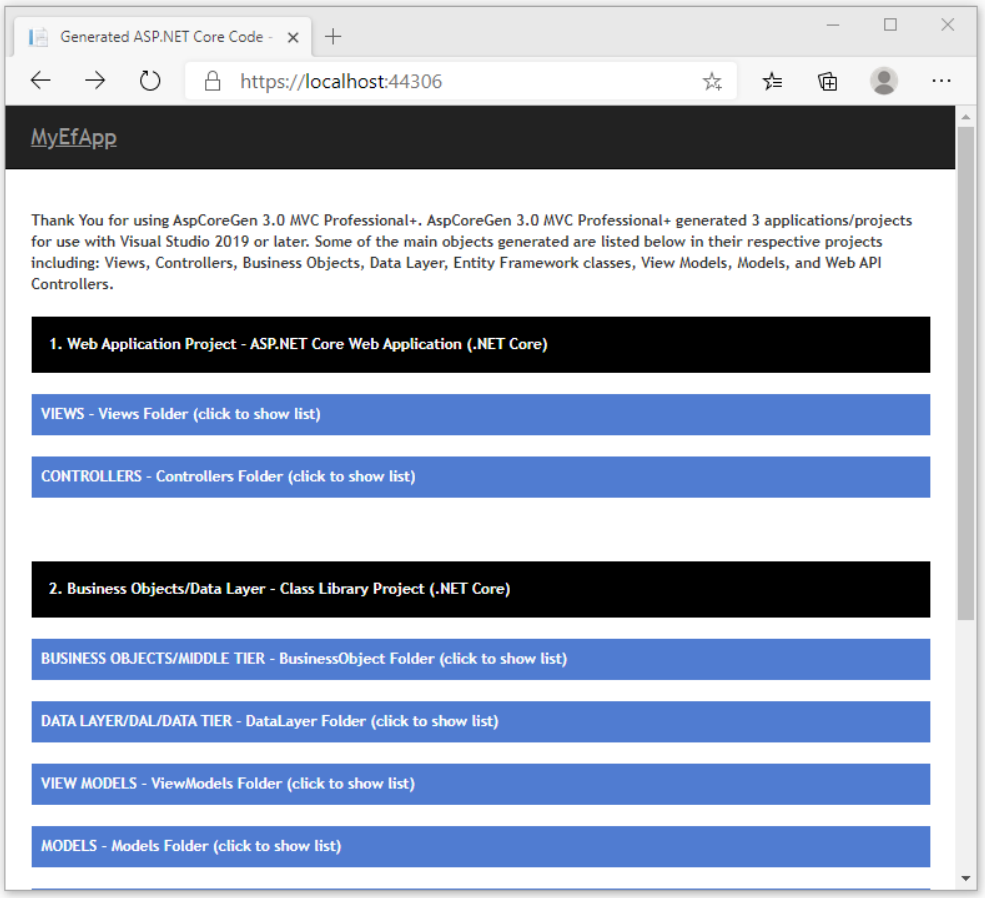

Figure 19 Three Generated Projects

14. Close the web page and go back to Visual Studio 2019. Expand each one of the projects in the *Solution Explorer*. The generated code's layout separated in 3 projects can be seen in Figure 20.

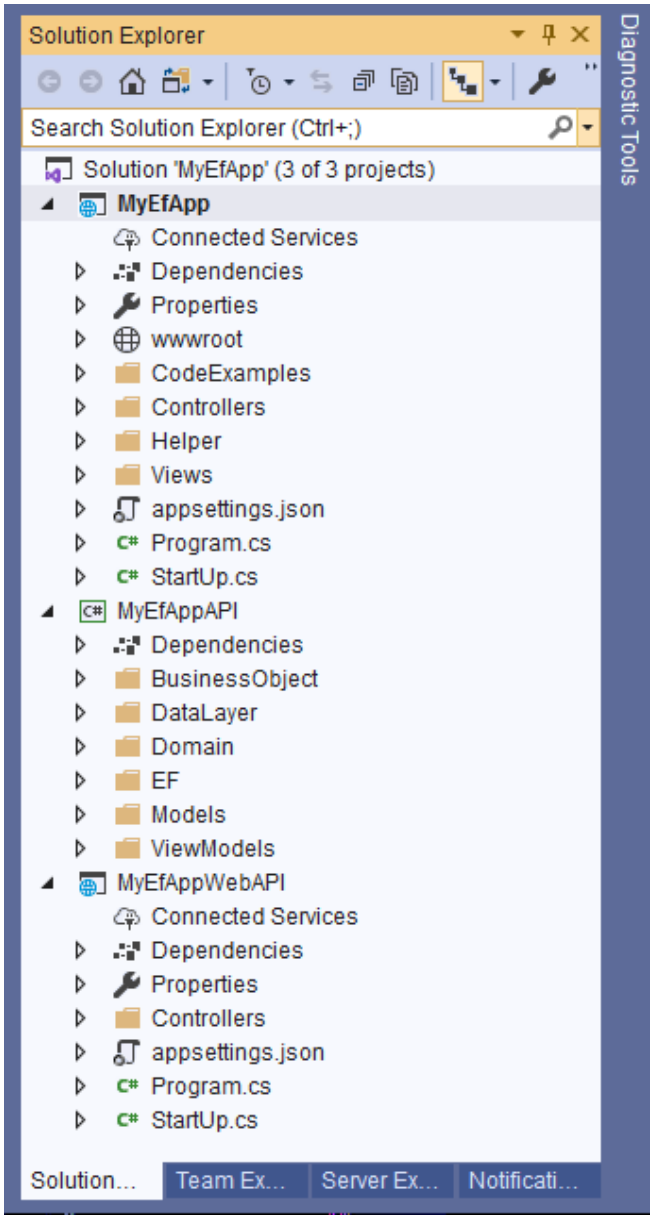

Figure 20 Three Projects Generated By AspCoreGen 3.0 MVC in Visual Studio 2019

This tutorial offers a quick look in using AspCoreGen 3.0 MVC. You can read end-to-end tutorials on more subjects on using AspCoreGen 3.0 MVC Professional Plus that came with your purchase. These tutorials are available to customers and are included in a link on your invoice when you purchase AspCoreGen 3.0 MVC Professional.

Note: Some features shown here are not available in the Express Edition.

End of tutorial.