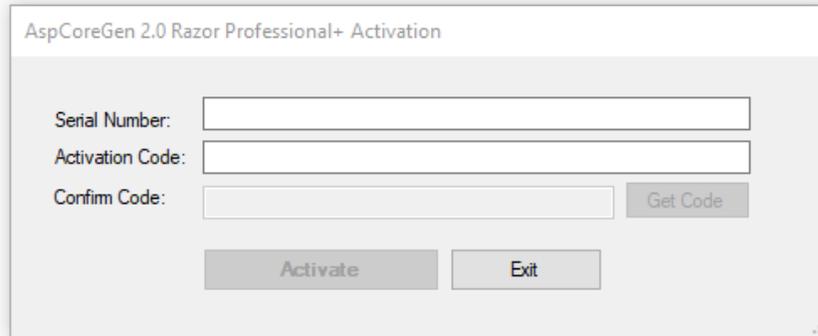


Activating AspCoreGen 2.0 Razor Pro Plus

Note: You need internet connection to activate AspCoreGen 2.0 Razor. You can install one (1) license of AspCoreGen 2.0 Razor 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.

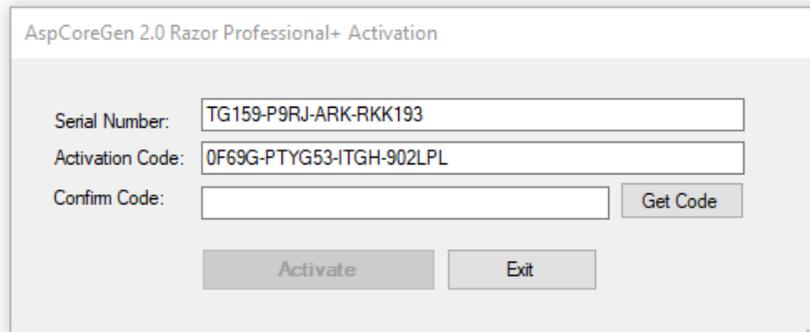
The first time you open AspCoreGen 2.0 Razor Professional Plus edition you will be presented with an activation form right after the Splash screen as shown in Figure 1. You will not be shown the activation form when using the Express edition.



The image shows a window titled "AspCoreGen 2.0 Razor Professional+ Activation". It contains three text input fields: "Serial Number:", "Activation Code:", and "Confirm Code:". To the right of the "Confirm Code:" field is a button labeled "Get Code". Below the input fields are two buttons: "Activate" and "Exit".

Figure 1 Activation Form

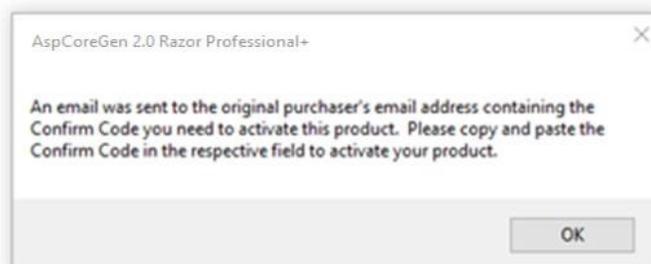
Enter the *Serial Number* and *Activation Code* we provided you in the respective text boxes. The *Serial Number* and *Activation Code* contains dashes "-", make sure to include these dashes when entering them. Then click the *Get Code* button to get the *Confirm Code*. See Figure 2.



The image shows the same window as Figure 1, but with data entered into the "Serial Number:" and "Activation Code:" fields. The "Serial Number:" field contains "TG159-P9RJ-ARK-RKK193" and the "Activation Code:" field contains "0F69G-PTYG53-ITGH-902LPL". The "Confirm Code:" field is empty. The "Get Code" button is still present.

Figure 2

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 3. Click the OK button.



The image shows a small pop-up window titled "AspCoreGen 2.0 Razor Professional+" with a close button (X) in the top right corner. The text inside reads: "An email was sent to the original purchaser's email address containing the Confirm Code you need to activate this product. Please copy and paste the Confirm Code in the respective field to activate your product." At the bottom right, there is an "OK" button.

Figure 3 Confirm Code Sent

Enter the *Confirm Code* in the respective box and then click the *Activate* button.

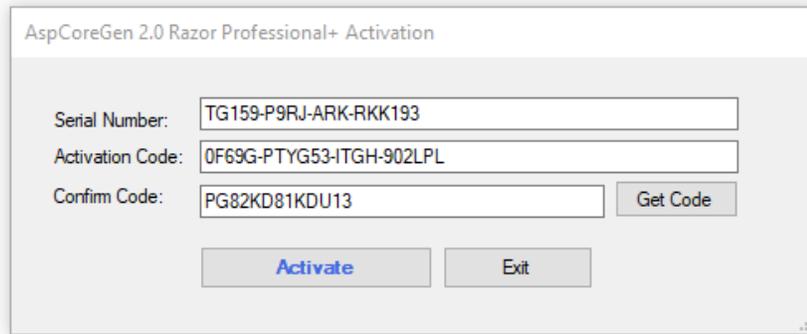


Figure 4 Enter Confirm Code

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 5), then re-enter the *Serial Number*, *Activation Code*, and *Confirm Code* Again.

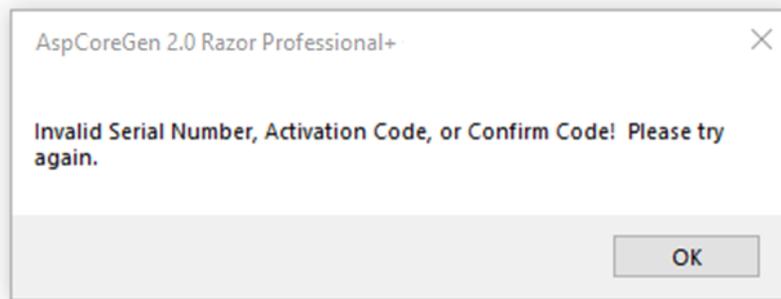


Figure 5 Invalid Serial Number, Activation Code, or Confirm Code

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 6. You will only see the activation form once. After that, you will always go straight to the main window.

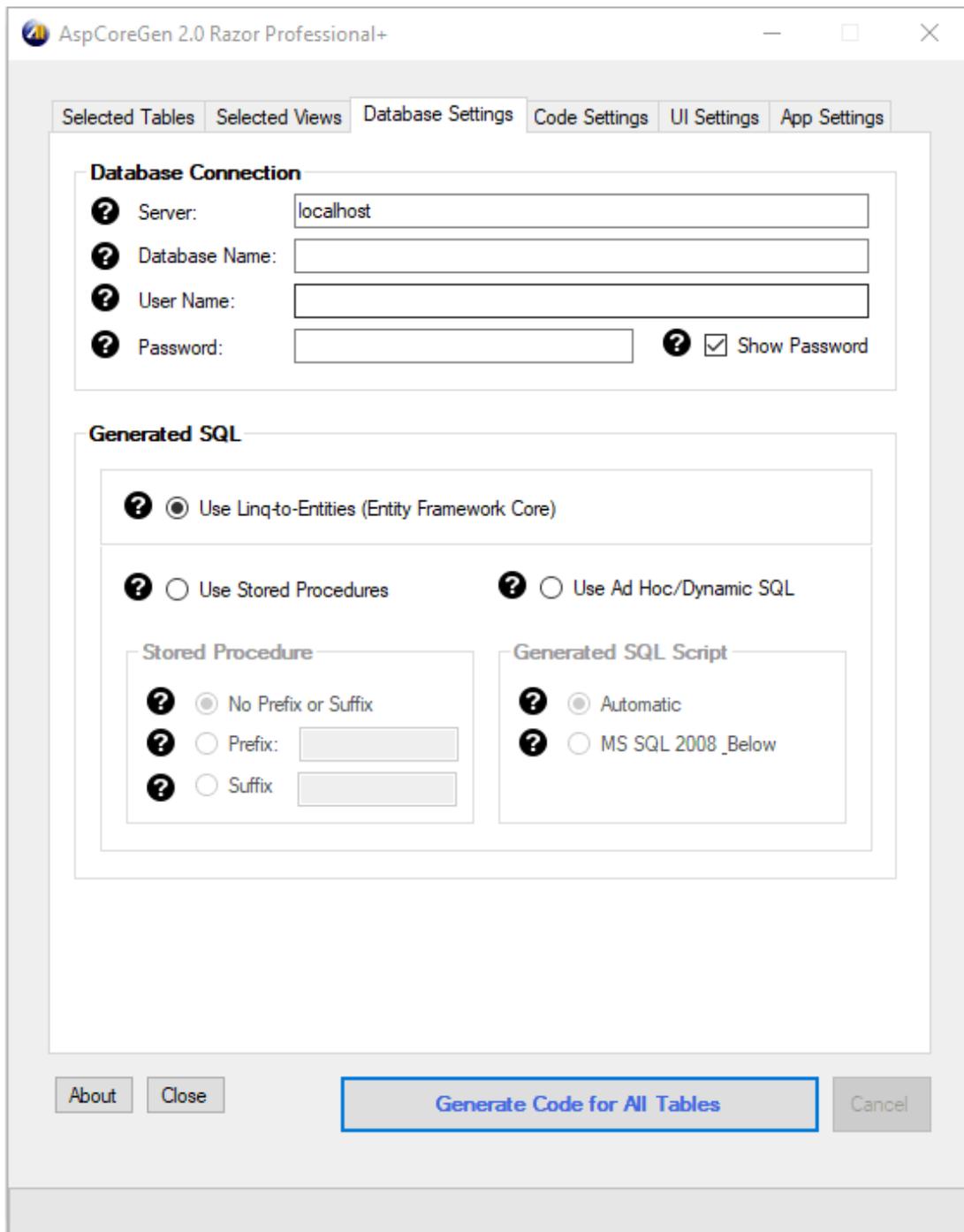


Figure 6 Main Window

A Simple Interface

To keep AspCoreGen 2.0 Razor simple, there's only one main interface as shown in Figure 6. The main window consists of six (6) tabs.

1. **Selected Tables:** AspCoreGen 2.0 Razor 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 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 database views 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. C# is the only language that will be generated because VB.NET is not supported in ASP.NET Core 2.1 during the time of this writing.

Three projects can be generated and are shown in this tab in 3 separate boxes:

- a. Web Application (ASP.NET Core 2.1 Project).
- b. Business Layer and Data Layer API (Core 2.1 Class Library Project).
- c. Web API (ASP.NET Core 2.1 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 Razor Pages here. You can choose themes for the JQuery UI controls. You can also select which Razor Pages to generate and the Razor Page's filename prefix to use for each Razor Pages.
6. **App Settings:** These are application settings. Almost all generated code/Razor Pages are overwritten every time you use AspCoreGen 2.0 Razor. However, you can choose not to overwrite some key files from here. 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 2.0 Razor remembers the last settings you used such as, e.g. server, database name, directory, namespace, language etc., the next time you open AspCoreGen 2.0 Razor, and you can just click the Generate... button, that simple.

A Quick Tour

Let's learn how to generate an ASP.NET Core Razor Web Application, a Class Library (our middle tier and data tier), the optional ASP.NET Core Web API project using AspCoreGen 2.0 Razor. **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. Click the "Generate Code for All Tables" button. Notice an error pops up showing the required fields to fill. See Figure 7.

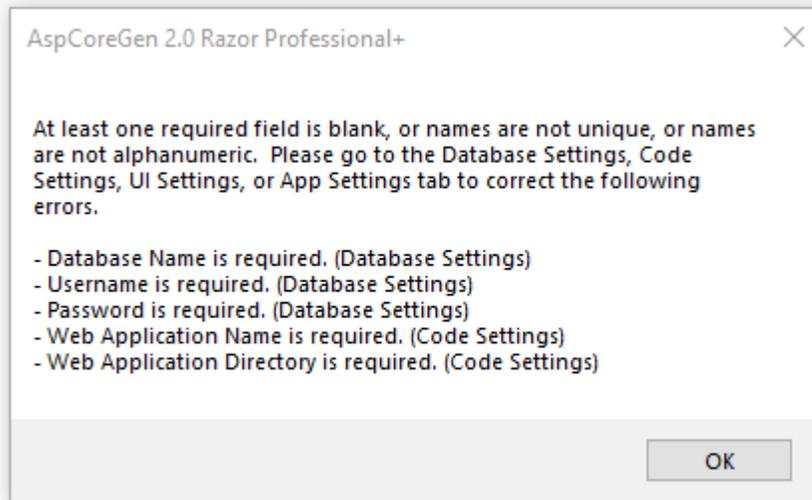


Figure 7 Error Message

2. Click the *OK* button to close the pop up. Go to the *Database Settings* tab and start filling the required fields. Do the same in the *Code Settings* tab. See Figures 8 and 9.

Note: Checking the *Show Password* in Figure 8 will remember this password so you don't have to enter it again the next time you open AspCoreGen 2.0 Razor.

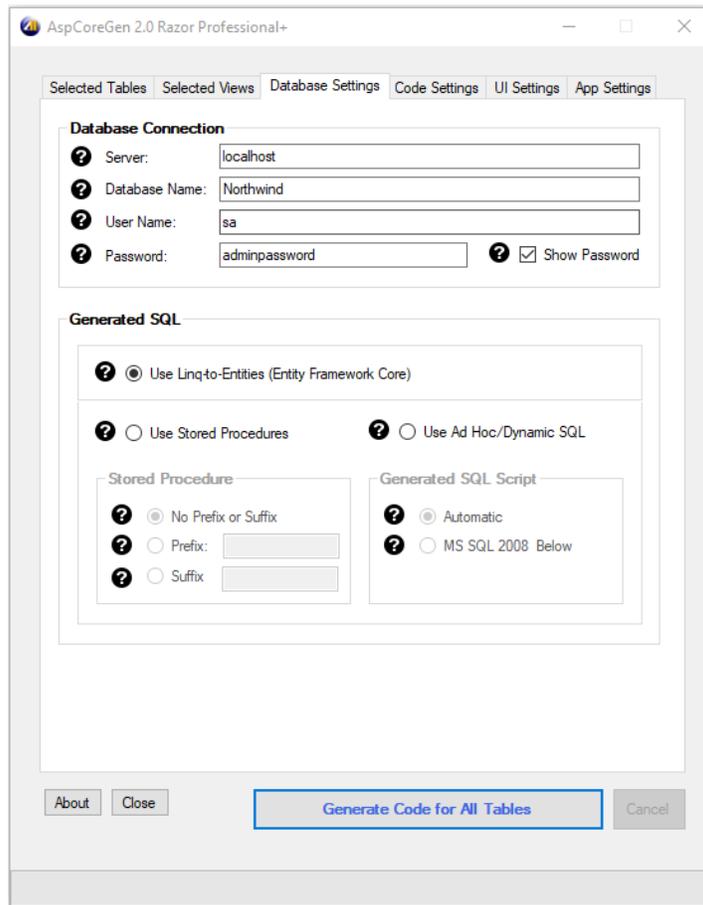


Figure 8 Database Settings Tab

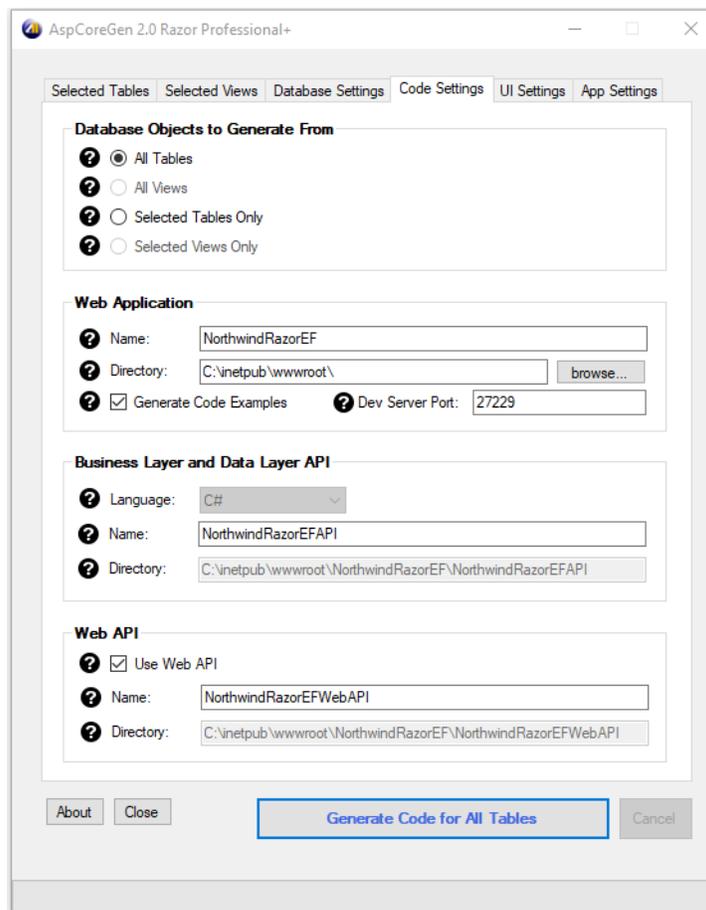


Figure 9 Code Settings Tab

- Simply hover over the *Question Mark* image  if you need information from the respective fields.
- Click the “*Generate Code for All Tables*” button. AspCoreGen 2.0 Razor will start generating objects. See Figure 10.

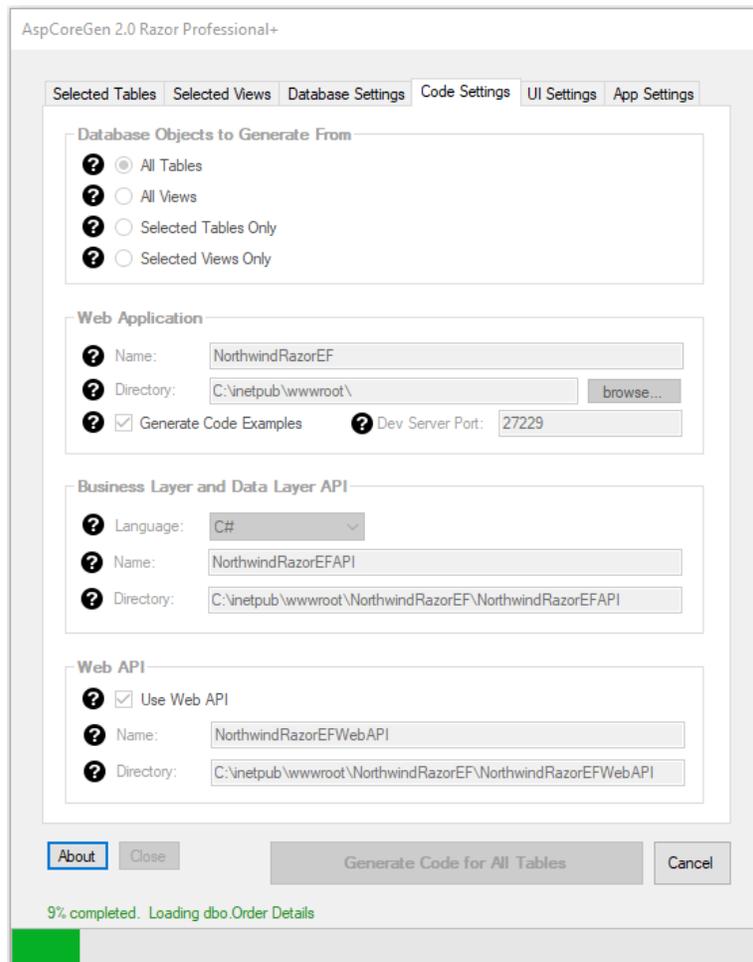


Figure 10 Generating Code

- Wait for a few seconds. When AspCoreGen 2.0 Razor is done generating objects, a message pops up. See Figure 11.

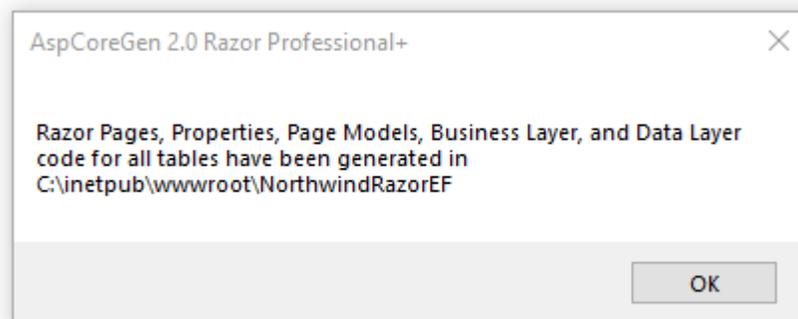


Figure 11 Done Generating Objects

- Click the *OK* button to close the message. Then close AspCoreGen 2.0 Razor by clicking the *Close* button.

- To view the generated web application, simply go to the directory you specified from the *Code Settings* tab of AspCoreGen 2.0 Razor. You will see three folders and a solution file like the one shown in Figure 12.

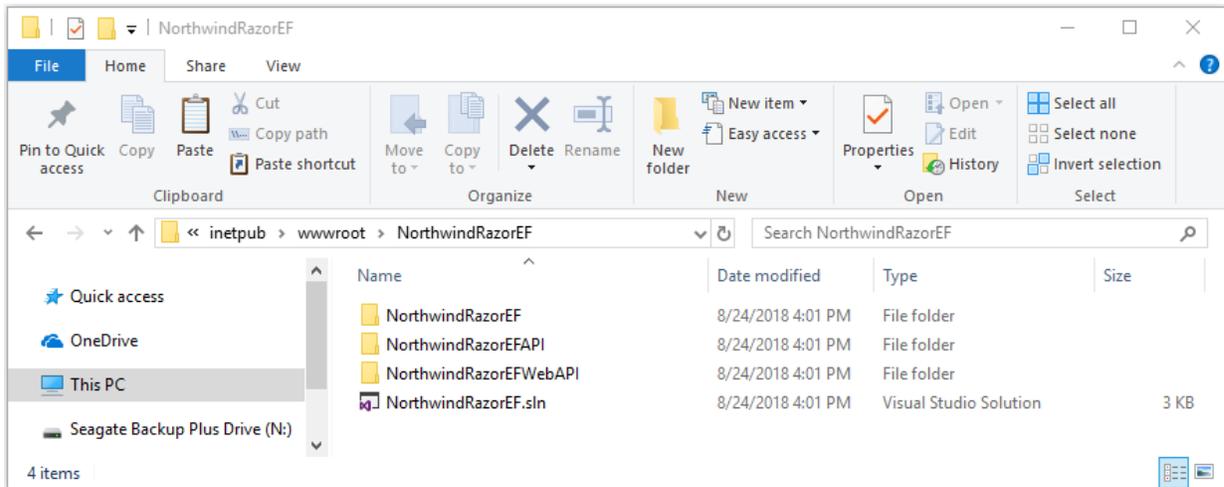


Figure 12 Generated Web Application

- To view the generated web application, simply go to the directory you specified from the *Code Settings* tab of AspCoreGen 2.0 Razor. You will see a list like the one shown in Figure 13. Each project is separated in the Folders specified in the *Code Settings* tab.

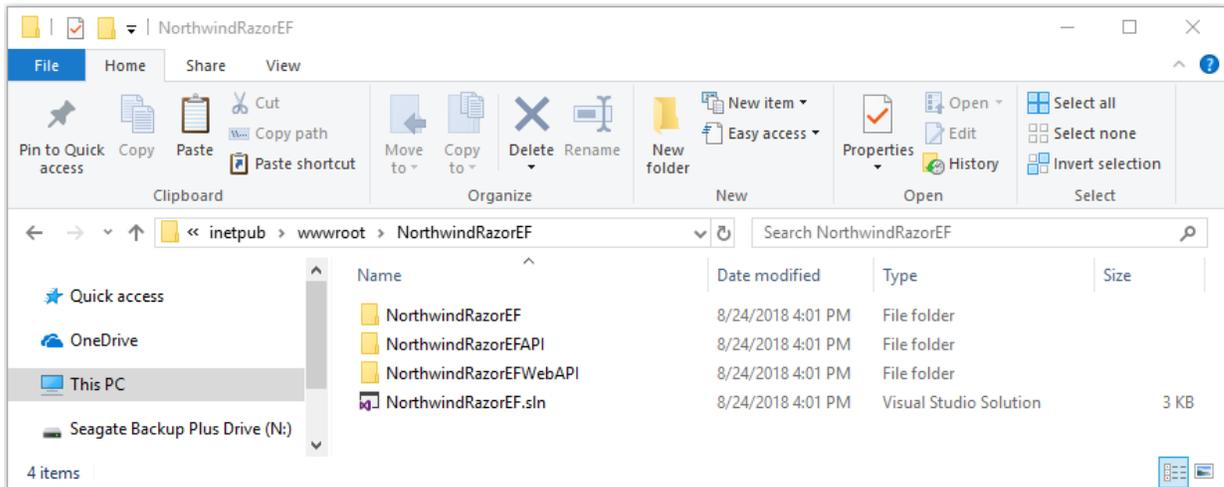


Figure 13 Generated Razor Web Application

- Double-Click* the solution file (for this example, it will be the *NorthwindRazorEF.sln*). The solution will open in Visual Studio 2017.

10. Run the solution by pressing *F5*. You will see a list of all the generated ASP.NET Core Razor Pages. See Figure 14. You can click any link to preview the functionality of each of the generated Razor Page. However, we're not going to discuss this in this tutorial.

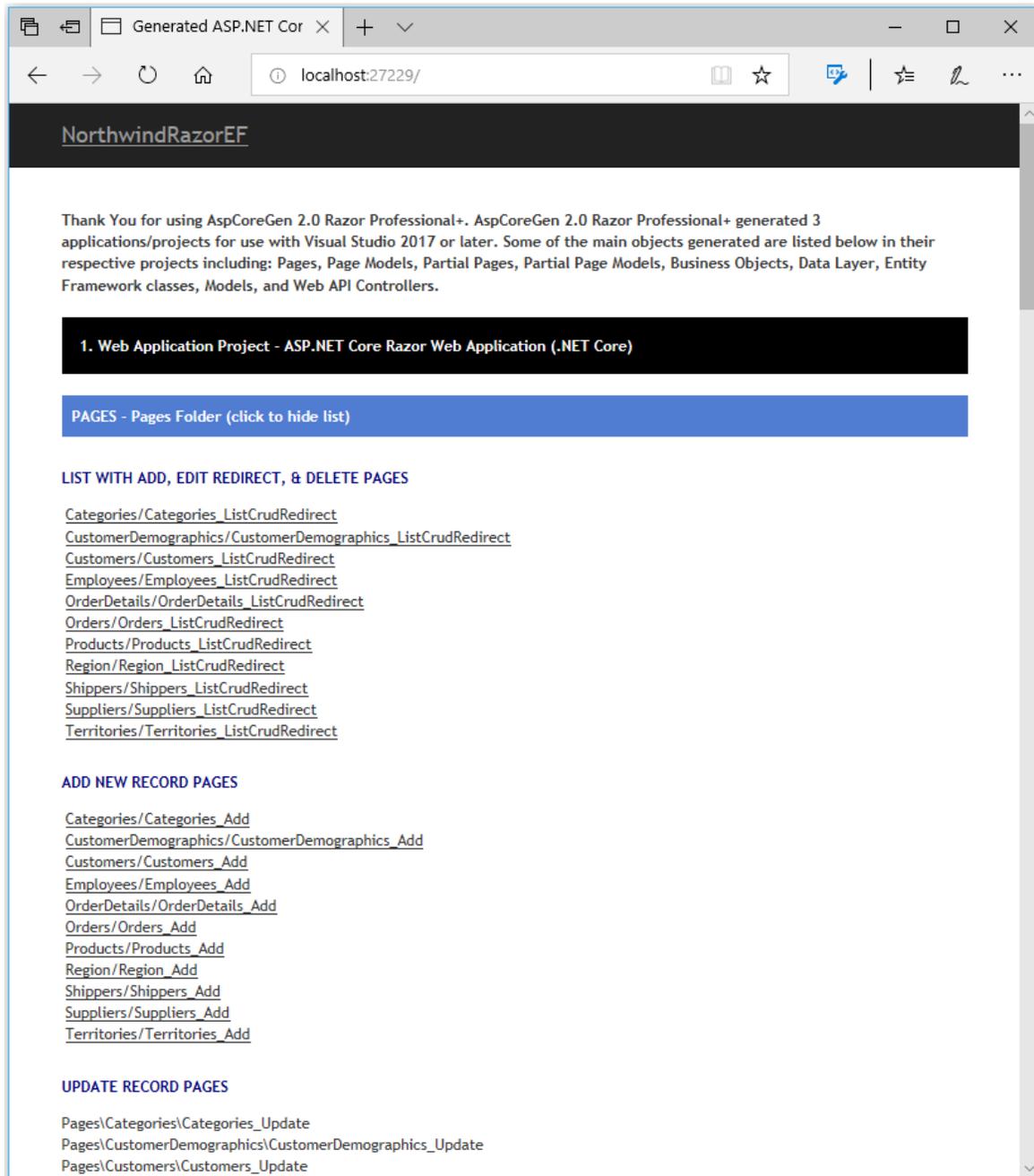


Figure 14 List of Generated Razor Objects

11. When you click any of the links you are redirected to that specific page. Play around to see the functionality of each web page.

12. Each black bar is the specific Project generated by AspCoreGen 2.0 Razor. 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 15.

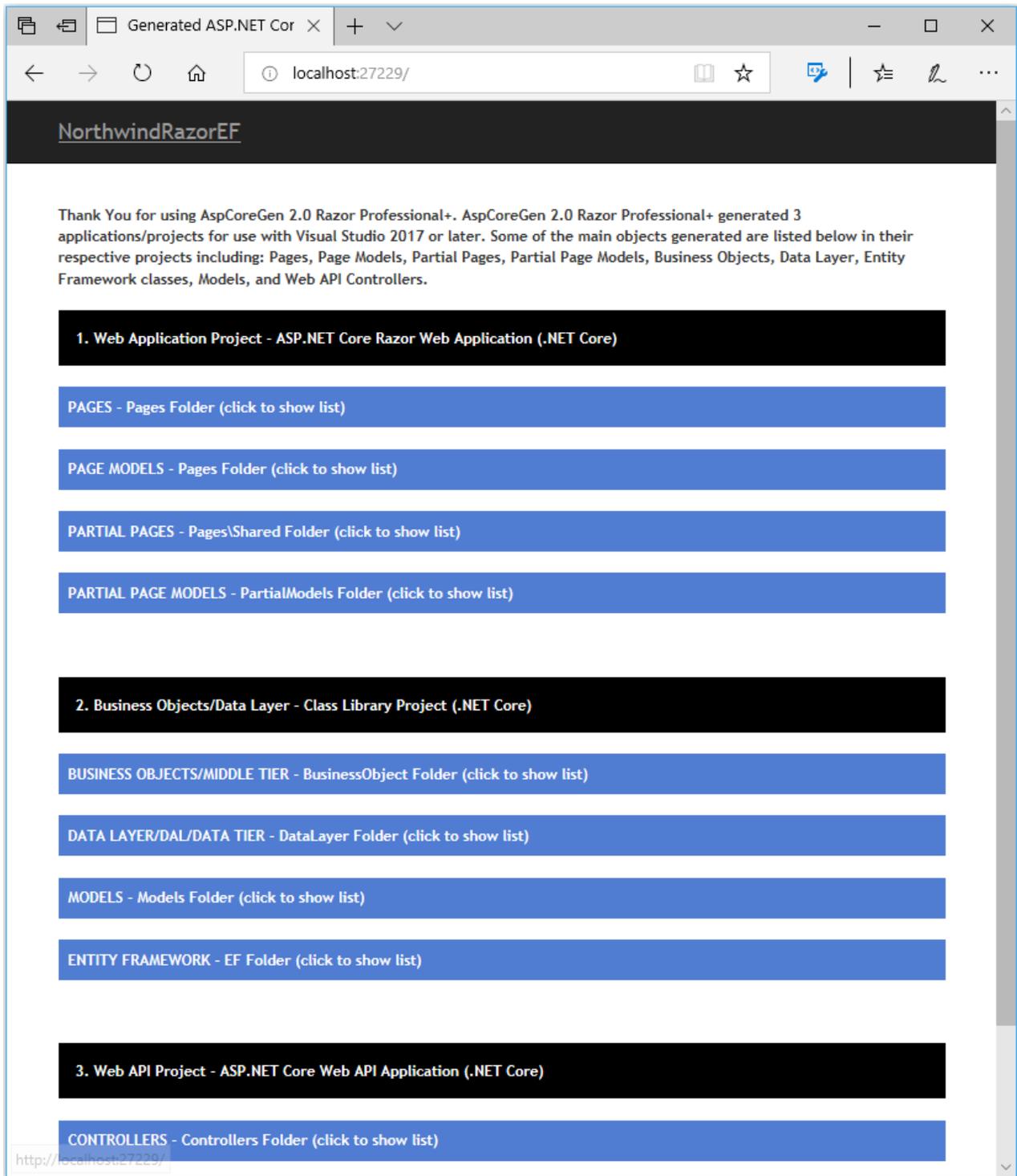


Figure 15 Three Generated Projects

13. Close the web page and go back to Visual Studio 2017. The generated code separated in 3 projects can be seen in Figure 16.

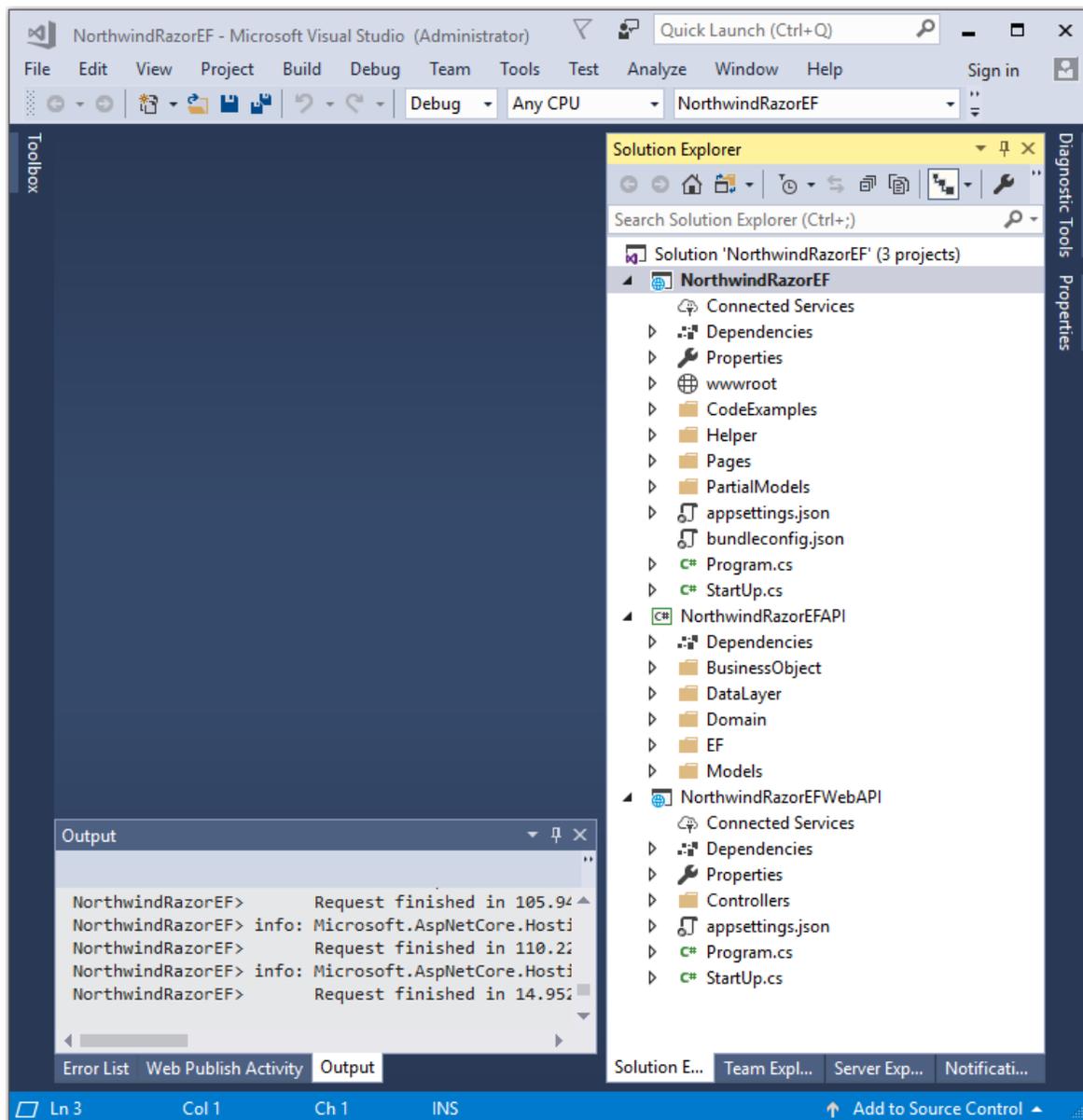


Figure 16 Three Projects Generated By AspCoreGen 2.0 Razor in Visual Studio 2017

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

End of tutorial.