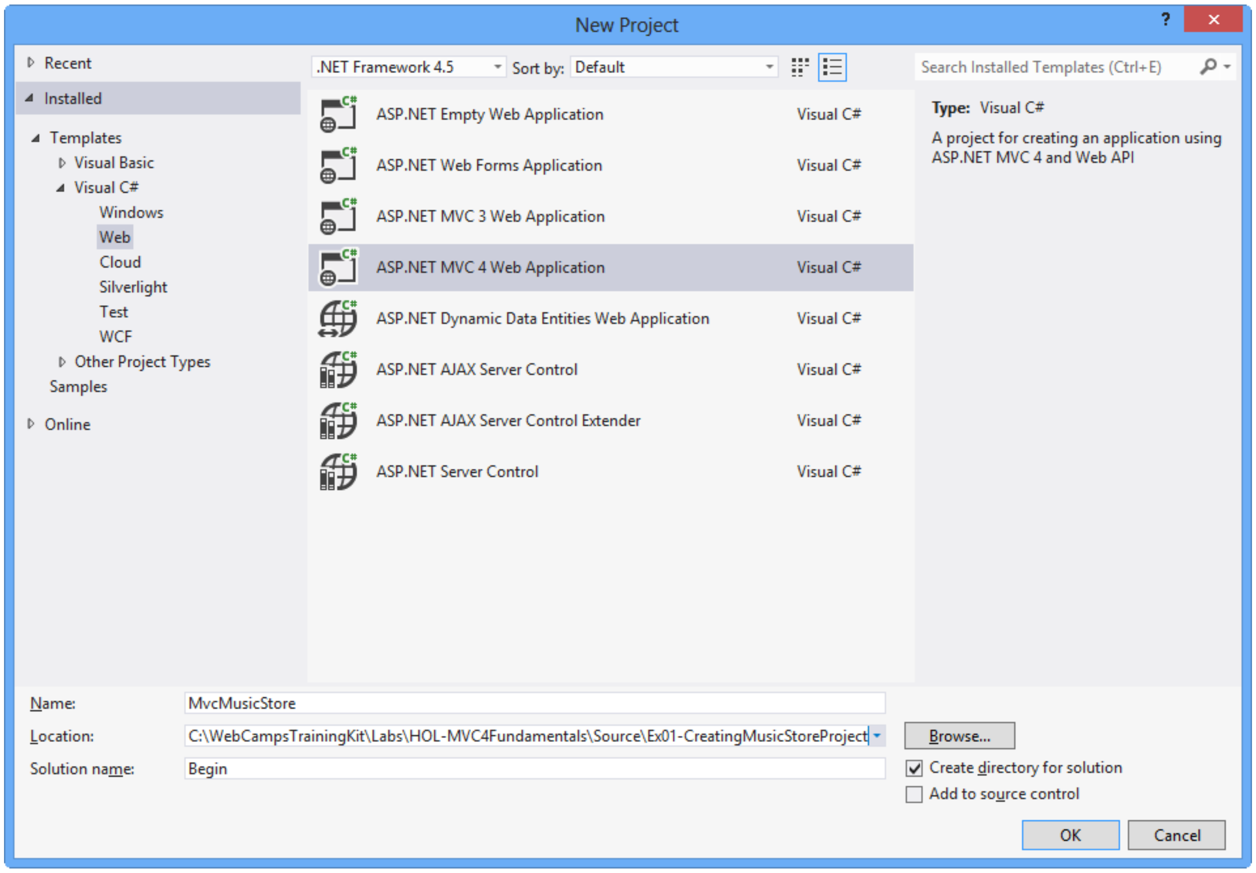
Progress Check Exercise 1

Creating MusicStore ASP.NET MVC Web Application Project

In this exercise, you will learn how to create an ASP.NET MVC application in Visual Studio 2012 Express for Web as well as its main folder organization. Additionally, you will learn how to add a new Controller and make it display a simple string in the application's home page.

Task 1 - Creating the ASP.NET MVC Web Application Project

1. In this task, you will create an empty ASP.NET MVC application project using the MVC Visual Studio template. Start **VS Express for Web**.
2. On the **File** menu, click **New Project**.
3. In the **New Project** dialog box select the **ASP.NET MVC 4 Web Application** project type, located under **Visual C#, Web** template list.
4. Change the **Name** to *MvcMusicStore*.
5. Set the location of the solution inside a new **Begin** folder in this Exercise's Source folder, for example **[YOUR-HOL-PATH]\Source\Ex01-CreatingMusicStoreProject\Begin**. Click **OK**.



ASP.NET Empty Web Application	Visual C#
ASP.NET Web Forms Application	Visual C#
ASP.NET MVC 3 Web Application	Visual C#
ASP.NET MVC 4 Web Application	Visual C#
ASP.NET Dynamic Data Entities Web Application	Visual C#
ASP.NET AJAX Server Control	Visual C#
ASP.NET AJAX Server Control Extender	Visual C#
ASP.NET Server Control	Visual C#

Type: Visual C#

A project for creating an application using ASP.NET MVC 4 and Web API

Name: MvcMusicStore

Location: C:\WebCampsTrainingKit\Labs\HOL-MVC4Fundamentals\Source\Ex01-CreatingMusicStoreProject

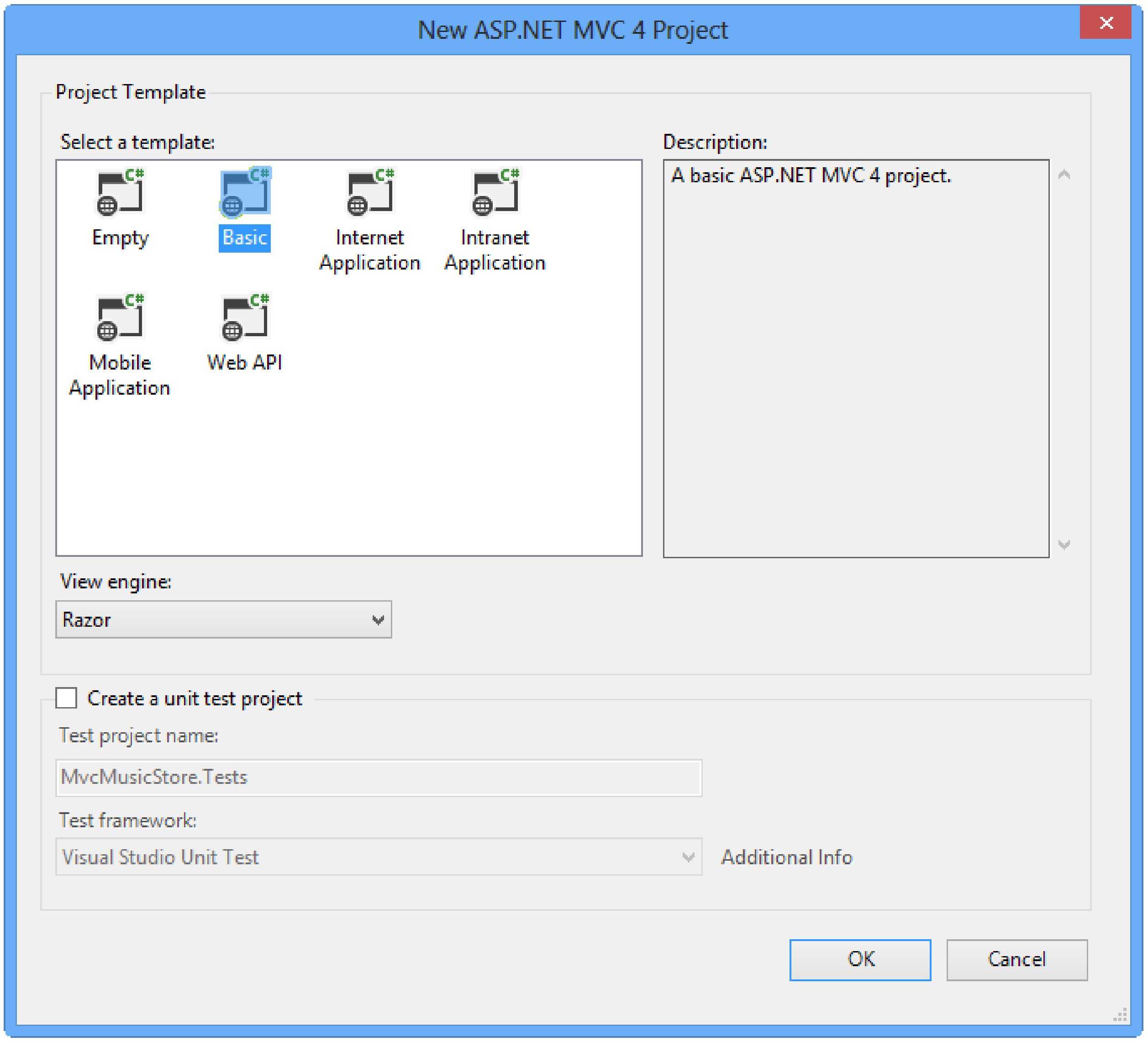
Solution name: Begin

[✓] Create directory for solution

[ ] Add to source control

Create New Project Dialog Box

6. In the **New ASP.NET MVC 4 Project** dialog box select the **Basic** template and make sure that the **View engine** selected is **Razor**. Click **OK**.


New ASP.NET MVC 4 Project

Project Template

Select a template: Basic

Description: A basic ASP.NET MVC 4 project.

View engine: Razor

Create a unit test project: [ ]

Test project name: MvcMusicStore.Tests

Test framework: Visual Studio Unit Test

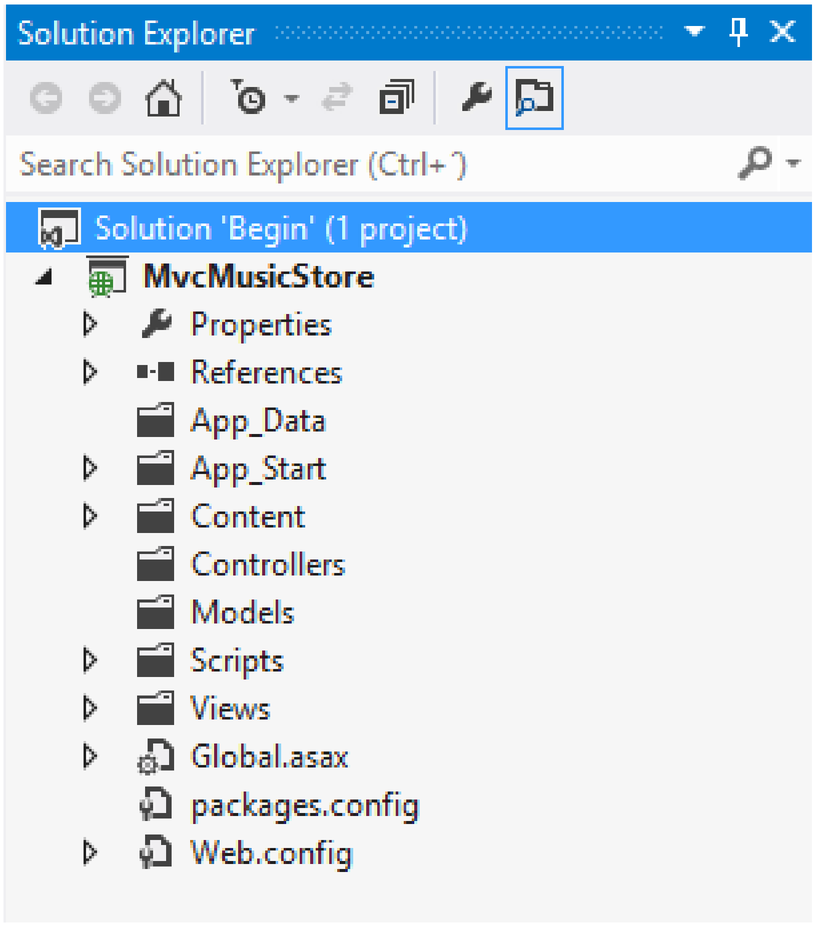
New ASP.NET MVC 4 Project Dialog Box

Task 2 - Exploring the Solution Structure

The ASP.NET MVC framework includes a Visual Studio project template that helps you create Web applications supporting the MVC pattern. This template creates a new ASP.NET MVC Web application with the required folders, item templates, and configuration-file entries.

In this task, you will examine the solution structure to understand the elements that are involved and their relationships. The following folders are included in all the ASP.NET MVC application because the ASP.NET MVC framework by default uses a "convention over configuration" approach, and makes some default assumptions based on folder naming conventions.

1. Once the project is created, review the folder structure that has been created in the Solution Explorer on the right side:



ASP.NET MVC Folder structure in Solution Explorer

   1. **Controllers**. This folder will contain the controller classes. In an MVC based application, controllers are responsible for handling end user interaction, manipulating the model, and ultimately choosing a view to render the UI.

Note

The MVC framework requires the names of all controllers to end with "Controller"- for example, HomeController, LoginController, or ProductController.

2. **Models**. This folder is provided for classes that represent the application model for the MVC Web application. This usually includes code that defines objects and the logic for interacting with the data store. Typically, the actual model objects will be in separate class libraries. However, when you create a new application, you might include classes and then move them into separate class libraries at a later point in the development cycle.
3. **Views**. This folder is the recommended location for views, the components responsible for displaying the application's user interface. Views use .aspx, .ascx, .cshtml and .master files, in addition to any other files that are related to rendering views. Views folder contains a folder for each controller; the folder is named with the controller-name prefix. For example, if you have a controller named **HomeController**, the Views folder will contain a folder named Home. By default, when the ASP.NET MVC framework loads a view, it looks for an .aspx file with the requested view name in the Views\controllerName folder (**Views[ControllerName][Action].aspx**) or (**Views[ControllerName][Action].cshtml**) for Razor Views.

   Note

   In addition to the folders listed previously, an MVC Web application uses the **Global.asax** file to set global URL routing defaults, and it uses the **Web.config** file to configure the application.

Task 3 - Adding a HomeController

In ASP.NET applications that do not use the MVC framework, user interaction is organized around pages, and around raising and handling events from those pages. In contrast, user interaction with ASP.NET MVC applications is organized around controllers and their action methods.

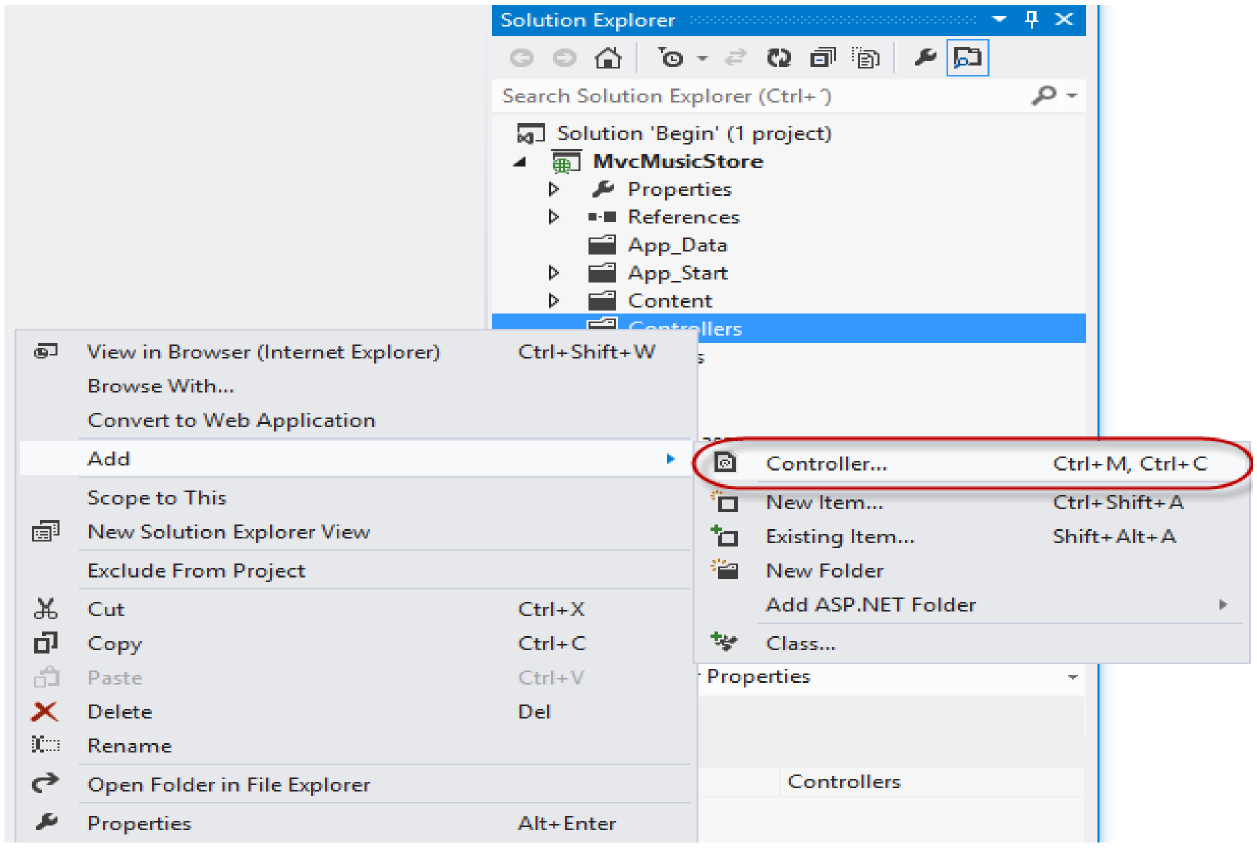
On the other hand, ASP.NET MVC framework maps URLs to classes that are referred to as controllers. Controllers process incoming requests, handle user input and interactions, execute appropriate application logic and determine the response to send back to the client (display HTML, download a file, redirect to a different URL, etc.). In the case of displaying HTML, a controller class typically calls a separate view component to generate the HTML markup for the request. In an MVC application, the view only displays information; the controller handles and responds to user input and interaction.

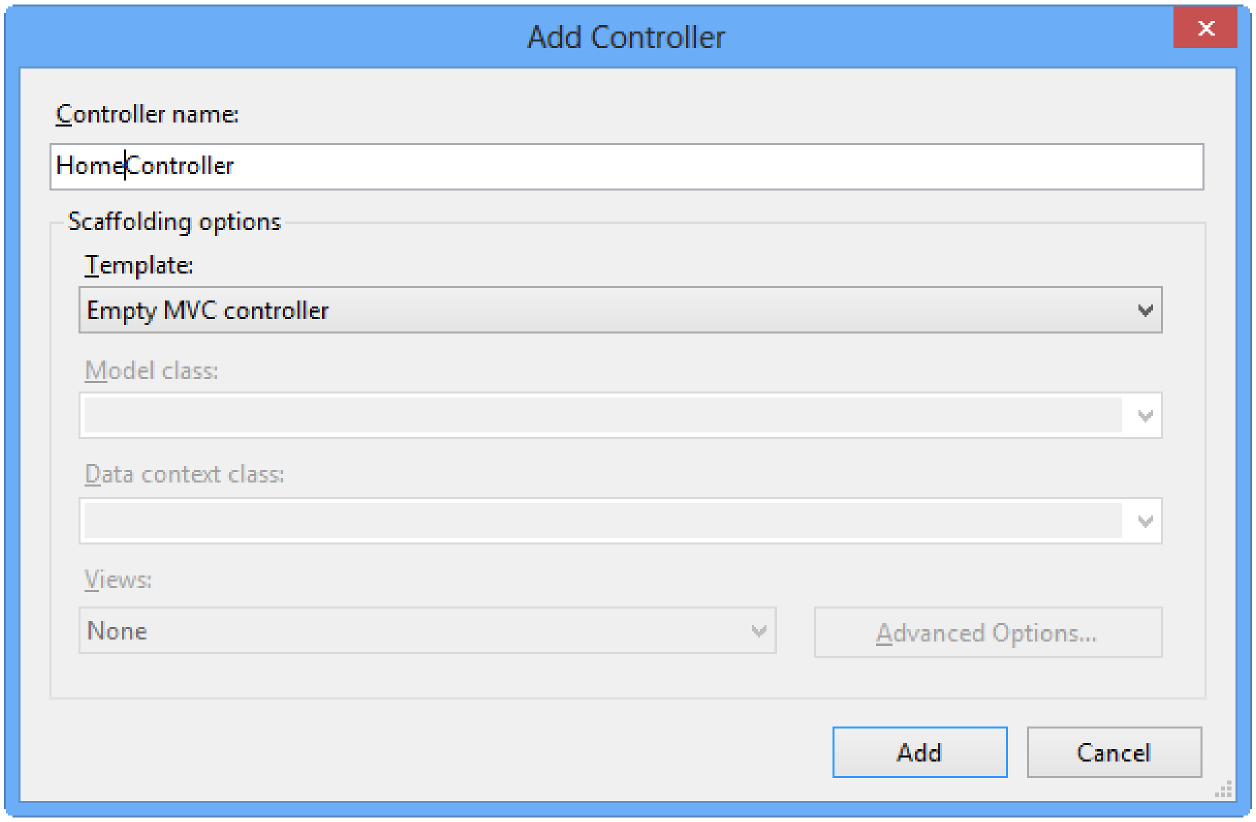
In this task, you will add a Controller class that will handle URLs to the Home page of the Music Store site.

1. Right-click **Controllers** folder within the Solution Explorer, select **Add** and then **Controller** command:

   Add Controller Command

2. The **Add Controller** dialog appears. Name the controller *HomeController* and press **Add**.




Add Controller

Controller name:

HomeController

Scaffolding options

Template:

Empty MVC controller

Model class:

Data context class:

Views:

None

Add Controller Dialog

3. The file **HomeController.cs** is created in the **Controllers** folder. In order to have the **HomeController** return a string on its Index action, replace the **Index** method with the following code:

   (Code Snippet - *ASP.NET MVC 4 Fundamentals - Ex1 HomeController Index*)

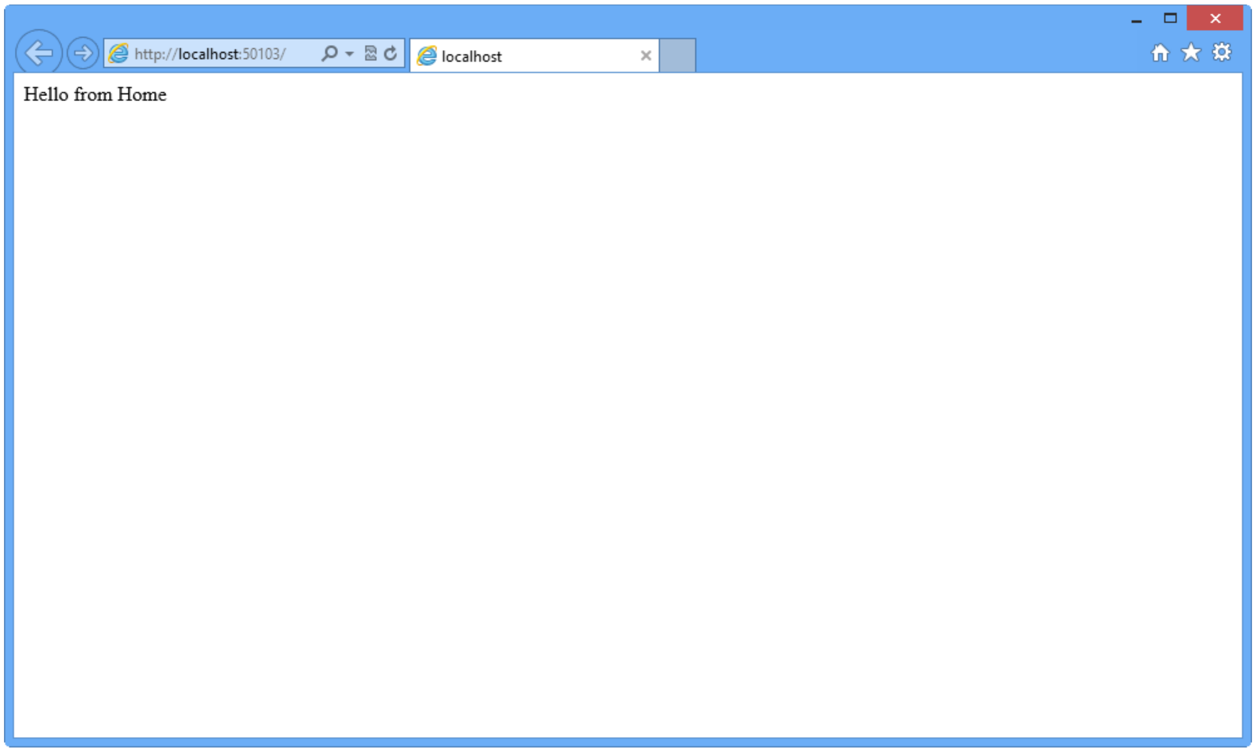
   C#

```
public string Index()
{
    return "Hello from Home";
}
```

Task 4 - Running the Application

In this task, you will try out the Application in a web browser.

1. Press **F5** to run the Application. The project is compiled and the local IIS Web Server starts. The local IIS Web Server will automatically open a web browser pointing to the URL of the Web server.

   Hello from Home

   Application running in a web browser

   Note

   The local IIS Web Server will run the website on a random free port number. In the figure above, the site is running at `http://localhost:50103/`, so it's using port 50103. Your port number may vary.

2. Close the browser.