Embedding Power BI Reports in a Custom Application

Setup Time: 60 minutes

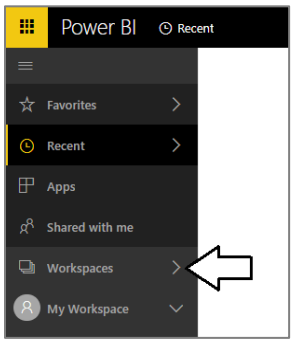
Lab Folder: C:\Student\Modules\08_PBIEmbedding\Lab

Overview:.

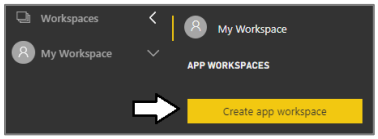
Exercise 1: Create a New App Workspace and Populate It with Content

In this exercise, you will create a new app workspace and then you will work to populate the workspace with content which will include a dataset, a report and a dashboard.

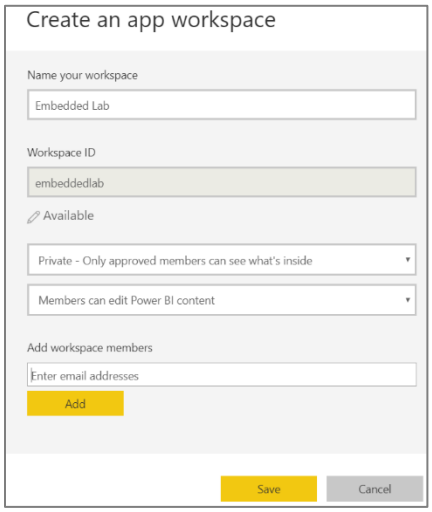
1. Make sure you are logged into the Power BI service with your primary user account.
2. Create a new app workspace named **Embedded Lab**.
   a) Click the **Workspace** flyout menu in the left navigation.



   b) Click the **Create app workspace** button to display the **Create an app workspace** dialog.

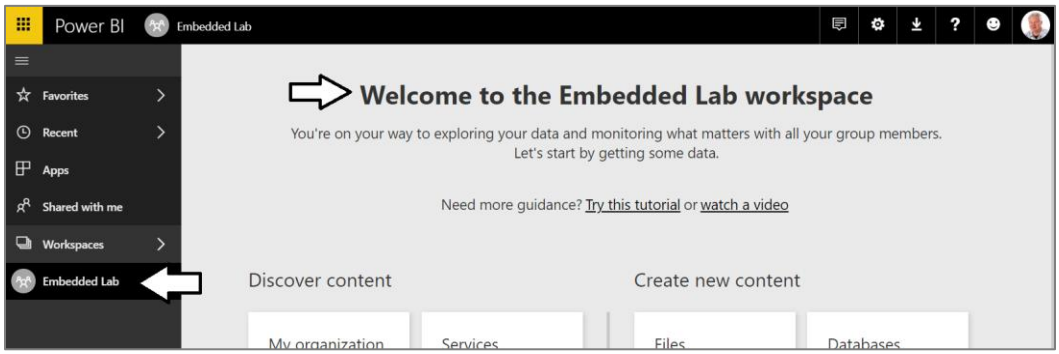


   c) In the **Create an app workspace** pane, enter a name of **Embedded Lab**.
   d) Accept all the other default settings and click **Save**.



Note that when you create a new app workspace, the account you are logged in as is automatically added as a workspace admin.

e) When you click **Save**, the Power BI service should create the new app workspace and then switch your current Power BI session to be running within the context of this new group workspace.



By creating a new workspace, you have created Power BI container for creating and managing the datasets, reports and dashboards.

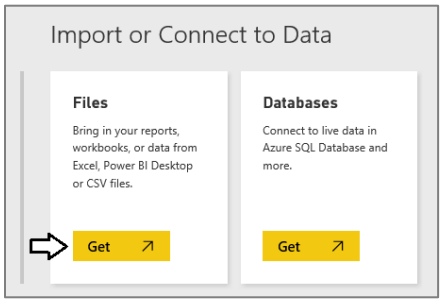
3. Import the **Wingtip Sales Analysis.pbix** project into the **Embedded** app workspace.

a) Using Windows Explorer, verify there is a PBIX file named **Wingtip Sales Analysis.pbix** at the following path.

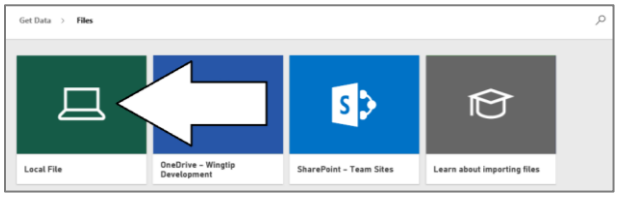
```
C:\Student\PBIX\Wingtip Sales Analysis.pbix
```

b) If you do not have a local copy of the students folder, you can download this PBIX file from here.

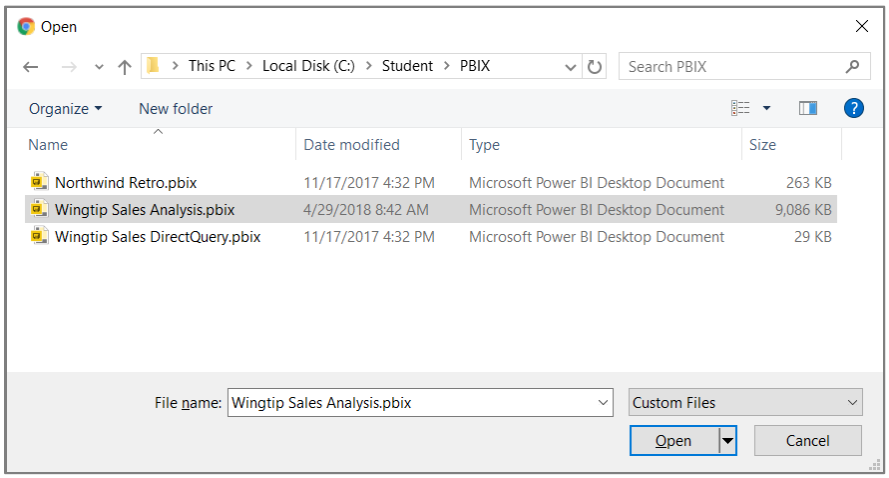
c) In the Power BI Service, click the **Get** button in the **Files** section of the Welcome page.



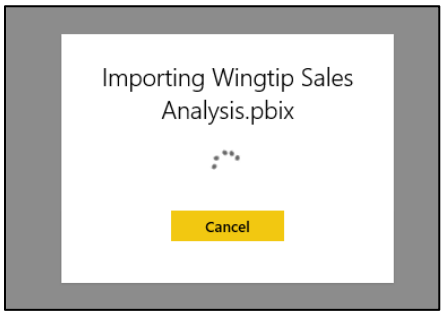
d) On the **Get Data > Files** page, click the **Local File** button to display the Windows **Open** file dialog.



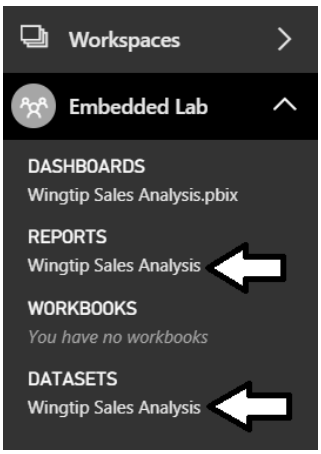
e) In the Windows **Open** file dialog, select the project file at **c:\Student\PBIX\Wingtip Sales Analysis.pbix** and click **Open**.



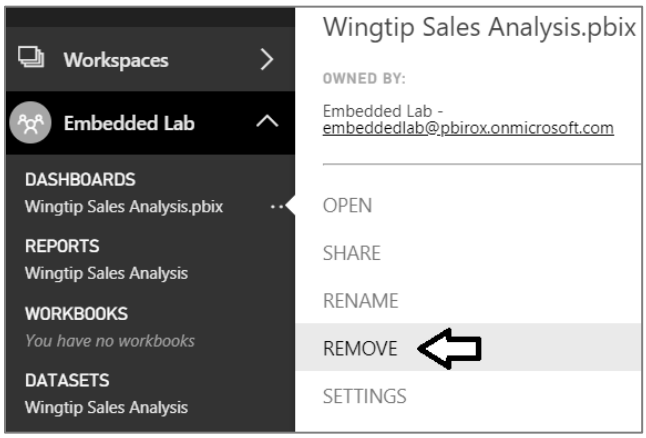
f) Wait while the Power BI service uploads the PBIX files and imports its assets into the **Wingtip Development** group workspace



g) Once the import process completes, you should see a new dataset, a new report and a new dashboard in the left nav menu.

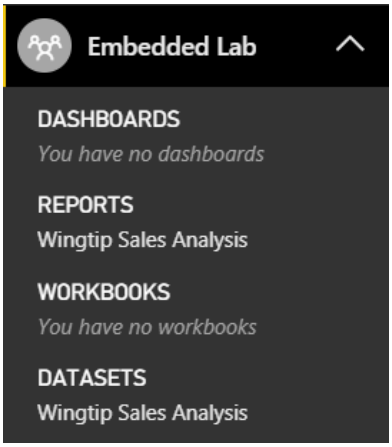


4. Remove the dashboard that was created during the import process.
   a) Dropdown the flyout menu for the **Wingtip Sales Analysis.pbix** dashboard and click the **REMOVE** menu command.

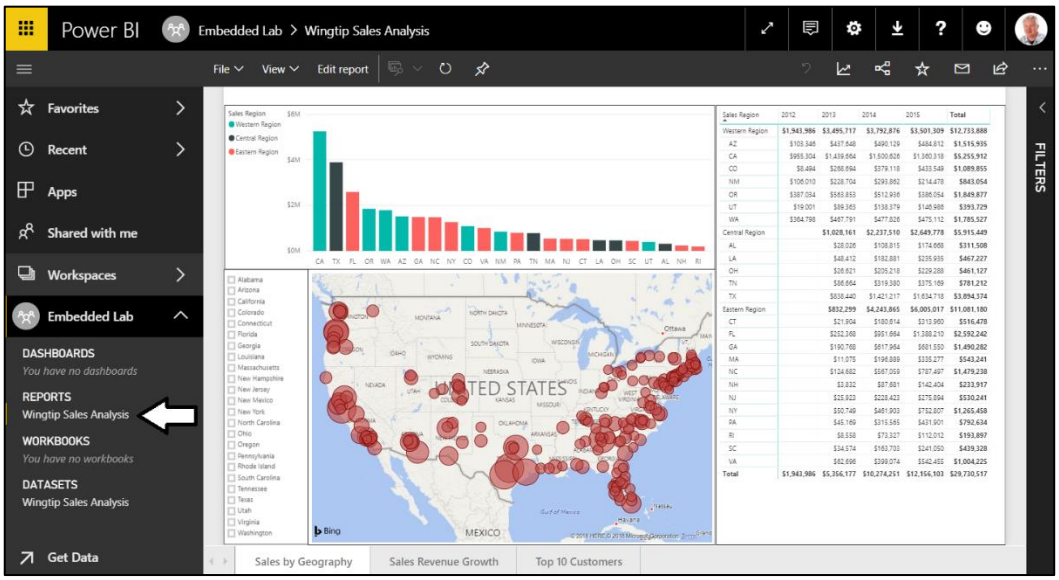


b) Confirm that you want to delete the dashboard by clicking the **Delete** button the **Delete dashboard** dialog.

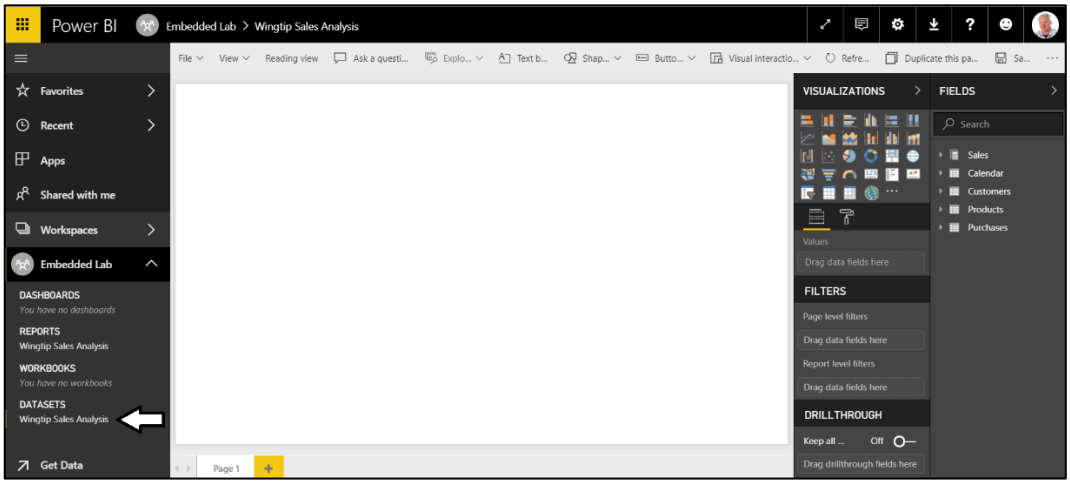
c) You should be able to confirm that the dashboard has been removed.



5. Click on the report named **Wingtip Sales Analysis** in the **Reports** section. Examine the pages in the report.



6. Click on the dataset named **Wingtip Sales Analysis** in the **Datasets** section. The Power BI service responds by displaying a new report that allows you to begin adding visuals.



When you navigate to a dataset in the Power BI service, it provides a different experience compared to when in Power BI Desktop. That's because Power BI Desktop allows you to customize and extend a dataset while the browser-based experience of the Power BI service only allows you to consume datasets but not to modify them. Given the fact that a dataset is a read-only object, the Power BI service responds to user's request to navigate to a dataset by opening a new report and showing the **Fields** list for that dataset.

7.
8.
9.
10. Create a new App workspace.
11. Open Notepad and track the GUID for the workspace ID.
12. In the new workspace, upload the PBIX file named Wingtip Sales Analysis.pbx
13. Creste a new report. Track the GUID for the Dataset ID.
14. Open the report. Track the GUID for the report ID.
15. Create a new Dashboard using visuals from the report.

16. Track the GUID for the report ID
17. Determine what account will be the master user account.
18. If the master user account is a different account than the account you used to create the new app workspace, you must add the account as an admin of the app workspace.
19.

Exercise 2: Create a new Azure Application to Call the Power BI Service API

In this exercise you will create a new Visual Studio project using the ASP.NET MVC framework.

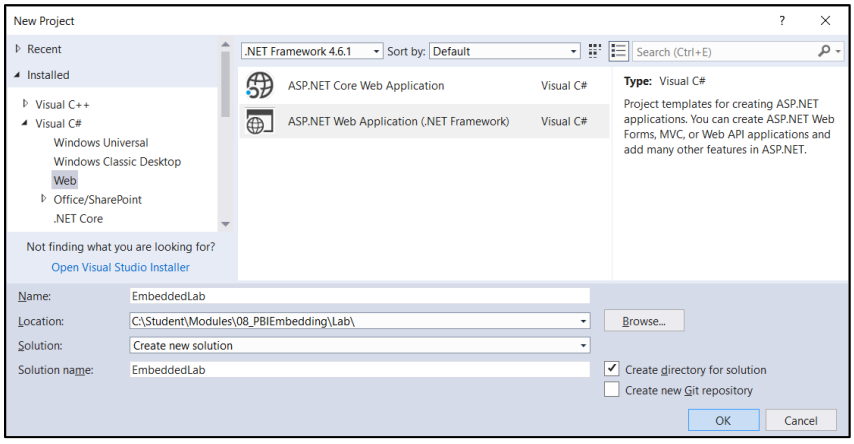
1. Go to Azure portal
2. Create a new Azure Application

3. Track the GUID for the App ID
4. Configure permissions for Power BI Service
5. Grant permissions through the Azure portal to remove the need for an interactive login.

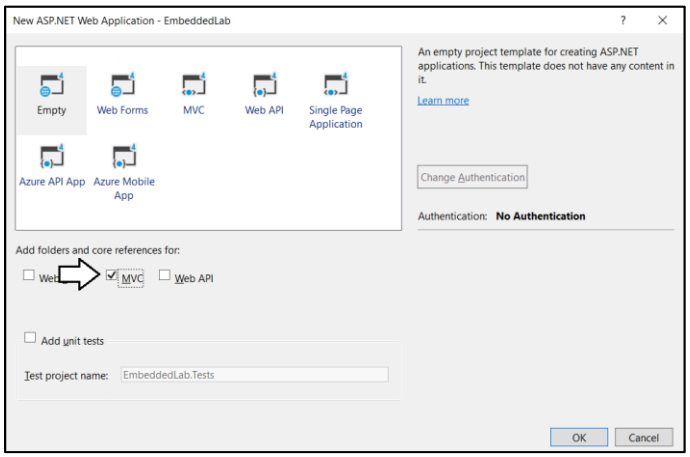
Exercise 3: Create a new MVC Application using Visual Studio 2017

In this exercise you will create a new Visual Studio project using the ASP.NET MVC framework.

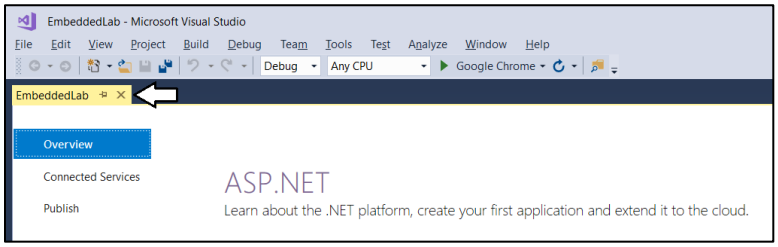
1. Launch **Visual Studio 2017**.
2. Create a new ASP.NET MVC project in Visual Studio 2017.
   a) In Visual Studio select **File > New > Project**.
   b) In the **New Project** dialog:
      i) Select **Installed > Templates > Visual C# > Web**.
      ii) Select the **ASP.NET Web Application** project template.
      iii) Name the new project **EmbeddedLab**.
      iv) Add the new project into the folder at **C:\Student\Modules\08_PBIEmbedding\Lab**.
      v) Click **OK** to display the **New ASP.Net Web Application** wizard.



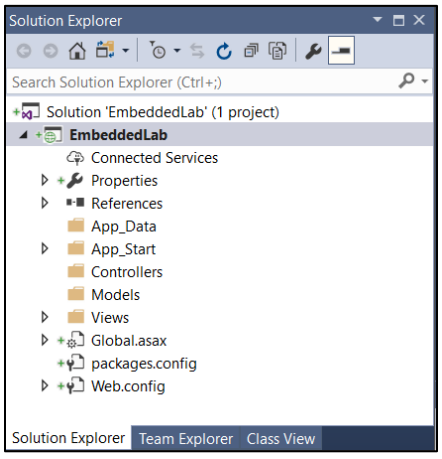
   c) In the **New ASP.Net Web Application** dialog, select the **Empty** template.
   d) In the section with the caption **Add folders and core references**, make sure the **MVC** checkbox is checked.
   e) Click the **OK** button to create the new project.



f) When Visual Studio finishes creating the project, it displays an information page. Close this page by clicking the **x** in the tab.



g) Take a minute to familiarize yourself with the structure of the project in the **Solution Explorer**.



3. Install and Update the Nuget Packages required for developing with Power BI embedding.

a) xx

```
Install-Package Microsoft.IdentityModel.Clients.ActiveDirectory
```

b) xxx

```
Install-Package Microsoft.PowerBI.Api
```

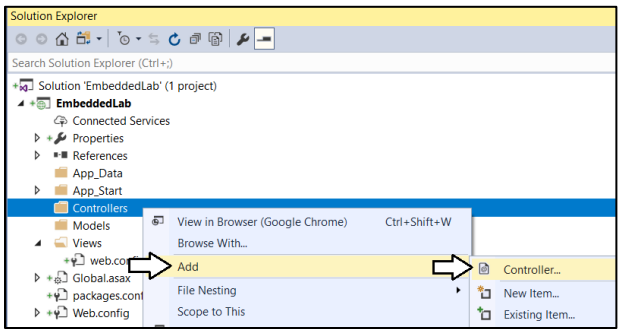
c) x

```
Install-Package Microsoft.PowerBI.JavaScript
```

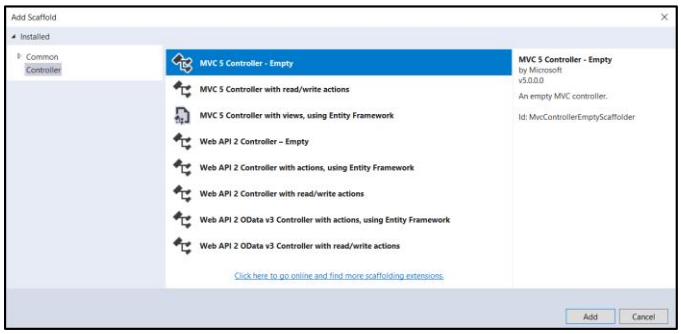
d) x

4.

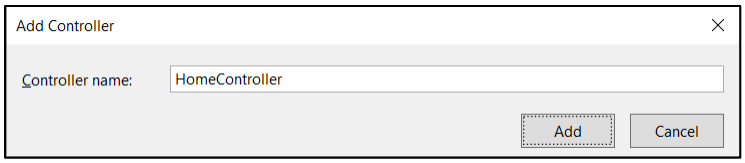
5. Add the Home controller.
   a) In Solution Explorer, right-click on the Controllers folder.



   b) X



   c) X

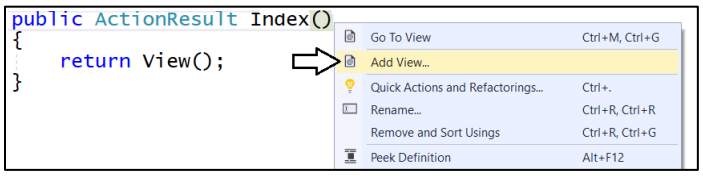


   d) X

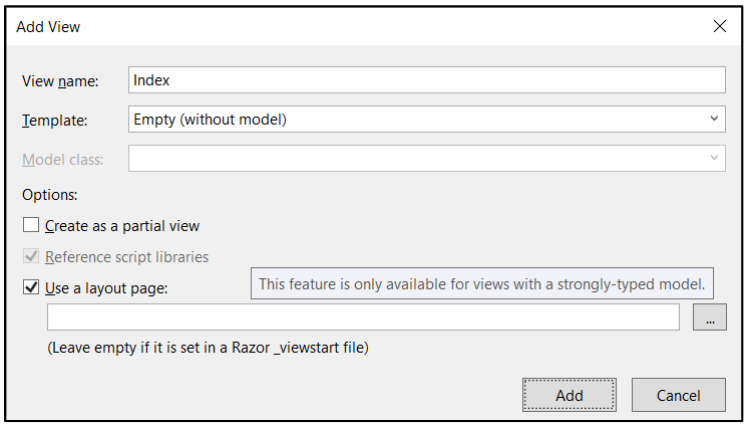
```
namespace EmbeddedLab.Controllers
{
    public class HomeController : Controller
    {
        // GET: Home
        public ActionResult Index()
        {
            return View();
        }
    }
}
```

   e) X

6. Add a view to the Index action method of the Home controller class.
   a) Right-click

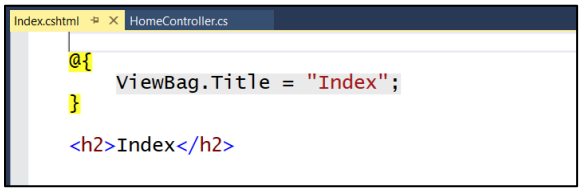


b) X



Up to this point, the EmbeddedLab project has not had a shared layouts page. However, when you create a new view and leave the **Use a layout page** option selected, a new shared layout page is added to the project at **Views/Shared/_Layouts.cshtml**.

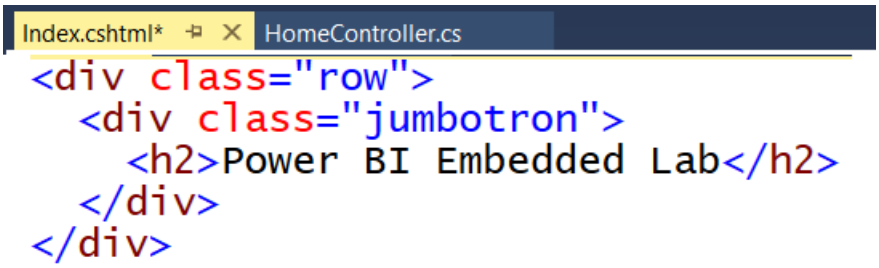
c) Here is the starting point



d) Replace with this.

```
<div class="row">
  <div class="jumbotron">
    <h2>Power BI Embedded Lab</h2>
  </div>
</div>
```

e) D

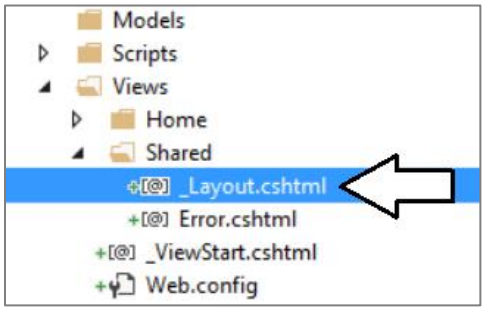


f) Save your changes and close Index.cshtml.

7. Modify the shared layouts page named **_Layouts.cshtml**.

a) In Solution Explorer, expand the **Views** folder and then expand the **Shared** folder.

b) Double-click on **_Layouts.cshtml** to open it in an editor window.



c) Delete the entire contents of **_Layouts.cshtml** and replace with the following HTML starter page.

```
<!DOCTYPE html>
<html>

<head>
</head>

<body>
</body>

</html>
```

d) Copy and paste the following HTML code to provide the **head** section

```
<head>
  <meta charset="utf-8" />
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Embedded Lab</title>
  <link href="~/Content/bootstrap.min.css" rel="stylesheet" type="text/css" />
  <link href="~/Content/Site.css" rel="stylesheet" type="text/css" />
</head>
```

e) Copy and paste the following HTML code to provide the **body** section of the page.

```
<body>

  <!-- add top nav here -->

  <!-- add main body content here -->

  @Scripts.Render("~/bundles/jquery")
  @Scripts.Render("~/bundles/bootstrap")
  @RenderSection("scripts", required: false)

</body>
```

f) Copy and paste the following code into the body just below the **add top nav here** comment.

```
<!-- add top nav here -->
<div class="container">
  <div class="row">
    <div class="navbar navbar-default " role="navigation">
      <div class="container-fluid">
        <div class="navbar-header">
          @Html.ActionLink("Product Manager", "Index", "Home", null, new { @class = "navbar-brand" })
        </div>
        <div class="navbar-collapse collapse">
          <ul class="nav navbar-nav">
            <!-- <li> elements with nav links go here -->
          </ul>
        </div>
      </div>
    </div>
```

```
  </div>
</div>
```

g) Copy and paste the following code into the body just below the **add main body content here** comment

```
<!-- add main body content here -->
<div class="container">
    <div class="container-fluid">
        @RenderBody()
    </div>
</div>
```

h) Save your changes and close **_Layouts.cshtml**.

8. Modify the **HomeController** class.

a) In Solution Explorer, expand the **Controllers** folder and then double-click on **HomeController.cs** open it in an editor window.

b) Delete all the existing code inside **HomeController.cs** and replace it with the following code

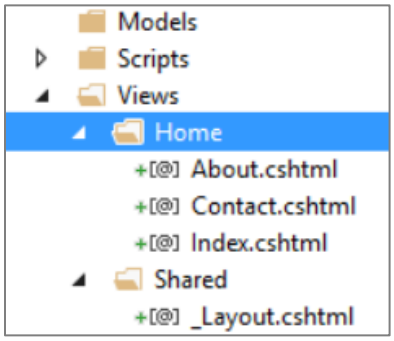
```
using System.Web.Mvc;

namespace ProductManagerMVC.Controllers {
  public class HomeController : Controller {
    public ActionResult Index() {
      return View();
    }
  }
}
```

c) Save and close on **HomeController.cs**.

9. Modify the **Index** view template for the **HomeController** class.

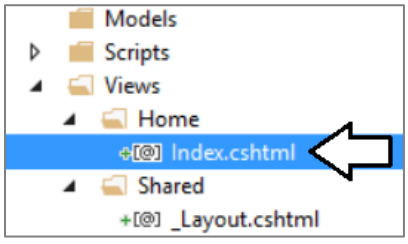
a) In Solution Explorer, expand the **Views** folder and then expand the **Home** folder.

b) You should see three views named **About.cshtml**, **Contact.cshtml** and **Index.cshtml**.



c) Delete the two views named **About.cshtml** and **Contact.cshtml**.

d) Double-click on the view named **Index.cshtml** to open it in an editor window.



e) Delete all the existing content inside **Index.cshtml** and replace it with the following HTML code.

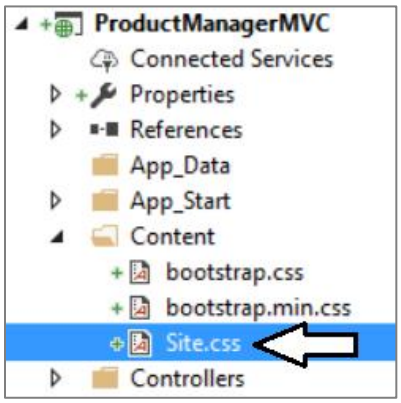
```
<div class="row">
    <div class="jumbotron">
        <h2>Welcome to Product Manager</h2>
```

```
    </div>
</div>

<div class="row">
    <div class="col-md-6">
        <div class="panel panel-primary">
            <div class="panel-heading">Add a New Product</div>
            <div class="panel-body"> Click the Add Product link above to add a new product.</div>
        </div>
    </div>
    <div class="col-md-6">
        <div class="panel panel-primary">
            <div class="panel-heading">See the Product Showcase </div>
            <div class="panel-body"> Click the Product Showcase link to see all Wingtip products.</div>
        </div>
    </div>
</div>
```

10. Modify the **Sites.css** file with a set of custom CSS styles.
    a) Using Windows Explorer, locate the snippet file named **Site.css.txt** in the **Students** at the following location.

    ```
    C:\Student\Modules\03_MvcWebApps\Lab\Snippets\Site.css.txt
    ```

    b) Double click on **Site.css.txt** to open it in Notepad.
    c) Select all the CSS code inside **Site.css.txt** and copy it to the Windows clipboard.
    d) Return to Visual Studio.
    e) In Solution Explorer, expand the **Content** folder and then double-click on **Sites.css** open it in an editor window.

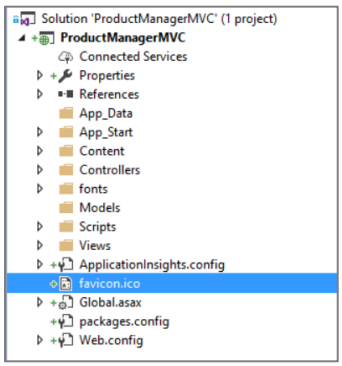


    f) Delete all the existing content from **Sites.css**.
    g) Paste the content of the Windows clipboard into **Sites.css**.
    h) Save your changes and close **Sites.css**.

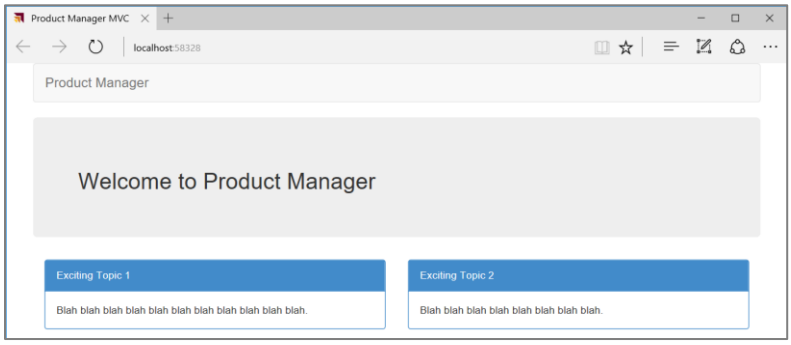
11. Add a **favicon.ico** file to the root folder of the **ProductManagerMVC** project.
    a) Using Windows Explorer, locate the file named **favicon.icon** in the **Students** folder at the following location.

    ```
    C:\Student\Modules\03_MvcWebApps\Lab\StarterFiles\favicon.ico
    ```

    b) Copy the file named **favicon.ico** to the root folder of your project.



12. Test out the **ProductManagerMVC** Project using the Visual Studio Debugger
    a) Press the **{F5}** key to start up the project in the Visual Studio debugger.
    b) When the project starts, the home page should load in the browser and match the following screenshot.



    c) Close the browser, return to Visual Studio and stop the debugger.

Exercise 1: Creating the EmbeddedLab Application using ASP.NET MVC

In this exercise,.

Exercise 2: xxx

In this exercise,.

13. Navigate to the following URL:

    https://go.microsoft.com/fwlink/p/?LinkID=698279&culture=en-US&country=US

Exercise 3: xxx

In this exercise,.

14. Navigate to the following URL:

    https://go.microsoft.com/fwlink/p/?LinkID=698279&culture=en-US&country=US

Exercise 4: xxx

In this exercise,.

15. Navigate to the following URL:

    https://go.microsoft.com/fwlink/p/?LinkID=698279&culture=en-US&country=US