

Optional 9 : Bootstrap

Objective

In this exercise you will Bootstrap the QA Forum website; you will use various Bootstrap styles to customise the look and feel of the website. You will use the Bootstrap's grid system to control the layout of the content and implement the forms for the website's create, update and delete pages.

This exercise will take around 30 minutes.

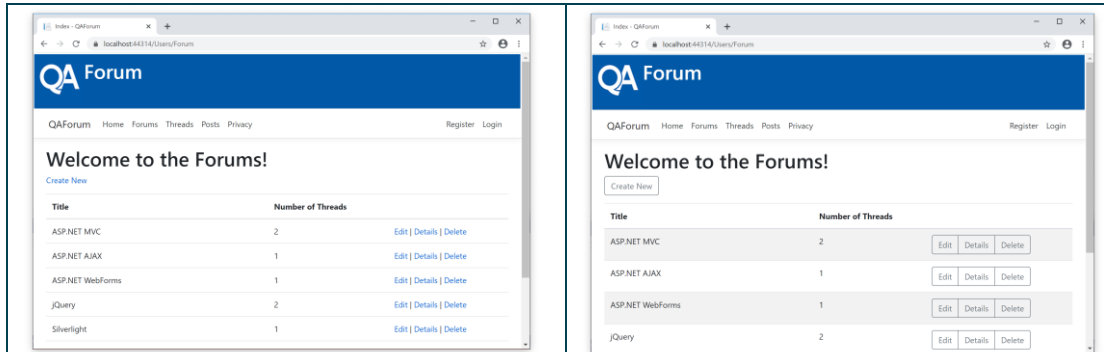
Referenced material

This exercise is based on material from the chapter "Bootstrap".

Actions

1	Open the 'Begin' solution in Visual Studio and compile (Shift Ctrl+B). The 'Begin' solution is identical to the 'End' solution from the previous lab.
2	<p>Modify the forums index view: Open the file <code>/Areas/Users/Views/Forum/Index.cshtml</code> Modify the Create link at the top of the page with the following code:</p> <pre><p> <a asp-action="Create" class="btn btn-outline-secondary">Create New </p></pre> <p>We are just adding the bootstrap button style to the generated html. Add the following class to the table element:</p> <pre><table class="table table-striped"></pre> <p>Modify the ActionLinks at the bottom of the page:</p> <pre><td> <div class="btn-toolbar"> <div class="btn-group"> @Html.ActionLink("Edit", "Edit", new { id = item.ForumId }, new { @class = "btn btn-outline-secondary" }) @Html.ActionLink("Details", "Details", new { id = item.ForumId }, new { @class = "btn btn-outline-secondary" }) @Html.ActionLink("Delete", "Delete", new { id = item.ForumId }, new { @class = "btn btn-outline-secondary" }) </div> </div> </td></pre> <p>Here we are adding a button group, at the end of each row. Note that the parameter after the route values is used to add HTML attributes to the generated tag. Also, we use an anonymous class, which contains a property called "class".</p>

But since “class” is a reserved word in C#, we have to prefix it with an @ sign to use it as the name of a property.
 Press **F5** to build and browse your changes
 Navigate to the **Forum** page to view the changes to the button look:
Was and Now:



3

We are going to make the partial view of Threads more generic.
 We are going to move the Create link at the top of the file out of the partial view.

Delete this from **Areas/Users/Views/Thread/_PartialThreadList.cshtml**

```
<p>
  <a asp-action="Create">Create New</a>
</p>
```

In **Views/Thread/Index.cshtml** insert this

```
<h1>Index</h1>

<p>
  <a asp-action="Create" class="btn btn-outline-secondary">Create New</a>
</p>

<partial name="_PartialThreadList" />
```

This means that when you look at the list of all the threads, you'll see the Create button. But when the list of threads is at the bottom of a forum details page, the Create button will no longer be shown.

4

Modify the partial view (**Views/Threads/_PartialThreadList.cshtml**):
 First of all, delete the entire <thead> section
 Then, replace the contents of the foreach loop as shown below:

```
@model IEnumerable<QAForum.Areas.Users.ViewModels.ThreadViewModel>

<table class="table">
  <tbody>
    @foreach (var item in Model)
    {
      <tr>
        <td>
          @Html.DisplayFor(modelItem => item.Title)<br />
          <small>by @Html.DisplayFor(modelItem => item.UserName)</small>
        </td>
        <td>
          <div class="btn-toolbar">
            <div class="btn-group btn-group-xs">
              @Html.ActionLink("Edit", "Edit", "Thread",
                new { id = item.ThreadId }, new { @class = "btn btn-outline-secondary" })
            </div>
          </div>
        </td>
      </tr>
    }
  </tbody>
</table>
```

```

                @Html.ActionLink("Details", "Details", "Thread",
                new { id = item.ThreadId }, new { @class = "btn btn-outline-secondary" })

                @Html.ActionLink("Delete", "Delete", "Thread",
                new { id = item.ThreadId }, new { @class = "btn btn-outline-secondary" })
            </div>
        </div>
    </td>
</tr>
}
</tbody>
</table>

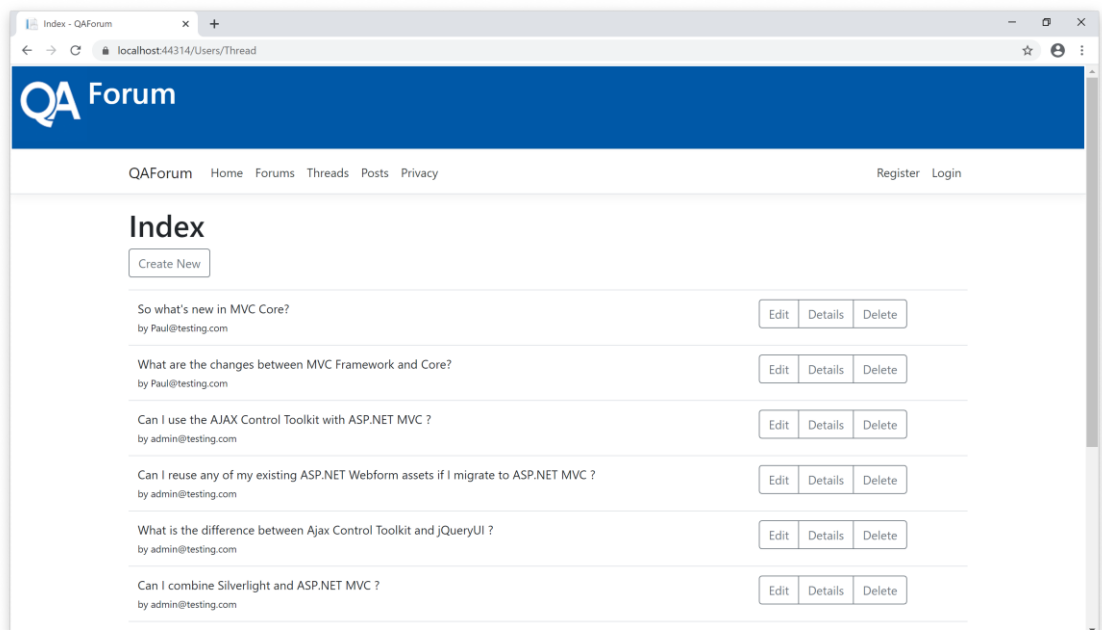
```

We are changing the layout of this page so that instead of rows made up of columns, it presents the list of threads in a more visually appealing way.

5

Run and select the Threads menu.

You should see this:



6

Modify the Posts index view:

Just as we did for the Thread views, move the Create link at the top of the **Views/Shared/Components/PostList/Default.cshtml** into **Views/Post/Index.cshtml** and modify as shown

```

<h1>Index</h1>

<p>
    <a asp-action="Create" class="btn btn-outline-secondary">Create New</a>
</p>

<vc:post-list thread-id="null"></vc:post-list>

```

7

Modify the component's partial view (**PostList/Default.cshtml**):

This time instead of using a table for controlling the layout of partial view, we are going to use Bootstrap's grid system of rows and columns.

Replace the content of whole file with the following mark-up:

```

@model IEnumerable<QAForum.Areas.Users.ViewModels.PostViewModel>

@foreach (var item in Model)
{

```

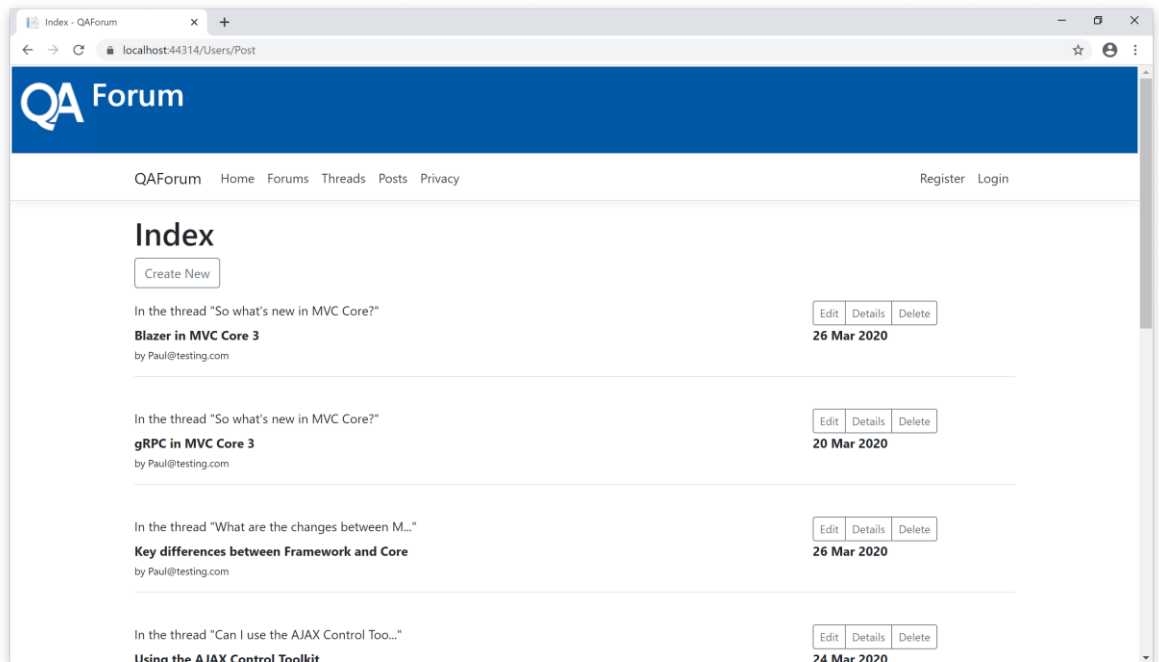
```

<div class="row">
  <div class="col-sm-9">
    In the thread "@Html.DisplayFor(modelItem => item.Thread)"
  </div>
  <div class="col-sm-3">
    <div class="btn-toolbar">
      <div class="btn-group btn-group-sm">
        @Html.ActionLink("Edit", "Edit", "Post",
          new { id = item.PostId }, new { @class = "btn btn-outline-secondary" })
        @Html.ActionLink("Details", "Details", "Post",
          new { id = item.PostId }, new { @class = "btn btn-outline-secondary" })
        @Html.ActionLink("Delete", "Delete", "Post",
          new { id = item.PostId }, new { @class = "btn btn-outline-secondary" })
      </div>
    </div>
  </div>
</div>
<div class="row">
  <div class="col-sm-9">
    <strong>@Html.DisplayFor(modelItem => item.Title)</strong>
    <br />
    <small>
      by @Html.DisplayFor(modelItem => item.UserName)
    </small>
  </div>
  <div class="col-sm-3">
    @Html.DisplayFor(modelItem => item.PostDateTime)
  </div>
</div>
<hr />
<br />
}

```

Here we are using Bootstrap's grid system to control the layout of the elements. This is usually preferred to tables, because it can be “responsive”, i.e. it can adjust

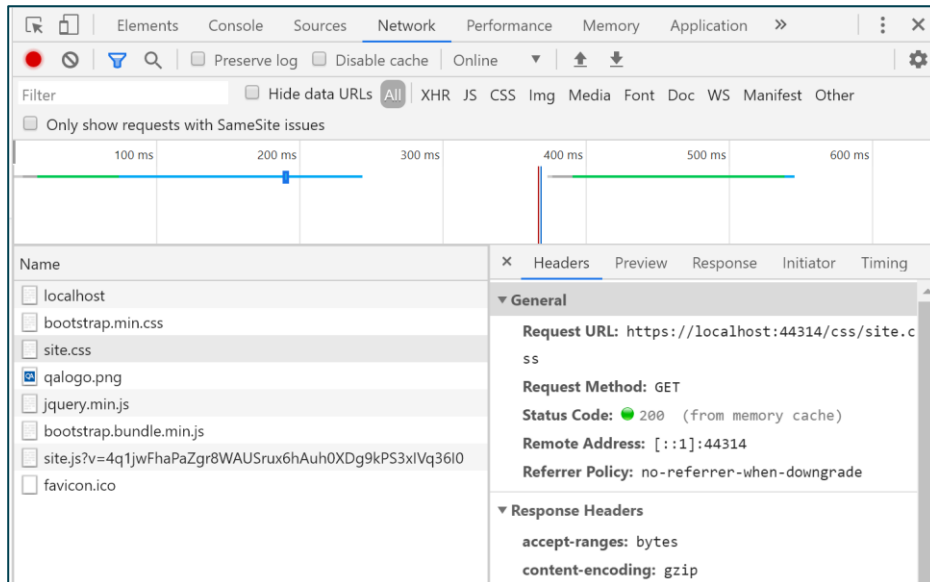
its layout depending on screen size, so that your website will look good on both mobile and desktop.
Press **F5** to build and browse the changes. Select the Post menu and view the changes:



If You Have Time: Bundling and Minifications

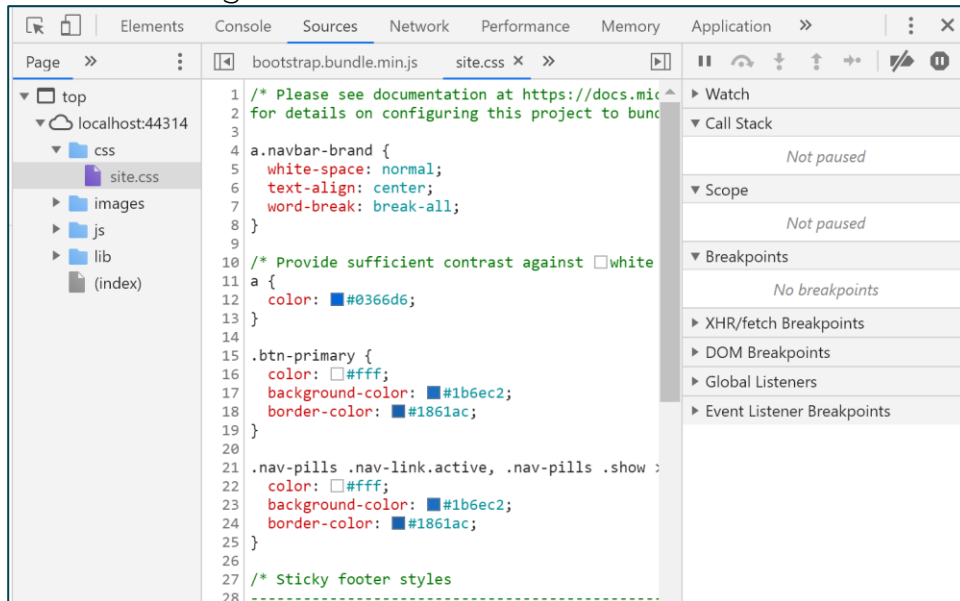
Run your program. In your web browser, open the developer tools – in most web browsers, you can do this using the F12 key. In the developer tools, go to the Network tab. Then, with your Network tab showing, refresh your web browser to reload the home page of the application.

The Network tab shows each individual request that the web browser makes to the server:



Wow, that's a lot of individual requests for a very simple page! Taking a look through them, we can see there's a request to "localhost" which is the request for the main web page. Then two requests for different CSS files. One request for the image in the header. Two separate requests for JavaScript files. And then a request for the "favicon" – the icon that's used to represent the website in places like the Favourites list, and also (depending on the web browser) in the web page's browser tab.

In Developer Tools, click on Sources, then navigation to CSS/Site.css. The exact way you do this will depend a little on the browser, but you should see something like this:



Hopefully, that file should look familiar. It's a file that we modified in an earlier lab, containing CSS that we added to our website (as well as some CSS that was in the website template).

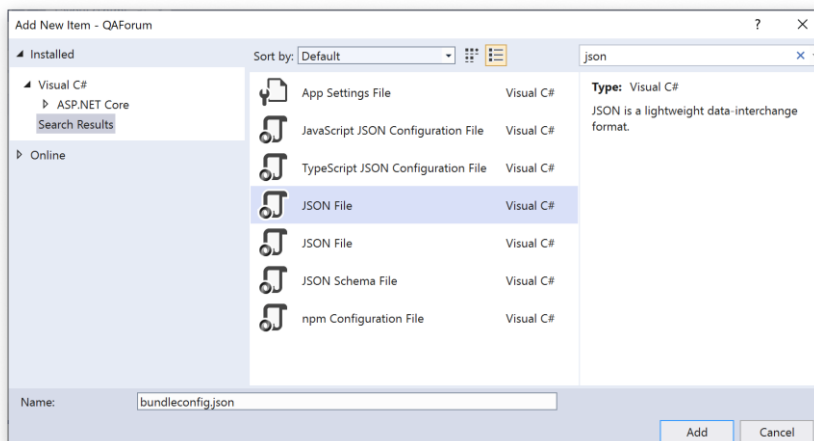
Close down your web browser.

Let's see if we can reduce the number of individual requests. There's no benefit in having our web page load two separate CSS files – if we requested just a single file, which had the combined contents of the two original files, we could considerably reduce the load time of the web page.

Add the following NuGet package to your project. This package will build a “bundle” for you:

BuildBundleMinifier

Right-click on the QAForum project, and select Add, New Item. In the search box, type “json”, and add a JSON file called bundleconfig.json:



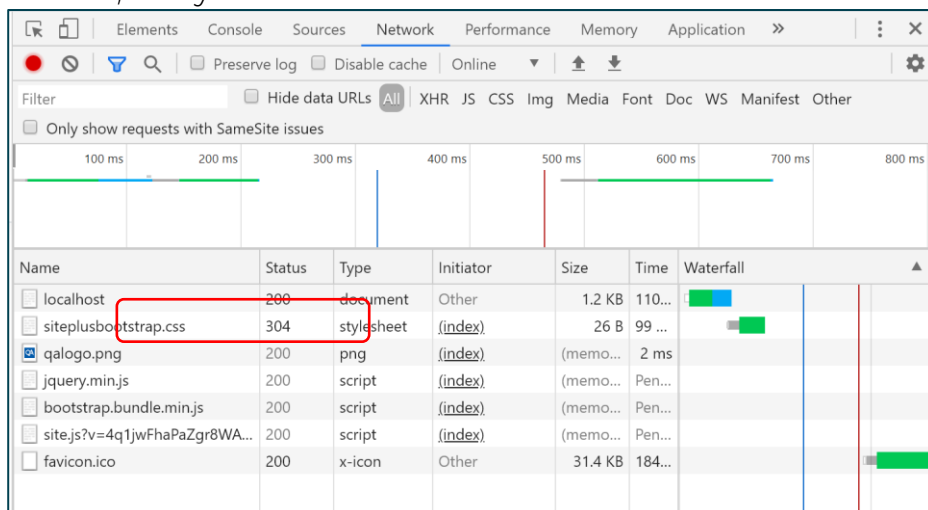
Paste the following into the new file, replacing its existing contents:

```
[
{
  "outputFileName": "wwwroot/css/siteplusbootstrap.css",
  "inputFiles": [
    "wwwroot/lib/bootstrap/dist/css/bootstrap.css",
    "wwwroot/css/site.css"
  ],
  "minify": {
    "enabled": false
  }
}
```

Modify _Layout.cshtml to use the new bundle instead of the two separate CSS links:

```
<link rel="stylesheet" href="~/lib/bootstrap/dist/css/bootstrap.min.css" />
<link rel="stylesheet" href="~/css/site.css" />
<link rel="stylesheet" href="~/css/siteplusbootstrap.css" />
```

Run your program, and once again use the Network tab in Developer Tools to see what's happening. This time, there's only one CSS page loaded, but your website looks identical!



The screenshot shows the Chrome DevTools Network tab. The filter is set to 'All'. The list of requests includes: localhost (200, document, 1.2 KB, 110 ms), siteplusbootstrap.css (304, stylesheet, 26 B, 99 ms), qalogo.png (200, png, 2 ms), jquery.min.js (200, script, Pen...), bootstrap.bundle.min.js (200, script, Pen...), site.js?v=4q1jwFhaPaZgr8WA... (200, script, Pen...), and favicon.ico (200, x-icon, 31.4 KB, 184 ms). A red box highlights the 'siteplusbootstrap.css' request.

Name	Status	Type	Initiator	Size	Time	Waterfall
localhost	200	document	Other	1.2 KB	110...	
siteplusbootstrap.css	304	stylesheet	(index)	26 B	99 ...	
qalogo.png	200	png	(index)	(memo...	2 ms	
jquery.min.js	200	script	(index)	(memo...	Pen...	
bootstrap.bundle.min.js	200	script	(index)	(memo...	Pen...	
site.js?v=4q1jwFhaPaZgr8WA...	200	script	(index)	(memo...	Pen...	
favicon.ico	200	x-icon	Other	31.4 KB	184...	

In the Sources tab, you can see the source file "siteplusbootstrap.css". It looks like the bootstrap.css file, complete with copyright comments at the top. But if you scroll right down to the bottom, you'll find the contents of site.css, including the two styles we added in an earlier lab.

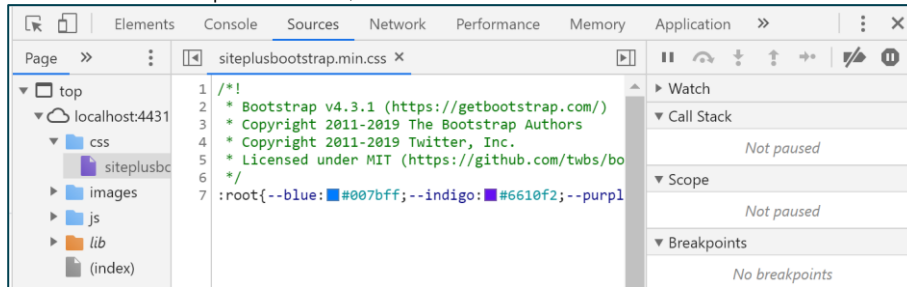
We can do even better! Modify bundleconfig.json to allow "minification" – this means that any characters that can be removed from the CSS will be, including removing comments and white space, and shortening variables names where possible:

```
"minify": {
  "enabled": true
}
```

Then, modify the layout page to use the minified version of the file:

```
<title>@ViewData["Title"] - QAForum</title>
<link rel="stylesheet" href="~/css/siteplusbootstrap.min.css" />
</head>
```


Run your program again. It still looks the same, but now go to the Sources tab in developer tools, and look our bundle:



Microsoft's research suggests that, for a typical website, bundling and minification can reduce the amount of data transmitted when loading assets by 41%, and can reduce the load time of those assets by 62%.

With minification being so important, let's add it for the site's JavaScript file too. That file is empty right now, but it's good practice to enable minification ready for later use anyway.

Add this section to bundleconfig.json:

```
    "wwwroot/css/site.css"
  ],
  "minify": {
    "enabled": true
  },
  {
    "outputFileName": "wwwroot/js/site.min.js",
    "inputFiles": [
      "wwwroot/js/site.js"
    ],
    "minify": {
      "enabled": true,
      "renameLocals": true
    },
    "sourceMap": false
  }
]
```

And then modify this line near the end of _Layout.cshtml to use the minified JavaScript file:

```
<script src="~/lib/jquery/dist/jquery.min.js"></script>
<script src="~/lib/bootstrap/dist/js/bootstrap.bundle.min.js"></script>
<script src="~/js/site.min.js" asp-append-version="true"></script>
@RenderSection("Scripts", required: false)
</body>
</html>
```

Don't expect to notice any change to your website when you do this – we haven't put anything into that file. But it's nice to know that any code we add in the future will be minified.

The only problem with bundling and minification is that it can make client-side debugging quite tricky. So let's change _Layout.cshtml so that bundling and minification are only used for production code, not during development. Make these changes at the top of the file for the CSS:

```
<title>@ViewData["Title"] - QAForum</title>
<environment include="Development">
  <link rel="stylesheet" href="~/lib/bootstrap/dist/css/bootstrap.min.css" />
  <link rel="stylesheet" href="~/css/site.css" />
</environment>
<environment exclude="Development">
  <link rel="stylesheet" href="~/css/siteplusbootstrap.min.css" />
</environment>
</head>
```

And make these changes at the bottom of the file for the JavaScript:

```
<environment include="Development">
  <script src="~/js/site.js" asp-append-version="true"></script>
</environment>
<environment exclude="Development">
  <script src="~/js/site.min.js" asp-append-version="true"></script>
</environment>
```

Run the project. Use the Network tab in Developer Tools to confirm that you are no longer being served bundled or minified files – great for development.

Stop the project. Open the Project > Properties and go to the Debug tab. Delete the word “Development” from the Environment Variables section of this tab – just leave it empty.

Configuration: N/A Platform: N/A

Profile: IIS Express [New... Delete]

Launch: Project

Application arguments: Arguments to be passed to the application

Working directory: Absolute path to working directory [Browse...]

☒ Launch browser: Absolute or relative URL

Environment variables:

Name	Value
! ASPNETCORE_ENVIRONMENT	Value

[Add Remove]

Another run of your program, and the Developer Tools network tab should confirm you are now receiving bundled and minified resources.

20 Finally, put back the Development setting.

In this chapter, we have seen how we can apply styling to our web pages using Bootstrap, in order to customise the look and feel of each page.

We have also seen how to reduce network usage and page load times by using bundling and minification.