Exercise 5c : Create QAForum App

Objective

In this lab we use the understandings so far to create a web app using an EntityFramework database. The Web App is an online forum, so consists of Forums, Threads and Posts

This exercise will take around 60 minutes.

1 Start a new ASP.NET Core Web App called “QAForum”.
Set Authentication to “Individual User Accounts”

2 Add a new area to the Areas folder, called Users.

In the new area, delete the Data folder, and rename Models to ViewModels.

3 Add a new controller to Areas/Users/Controllers. Call the controller ForumController, and use the MVC Controller – Empty template.

Add the [Area] attribute to the controller:

```
    [Area("Users")]
    public class ForumController : Controller
```

4 Right-click on the Index action, and add a view. Leave the template set to Empty (Without Model), and just create a default view. We will update this to something more useful soon.

5 Let’s customise our website’s appearance. Open Views/Home/Index.cshtml. Change the line with the link so that it links to our forums:

```
<div class="text-center">
    <h1 class="display-4">Welcome</h1>
    <p>Please <a asp-area="Users" asp-controller="Forum" asp-
action="Index">click here to see the Forums</a></p>
</div>
```

Now, open Views/Shared/_Layout.cshtml, and add a link in the navigation bar so that users can get to the forums:

```
<ul class="navbar-nav flex-grow-1">
    <li class="nav-item">
        <a class="nav-link text-dark" asp-area="" asp-controller="Home" asp-
action="Index">Home</a>
    </li>
    <li class="nav-item">
        <a class="nav-link text-dark" asp-area="Users" asp-controller="Forum"
asp-action="Index">Forums</a>
    </li>
    <li class="nav-item">
        <a class="nav-link text-dark" asp-area="" asp-controller="Home" asp-
action="Privacy">Privacy</a>
    </li>
</ul>
```

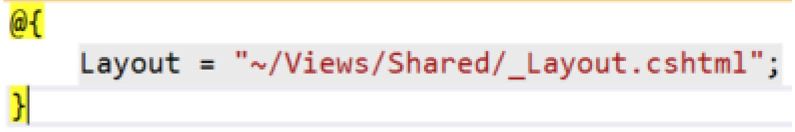
6 We need to customise our routes so that we can route to controllers inside an area. In the Program.cs file, add a new controller route alongside the others:

```
    app.MapControllerRoute(
         name: "areas",
         pattern: "{area:exists}/{controller}/{action=Index}/{id?}");
    app.MapControllerRoute(
         name: "default",
         pattern: "{controller=Home}/{action=Index}/{id?}");
    app.MapRazorPages();
```

Press F5. When the home page appears, click on the link that says “Click here to see the Forums”. It takes you to a new page, but it doesn’t look like part of our web site. It doesn’t have the navigation bar, or the copyright message, or the same fonts as the home page, for example.

There was an extra step that we needed to do to use views in our area, which we saw in the previous lab. Do you remember what it was?

7 What we hadn’t done was copy the _ViewStart.cshtml and

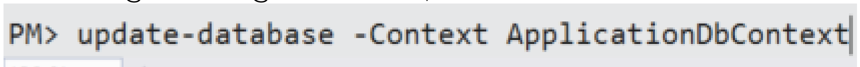
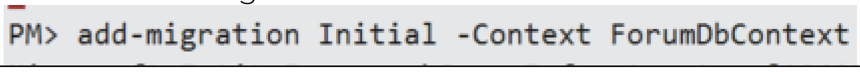
	_ViewImports.cshtml from /Views to the Areas/Users/Views.
	We also need to modify the contents of ViewStart.cshtml:
	
	`@{`
	`    Layout = "~/Views/Shared/_Layout.cshtml";`
	`}`
	
	Now try again – it should look a lot better.
	
	Make sure that all of the links in the navigation bar work too.

8	In the Assets folder, we have provided the 3 model files in a Models folder. Drag these into the Models folder
9	There is also an EF folder. Drag this onto your project and look at the ForumDbContext

10	Now go to the method Program.cs and configure the service. Note how SqlServer is configured for the authentication (the “DefaultConnection”). Using this as your template, do the same for ForumDbContext, using the connection string name “Forum.Data”. So it looks like this:
	
	`builder.Services.AddDbContext<ForumDbContext>(options =>`
	`    options.UseSqlServer(`
	`        builder.Configuration.GetConnectionString("Forum.Data")));`

11	Change “DefaultConnection” to “Forum.Users”
	
	`var connectionString =`
	`builder.Configuration.GetConnectionString("Forum.Users") ?? throw new`
	`InvalidOperationException("Connection string 'Forum.Users' not found.");`

12	In appsettings.json, change the connection string name “DefaultConnection” to “Forum.Users” and its value to this:
	`"Forum.Users":`
	`"Server=.\\SqlExpress;Database=Forum.Users;Trusted_Connection=True;`
	`                               MultipleActiveResultSets=true"`
	NB the connection string here is only across 2 lines here because of the width constraint of an A4 page. Make sure when you copy this line that it does not have line breaks in it.

13	Copy this connection string and change the name to “Forum.Data”; change the database to “Forum.Data”
14	In Package Manager Console, create the Forum.Users database  `PM> update-database -Context ApplicationDbContext`
15	Add an initial migration for the Forum.Data database  `PM> add-migration Initial -Context ForumDbContext`
16	Add a C# Console App (.NET Core) called “SeedDatabase” to the solution.
17	In the SeedDatabase project, install the NuGet packages : Microsoft.EntityFrameworkCore.SqlServer Microsoft.EntityFrameworkCore.Tools And add a reference to QAForum. Ensure all compiles.

18	In the Assets folder is a file Program.cs. Drag this over the one you have in the SeedDatabase project. This project will create the database for us by running the Entity Framework migrations. It will also add some sample data that we can use throughout the development process. (Note that Entity Framework has a built-in technique for seeding data, which is not covered on this course, and also would not be suitable here, because this seed data is only intended to be used during development, and will not be included when we release our system. If you need to add data to your production database as a one-off process when the database is created, you can find out more at https://docs.microsoft.com/en-us/ef/core/modeling/data-seeding.)
19	Right-click on the SeedDatabase project, and select Set As Startup Project
20	Press F5 to run the SeedDatabase program – then, using SSMS, check that your Forum.Data database has Forum, Thread and Post records in it. Change the startup program back to QAForum.
21	Now, let's show some data. Modify the ForumController so that it gets an instance of the data context injected into it. Then pass the list of Forums into the view:

```
    [Area("Users")]
    public class ForumController : Controller
    {
        private readonly ForumDbContext context;

        public ForumController(ForumDbContext context)
        {
            this.context = context;
        }

        public IActionResult Index()
        {
            return View(context.Forums);
        }
    }
```

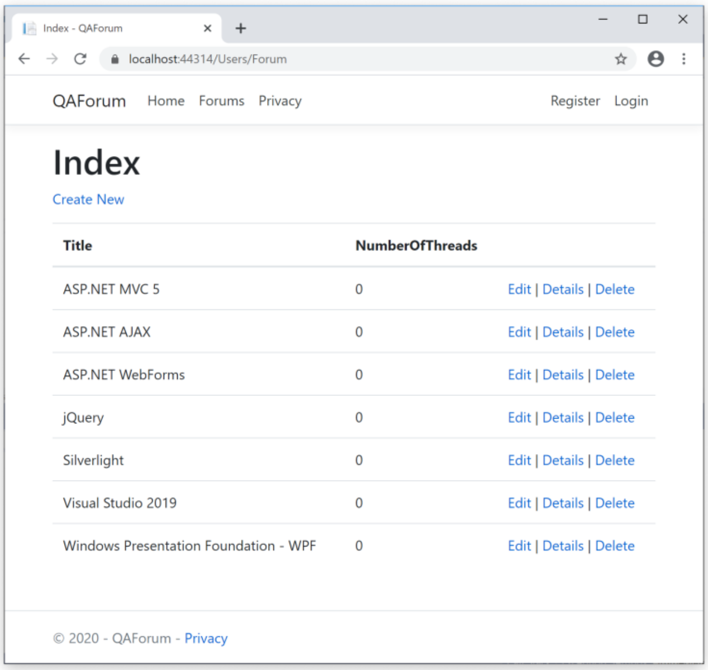
	Then, right-click on the Index action, and select Add View. Choose the appropriate options… see if you can work out what they are before you look down. (You will need to confirm that you want to overwrite the existing view.)
22	Did you choose the List template, and the Forum class? If so, well done! Run the program, and check you can see the list of forums!
23	In an earlier lab, we saw that using a view-model gives us much more control over how a specific view looks. Let's reinstate the view-model class from that earlier lab. You will find the code in Assets/ViewModels/ForumViewModel.cs. Drag that into the Areas/Users/ViewModels folder. It's almost identical to the one from the previous lab, with the namespaces updated for this lab and with the "Date of latest thread" removed. Modify the controller to use the view-model:

```
        public IActionResult Index()
        {
            return View(ForumViewModel.FromForums(context.Forums));
        }
```

	And then re-create the view, using the List template but the ForumViewModel

class.

Run the application, and go to the Forum page:



Index

Create New

Title	NumberOfThreads	
ASP.NET MVC 5	0	Edit | Details | Delete
ASP.NET AJAX	0	Edit | Details | Delete
ASP.NET WebForms	0	Edit | Details | Delete
jQuery	0	Edit | Details | Delete
Silverlight	0	Edit | Details | Delete
Visual Studio 2019	0	Edit | Details | Delete
Windows Presentation Foundation - WPF	0	Edit | Details | Delete

© 2020 - QAForum - Privacy

Do you know why the number of threads is showing as 0 for every forum? We know this isn't true, because the seed data definitely added threads to the forums.

24 Use the Nuget Package Manager to check which version of the Microsoft.EntityFrameworkCore.SqlServer package is installed. Add the following NuGet package, but be sure to add the same version number as the version of Entity Framework that's already installed:

Microsoft.EntityFrameworkCore.Proxies

Then, add the following code to ForumDbContext:

```
protected override void OnConfiguring(DbContextOptionsBuilder optionsBuilder)
{
    optionsBuilder.UseLazyLoadingProxies();
    base.OnConfiguring(optionsBuilder);
}
```

Run the program again. That's better!

Remember, this solution results in many separate queries being sent to the database. It's fine for the small amount of data that we have here, but a better solution if you have larger amounts of data might be to change the controller as follows:

```
return View(ForumViewModel.FromForums(context.Forums.Include(f => f.Threads)));
```

This alternative method results in just a single request to the database, which brings back the forums and the threads together. But we will stick with lazy loading proxies for the rest of the course, because it has the advantage of being done just once and then working through the whole application without having to think about it again.

25	In the Assets/ViewModels folder are two more view-models, one for threads and one for posts. Add these to the ViewModels folder and take a look at them. They mostly contain the same data as their corresponding models, but the thread view model extracts the name of the forum which contains the thread, and the post view model does likewise for the containing thread. You may find Visual Studio is unsure whether the Thread class is to come from the QAForum.Models or System.Threading namespaces. If this does occur then configure the code as necessary. Then, add two new controllers, ThreadController and PostController. And add the relevant code and views so that these show the relevant data. Finally, modify the navigation bar to include links to the two new controllers, and test that everything works correctly.