Identity Modding in ASP.NET Core 2.1 and above

John Calder v02 2019-02-26 minor spelling corrections and check  
John Calder v01 2018-07-31 rewrite for major changes from 2.0 to 2.1

Based on:   
<https://docs.microsoft.com/en-us/aspnet/core/security/authentication/add-user-data?view=aspnetcore-2.1&tabs=visual-studio>

With added screen shots and text editing to adapt to our "MVCManukauTech" standard.

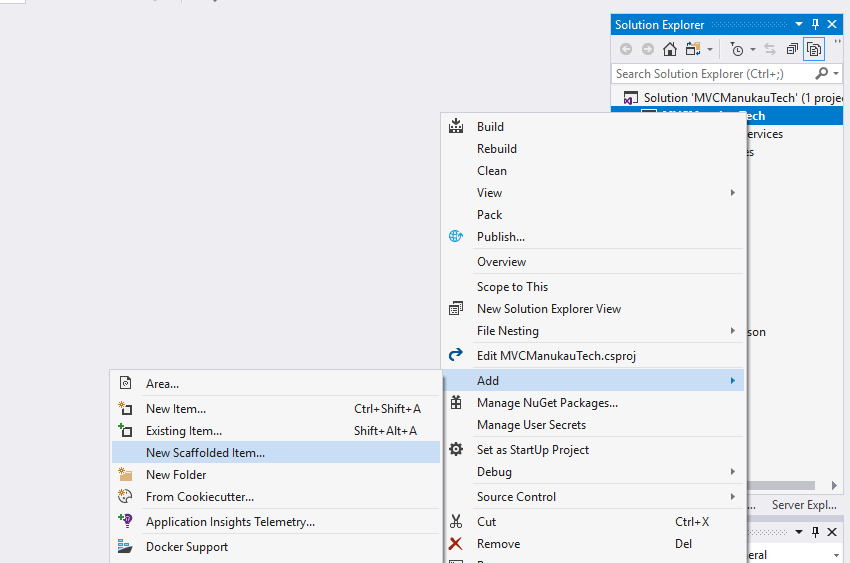
John Calder's interpretation of what has changed

We look at a 2.1 version app and the Identity files that we had in 2.0 and below have gone.  
This is because the Identity code now exists as compiled into "intermediate language" in a file probably under folder "bin".  
We cannot start editing "intermediate language"  
But Microsoft do offer the option of bringing back some or all of Identity as source code so we can edit it, in a similar manner to how we did it before.

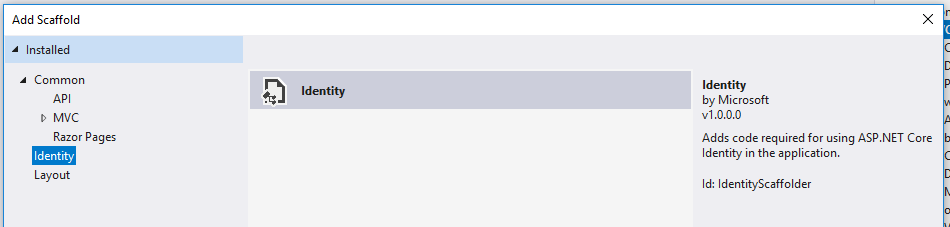
Start with an existing or new "ASP.NET MVC" app for ASP.NET Core 2.1 or above.

Run the Identity Scaffolder

From Solution Explorer, right-click on the project > Add > New Scaffolded Item.

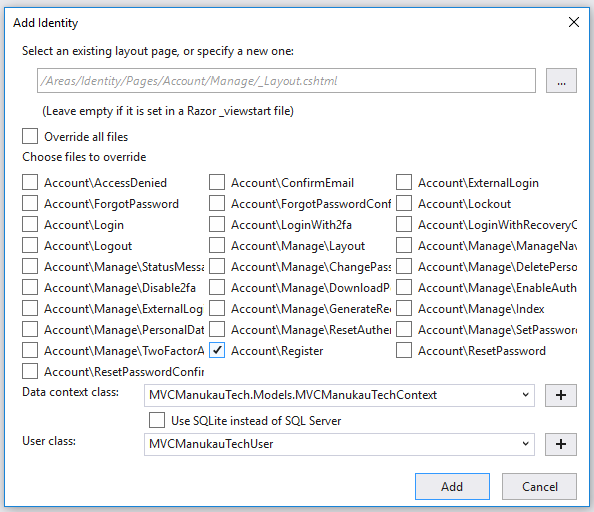


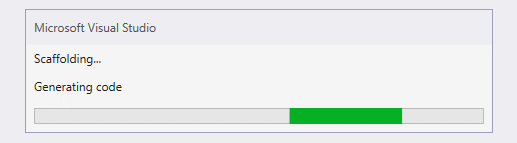
From the left pane of the Add Scaffold dialog, select Identity > ADD



In the **ADD Identity** dialog, the following options:

* Stay with the existing layout file *~/Pages/Shared/\_Layout.cshtml*
* Select the following files to override:
  + **Account/Register**
* Select the **+** button to create a new **Data context class**. Accept the type **MVCManukauTech.Models.MVCManukauTechContext**   
  JPC Comment - this is another good reason to have standard namespaces for our Manukau community.
* Select the **+** button to create a new **User class**. Accept the type **ManukauTechUser**
* Click the "Add" button in the bottom right of the dialog.





A help file appears in Visual Studio - "ScaffoldingReadme.txt".

This page selects and explains the information in there that is relevant to us.  
  
-----------------------------------------------------------

"Configuration of the Identity related services can be found in the Areas/Identity/IdentityHostingStartup.cs file.

If your app was previously configured to use Identity, then you should remove the call to the AddIdentity method from your ConfigureServices method."

Find file "startup.cs" in the root of the application.

Delete the code there for setup of Identity because the new file is taking over that job.

Comment-out this for testing, then Delete after successful testing

|  |
| --- |
| services.AddDbContext<ApplicationDbContext>(options =>  options.UseSqlServer(  Configuration.GetConnectionString("DefaultConnection")));  services.AddDefaultIdentity<IdentityUser>()  .AddEntityFrameworkStores<ApplicationDbContext>(); |

Now look at the new file: Areas/Identity/IdentityHostingStartup.cs

The "scaffolder" (code generator) makes what is for our case an error with ConnectionString  
We are using DefaultConnection for our Identity connection as well as our general connection.  
We therefore need to change " "MVCManukauTechContextConnection"" to "DefaultConnection"

Before:

|  |
| --- |
| context.Configuration.GetConnectionString("MVCManukauTechContextConnection"))); |

After:

|  |
| --- |
| context.Configuration.GetConnectionString("DefaultConnection"))); |

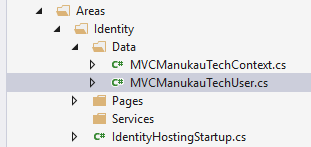
There is a related cleanup to do in the file appsettings.json

The scaffolder creates a new connection string - "MVCManukauTechContextConnection"   
Delete this.  
  
The scaffolder is designed around the idea of having separate databases for Identity vs General Business

We do not do that. Our workshop standard is that it is better to have one database for an app.

Add custom user data to the Identity DB

Update the IdentityUser derived class with custom properties.  
If you named your project MVCManukauTech, the file is named Areas/Identity/Data/MVCManukauTechUser.cs.

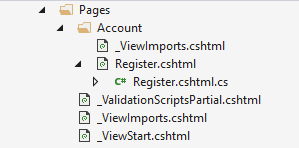


Delete the contents of this file and replace with the following code -  
- we can then edit the [PersonalData] fields to meet our needs for the case we are working on:

|  |
| --- |
| using Microsoft.AspNetCore.Identity;  using System;  namespace MVCManukauTech.Areas.Identity.Data  {  public class MVCManukauTechUser : IdentityUser  {  [PersonalData]  public double Height { get; set; }  [PersonalData]  public DateTime Birthdate { get; set; }  }  } |

Update the Account/Register.cshtml page

Update the InputModel in Areas/Identity/Pages/Account/Register.cshtml.cs with the following highlighted code:  
Register.cshtml.cs is a child of Register.cshtml and you may need to click on the expansion arrow to the left of the parent file to see it.



Modify and/or add these line.

Check that this has been automatically added to the using statements at the top of the file. Else add manually.

|  |
| --- |
| using MVCManukauTech.Areas.Identity.Data; |

We need to add new code for each field to public class InputModel

|  |
| --- |
| public class InputModel  {  [Required]  [EmailAddress]  [Display(Name = "Email")]  public string Email { get; set; }  [Required]  [StringLength(100, ErrorMessage = "The {0} must be at least {2} and at max {1} characters long.", MinimumLength = 6)]  [DataType(DataType.Password)]  [Display(Name = "Password")]  public string Password { get; set; }  [DataType(DataType.Password)]  [Display(Name = "Confirm password")]  [Compare("Password", ErrorMessage = "The password and confirmation password do not match.")]  public string ConfirmPassword { get; set; }  //20180731 JPC. Attributes in square brackets are optional.  // They work as a short code way of providing validation and labelling  [Required]  [Display(Name = "Height")]  public double Height { get; set; }  //20180731 JPC Note need for DataType.Date rather than DataType.DateTime  // This is because DataType.DateTime forces data entry of date and time.  // We only want Date  [Required]  [Display(Name = "Birth Date")]  [DataType(DataType.Date)]  public DateTime Birthdate { get; set; }  } |

Look or search further down for this code:

|  |
| --- |
| var user = new MVCManukauTechUser { UserName = Input.Email, Email = Input.Email } |

Modify with the extra fields to give a longer version (with more readable layout):

|  |
| --- |
| //20180731 JPC add additional fields Height and Birthdate  var user = new MVCManukauTechUser {  UserName = Input.Email,  Email = Input.Email,  Height = Input.Height,  Birthdate = Input.Birthdate  }; |

Then work on the parent UI page: Register.cshtml

JPC recommends copying and pasting the form-group for Input.Email to add the UI for other fields.  
Or we can copy and adapt the JPC new field code.

|  |
| --- |
| @\*20180731 JPC add new custom fields Height and Birthdate\*@  <div class="form-group">  <**label** **asp-for**="Input.Height"></**label**>  <**input** **asp-for**="Input.Height" class="form-control" />  <**span** **asp-validation-for**="Input.Height" class="text-danger"></**span**>  </div>  <div class="form-group">  <**label** **asp-for**="Input.Birthdate"></**label**>  <**input** **asp-for**="Input.Birthdate" class="form-control" />  <**span** **asp-validation-for**="Input.Birthdate" class="text-danger"></**span**>  </div> |

In this work we have replaced the default class IdentityUser with our custom class MVCManukauTechUser

We have one file where IdentityUser is still active and we need to change that.

The file is

Views\Shared\\_LoginPartial.cshtml

Before:

|  |
| --- |
| @using Microsoft.AspNetCore.Identity  @inject SignInManager<IdentityUser> SignInManager  @inject UserManager<IdentityUser> UserManager |

After:

|  |
| --- |
| @using Microsoft.AspNetCore.Identity  @using MVCManukauTech.Areas.Identity.Data;  @inject SignInManager<MVCManukauTechUser> SignInManager  @inject UserManager<MVCManukauTechUser> UserManager |

END of Document