



# NETCore3 Identity Solution #1

**Base Folder**

**L:\Samples\ASPNET-Identity\**

**Source Folder**

**~\Core3IdentitySolution1**

**Git-Archives**

**~\Git-Archives\Core3IdentitySolution1\**

**Database Backups**

**~\DBBackups\Core3IdentitySolution1**

**Generated 11/02/2019 10:55**



# **NETCore3 Identity Solution #1**

## **Documentation & URLs**

**[https://stackoverflow.com/questions/58584446/asp-net-core-identity-usermanager-isinrole-call-works-in-2-2-but-throws-invalido?noredirect=1#comment103484000\\_58584446](https://stackoverflow.com/questions/58584446/asp-net-core-identity-usermanager-isinrole-call-works-in-2-2-but-throws-invalido?noredirect=1#comment103484000_58584446)**



# NETCore3 Identity Solution #1



# **NETCore3 Identity Solution #1**

**Starting GitId: NA**

**No commit**

**Create Core3IdentitySolution1 VS2019 Solution Next**

**Base Folder**

**L:\Samples\ASPNET-Identity\**

**Source Folder**

**~\Core3IdentitySolution1**

**Git-Archives**









**~\Git-Archives\Core3IdentitySolution1\  
None**

**11/02/2019 10:55**

**Page 4**

# Create a new project

## Recent project templates

-  ASP.NET Web Application (.NET Framework) C#
-  ASP.NET Core Web Application C#
-  Blank Solution
-  SQL Server Database Project Query Language
-  Console App (.NET Framework) C#
-  Class Library (.NET Framework) C#
-  Console App (.NET Core) Visual Basic
-  Blank Solution

Blank Solution

Clear

All Languages

All Platforms

All Project Types



### Blank Solution

Create an empty solution containing no projects

Other



### Blank App (Universal Windows)

A project for a single-page Universal Windows Platform (UWP) app that has no predefined controls or layout.

C# Windows Xbox UWP Desktop



### Blank App (Universal Windows)

A project for a single-page Universal Windows Platform (UWP) app that has no predefined controls or layout.

Visual Basic Windows Xbox UWP Desktop



### Blank Node.js Console Application

An empty Node.js application.

JavaScript Windows Linux macOS Console



### Blank Node.js Web Application

An empty Node.js Web application.

JavaScript Windows Linux macOS Web



### Blank Node.js Console Application

Next

11/02/2019 11:07 - Create blank VS2019 Solution

# Configure your new project


Blank Solution ☒ Other

Project name

Core3IdentitySolution1

Location

L:\Samples\ASPNET-Identity\

Solution name 

Core3IdentitySolution1

Back

Create

11/02/2019 11:07 - Create L:\Samples\ASPNET-Identity\Core3IdentitySolution1

# Add a new project

## Recent project templates

- ASP.NET Web Application (.NET Framework) C#
- ASP.NET Core Web Application C#
- SQL Server Database Project Query Language
- Console App (.NET Framework) C#
- Class Library (.NET Framework) C#
- Console App (.NET Core) Visual Basic

Class Library

Clear

All Languages

All Platforms

All Project Types



### Class Library (For U-SQL Application)

A project for creating a C# class library(.dll) that can run on U-SQL.



### Class Library (.NET Framework)

A project for creating a C# class library(.dll)

C#

Windows

Library



### Class Library (.NET Core)

A project for creating a class library that targets .NET Core.

C#

Windows

Linux

macOS

Library



### Class Library (.NET Framework)

A project for creating a VB class library(.dll)

Visual Basic

Windows

Library



### Class Library (.NET Core)

A project for creating a class library that targets .NET Core.

Visual Basic

Windows

Linux

macOS

Library



### Class Library (.NET Standard)

A project for creating a class library that targets .NET Standard.

F#

Android

iOS

Linux

macOS

Windows

Library

Next

11/02/2019 11:16 – Add 2 .NET Core 3.0 Class Libraries

# Configure your new project

Class Library (.NET Core)

C#

Windows

Linux

macOS

Library

Project name

Core3ID.Domain

Location

L:\Samples\ASPNET-Identity\Core3IdentitySolution1

...

Back

Create

11/02/2019 11:19 – Configure ~\Core3IdentitySolution1\Core3.Domain



# Configure your new project

Class Library (.NET Core)

C#

Windows

Linux

macOS

Library

Project name

Core3Identity.DataAccess

Location

L:\Samples\ASPNET-Identity\Core3IdentitySolution1


Back


Create


11/02/2019 11:29 – Configure ~\Core3IdentitySolution1\Core3.DataAccess

# Add a new project

## Recent project templates


 ASP.NET Web Application (.NET Framework) C#

 ASP.NET Core Web Application C#

 SQL Server Database Project Query Language

 Console App (.NET Framework) C#

 Class Library (.NET Framework) C#

 Console App (.NET Core) Visual Basic

Search for templates (Alt+S)

All Languages

All Platforms

All Project Types



Console App (.NET Core)

A project for creating a command-line application that can run on .NET Core on Windows, Linux and MacOS.

C# Linux macOS Windows Console



Console App (.NET Core)

A project for creating a command-line application that can run on .NET Core on Windows, Linux and MacOS.

Visual Basic Windows Linux macOS Console



ASP.NET Core Web Application

Project templates for creating ASP.NET Core web apps and web APIs for Windows, Linux and macOS using .NET Core or .NET Framework. Create web apps with Razor Pages, MVC, or Single Page Apps (SPA) using Angular, React, or React + Redux.

C# Linux macOS Windows Cloud Service Web



Blazor App

Project templates for creating Blazor apps that that run on the server in an ASP.NET Core app or in the browser on WebAssembly. These templates can be used to build web apps with rich dynamic user interfaces (UIs).

C# Linux macOS Windows Cloud Web



ASP.NET Web Application (.NET Framework)

Next

11/02/2019 12:22 – Add .NET Core 3.0 Web Application

# Configure your new project

ASP.NET Core Web Application

C#

Linux

macOS

Windows

Cloud

Service

Web

Project name

Core3Identity.WebUI

Location

L:\Samples\ASPNET-Identity\Core3IdentitySolution1

...

Back

Create

11/02/2019 12:24 – Configure Configure ~\Core3IdentitySolution1\Core3.WebUI

# Create a new ASP.NET Core web application

.NET Core ASP.NET Core 3.0



## Empty

An empty project template for creating an ASP.NET Core application. This template does not have any content in it.



## API

A project template for creating an ASP.NET Core application with an example Controller for a RESTful HTTP service. This template can also be used for ASP.NET Core MVC Views and Controllers.



## Web Application

A project template for creating an ASP.NET Core application with example ASP.NET Razor Pages content.



## Web Application (Model-View-Controller)

A project template for creating an ASP.NET Core application with example ASP.NET Core MVC Views and Controllers. This template can also be used for RESTful HTTP services.



## Angular

A project template for creating an ASP.NET Core application with Angular



## React.js

A project template for creating an ASP.NET Core application with React.js

[Get additional project templates](#)

## Authentication

No Authentication

[Change](#)

## Advanced

☒ [Configure for HTTPS](#)

☐ [Enable Docker Support](#)  
(Requires [Docker Desktop](#))

Linux

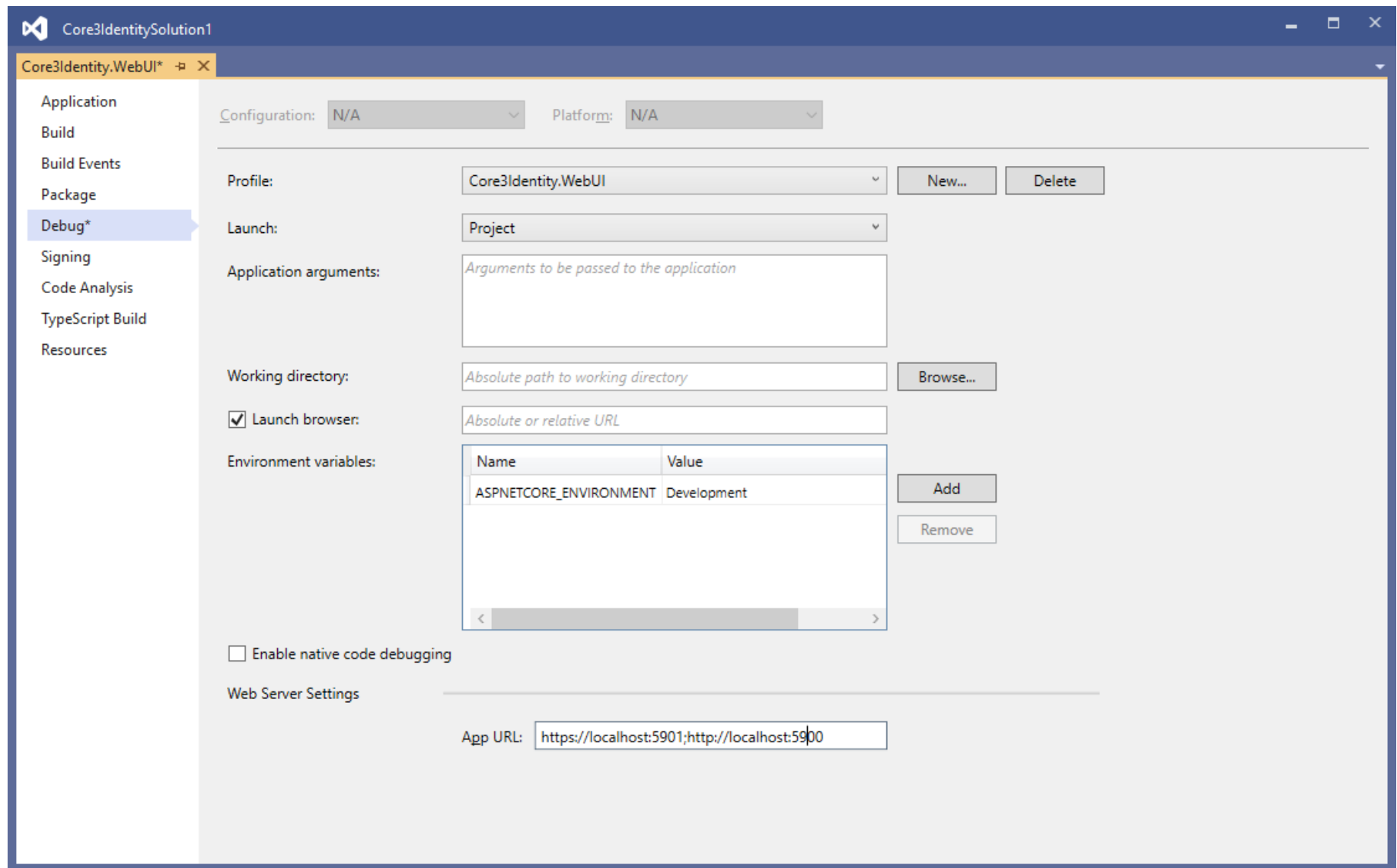
Author: Microsoft

Source: .NET Core 3.0.0

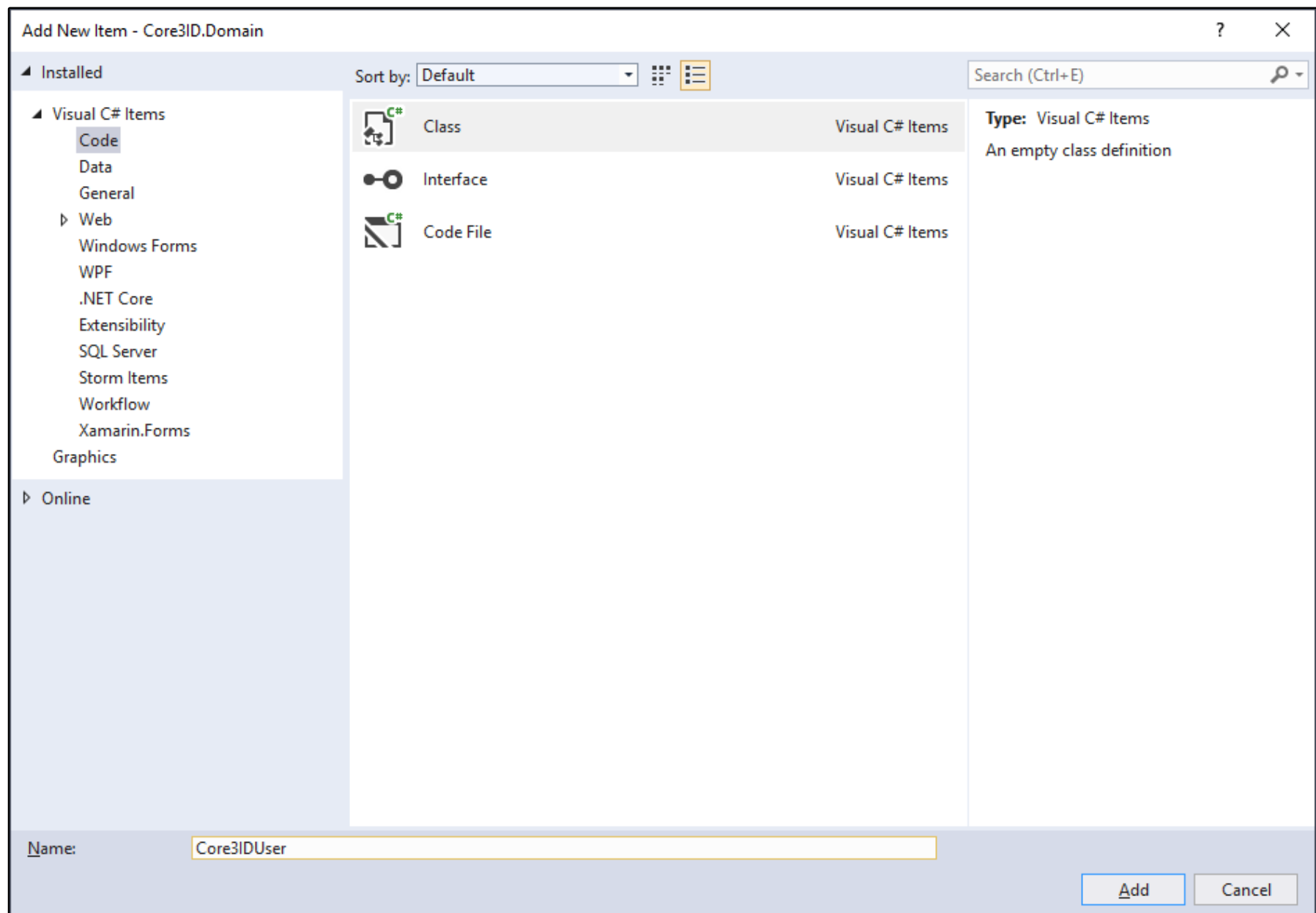
[Back](#)

Create

11/02/2019 12:27 - Create ~\Core3IdentitySolution1\Core3.WebUI Razor Page Application



11/02/2019 12:31 – Set Core3IdentitySolution1\Core3.WebUI Debug Properties  
Run on localhost ports, 5900 & 5901 in Kestrel



11/02/2019 15:34 - Create ~\Core3IdentitySolution1\Core3.Domain\Core3IDUser.cs

```
using System.ComponentModel.DataAnnotations;
using Microsoft.AspNetCore.Identity;

namespace Core3ID.Domain
{
    /// <summary>
    /// Core 3.0 Identity User
    /// </summary>
    /// <remarks>
    /// This class inherits from IdentityUser and include customized data for all authenticated users.
    /// An instance of this class is created upon registration.
    /// </remarks>
    public class Core3IDUser : IdentityUser
    {
        /// <summary>
        /// User's Last Name
        /// </summary>
        [PersonalData]
        public string LastName { get; set; }

        /// <summary>
        /// User's First Name
        /// </summary>
        [PersonalData]
        public string FirstName { get; set; }
    }
}
```

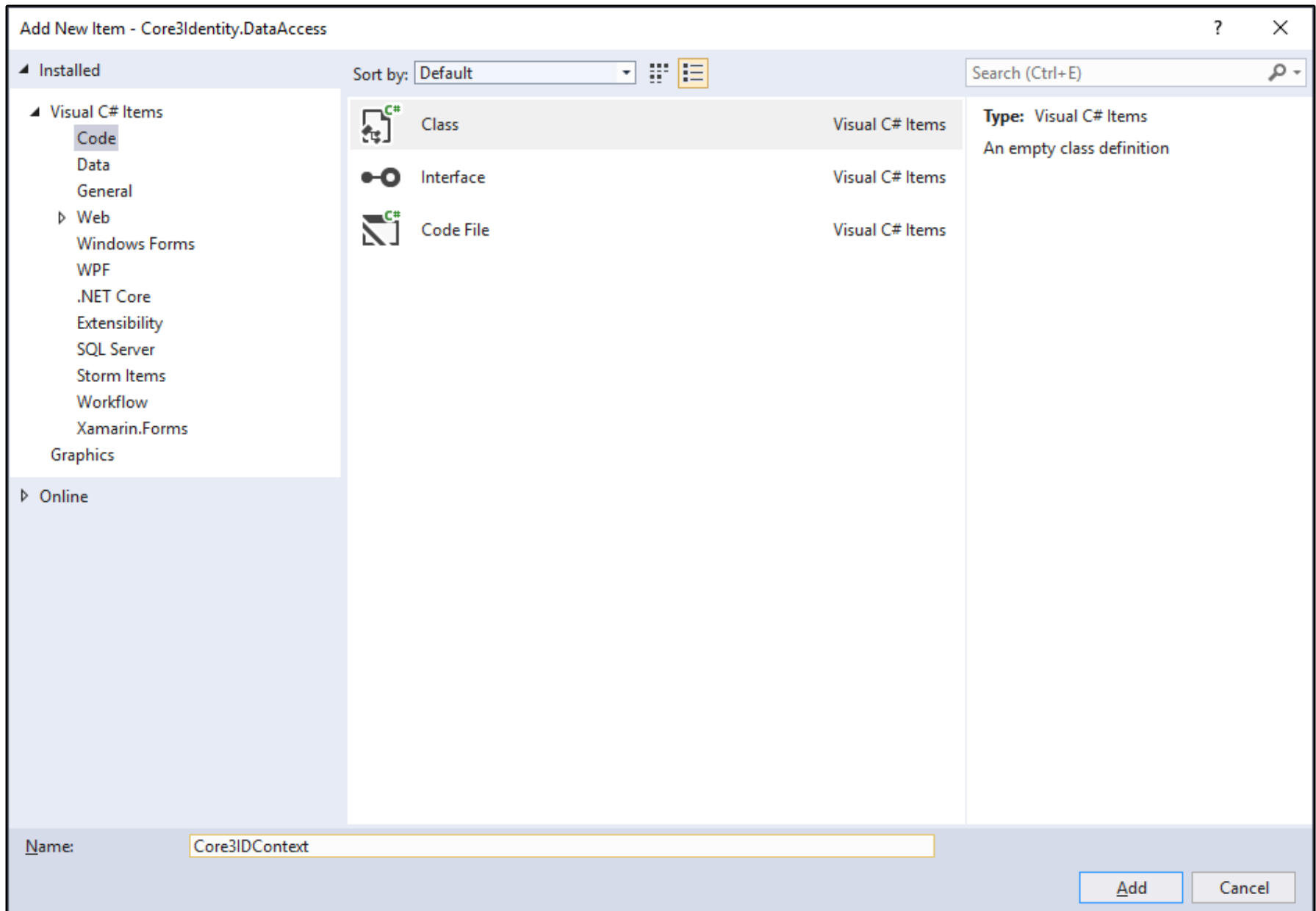
**. . . Continued on next page**

**. . . Continued from last page**

```
    /// <summary>
    /// Phone Number
    /// </summary>
    /// <remarks>
    /// This is override of IdentityUser.PhoneNumber. Ideally, it should be
    /// used in any Two-Factor-Authentication, if used.
    /// </remarks>
    [PersonalData]
    [DataType(DataType.PhoneNumber)]
    public new string PhoneNumber { get; set; }
} // end public class Core3IDUser : IdentityUser
} // end namespace Core3ID.Domain
```

**11/02/2019 15:41 - Initial ~\Core3IdentitySolution1\Core3.Domain\Core3IDUser.cs, p. 2**





11/02/2019 15:46 - Create ~\Core3IdentitySolution1\Core3.DataAccess\Core3IDContext.cs

Core3IdentitySolution1 - NuGet: Core3Identity.DataAccess

NuGet: Core3Identity.DataAccess

Browse Installed Updates

Microsoft.EntityFrameworkCore.SqlServer x Include prerelease Package source: nuget.org

**Microsoft.EntityFrameworkCore.SqlServer** by Microsoft, 39 v3.0.0 ↓  
Microsoft SQL Server database provider for Entity Framework Core.

**Microsoft.EntityFrameworkCore** by Microsoft, 58.7M download v3.0.0  
Entity Framework Core is a lightweight and extensible version of the popular Entity Framework data access technology.

**Microsoft.EntityFrameworkCore.Relational** by Microsoft, 51 v3.0.0  
Shared Entity Framework Core components for relational database providers.

**Microsoft.EntityFrameworkCore.Analyzers** by Microsoft, 36 v3.0.0  
CSharp Analyzers for Entity Framework Core.

**Microsoft.EntityFrameworkCore.Design** by Microsoft, 37.8M v3.0.0  
Shared design-time components for Entity Framework Core tools.

**Microsoft.EntityFrameworkCore.Tools** by Microsoft, 31.7M v3.0.0  
Entity Framework Core Tools for the NuGet Package Manager Console in Visual Studio.

**Microsoft.EntityFrameworkCore.Abstractions** by Microsoft v3.0.0

**Microsoft.EntityFrameworkCore.SqlServer** nuget.org

Version: Latest stable 3.0.0 Install

Options

Description

Microsoft SQL Server database provider for Entity Framework Core.

Version: 3.0.0

Author(s): Microsoft

License: Apache-2.0

Date published: Monday, September 23, 2019 (9/23/2019)

Project URL: <https://docs.microsoft.com/ef/core/>

Report Abuse: <https://www.nuget.org/packages/Microsoft.EntityFrameworkCore.SqlServer/3.0.0/ReportAbuse>

Tags: EF, Core, Framework, EntityFrameworkCore, O/RM, Data, entity-framework-core, EntityFramework SQL Server

11/02/2019 15:55 – Add NuGet pkg, Microsoft.EntityFrameworkCore.SqlServer 3.0.0 to Core3Identity.DataAccess

New Database

Select a page

General

Options

Filegroups

Connection

Server:  
(localdb)\mssqllocaldb

Connection:  
SIMBA\Venj

View connection properties

Progress

Ready

Script

Help

Database name:  
Core3IDDB

Owner:  
<default>

☒ Use full-text indexing

Database files:

Logical Name	File Type	Filegroup	Initial Size (MB)	Autogrowth / Maxsize	Path	File Name
Core3IDDB	ROWS Data	PRIMARY	8	By 64 MB, Unlimited	C:\Users\Venj\	
Core3IDDB_log	LOG	Not Applicable	8	By 64 MB, Unlimited	C:\Users\Venj\	

Add

Remove

OK

Cancel

New Database

Select a page

General

Options

Filegroups

Connection

Server:  
SIMBA\MSSQLSERVER17

Connection:  
SIMBA\Venj

View connection properties

Progress

Ready

Script

Help

Database name:  
Core3IDDB

Owner:  
<default>

☒ Use full-text indexing

Database files:

Logical Name	File Type	Filegroup	Initial Size (M...	Autogrowth / Maxsize	Path	File Name
Core3IDDB	ROWS Data	PRIMARY	8	By 64 MB, Unlimited	L:\Samples\ASPNET-Identity\Data\	
Core3IDDB_...	LOG	Not Applicable	8	By 64 MB, Unlimited	L:\Samples\ASPNET-Identity\Data\	

Add

Remove

OK

Cancel

11/02/2019 16:22 – SSMS: Create empty database, SIMBA\MSSQLSERVER17\Core3IDDB

Add Connection ? X

Enter information to connect to the selected data source or click "Change" to choose a different data source and/or provider.

Data source:  
Microsoft SQL Server (SqlClient) Change...

Server name:  
SIMBA\MSSQLSERVER17 Refresh

Log on to the server

Authentication: Windows Authentication

User name:

Password:

☐ Save my password

Connect to a database

☒ Select or enter a database name:  
Core3IDDB

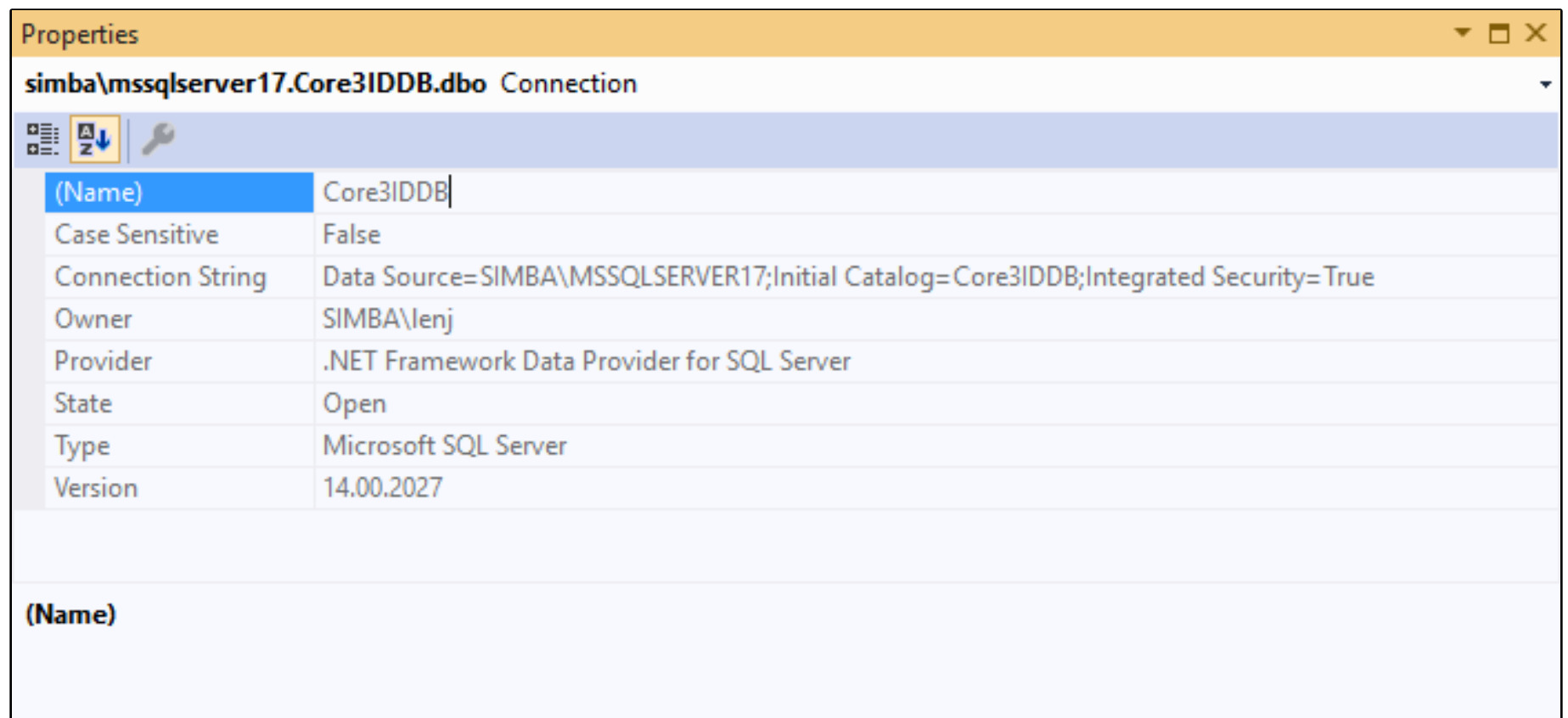
☐ Attach a database file:  
 Browse...

Logical name:

Advanced...

Test Connection OK Cancel

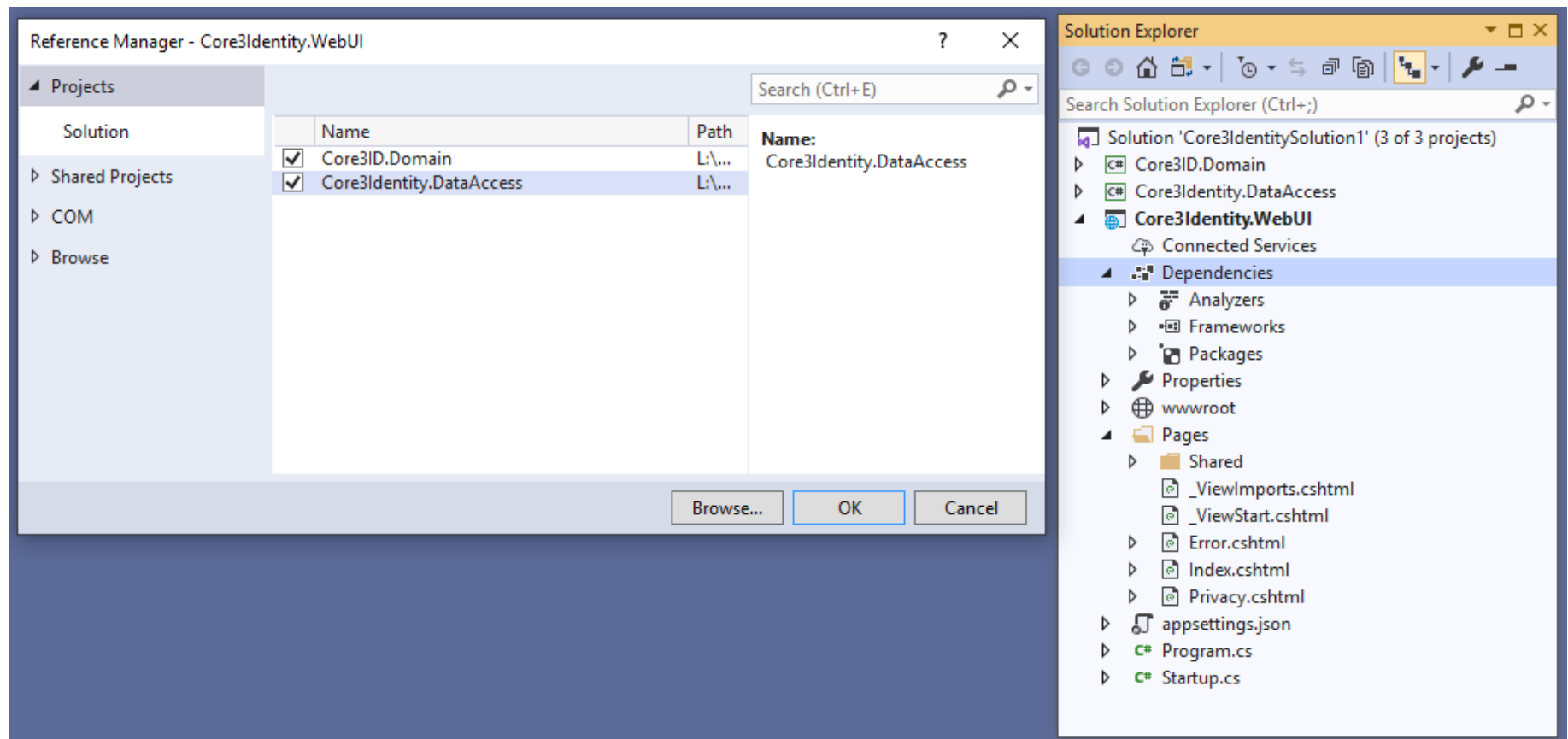
11/02/2019 16:22 – Server Explorer: Create connection to Core3IDDB



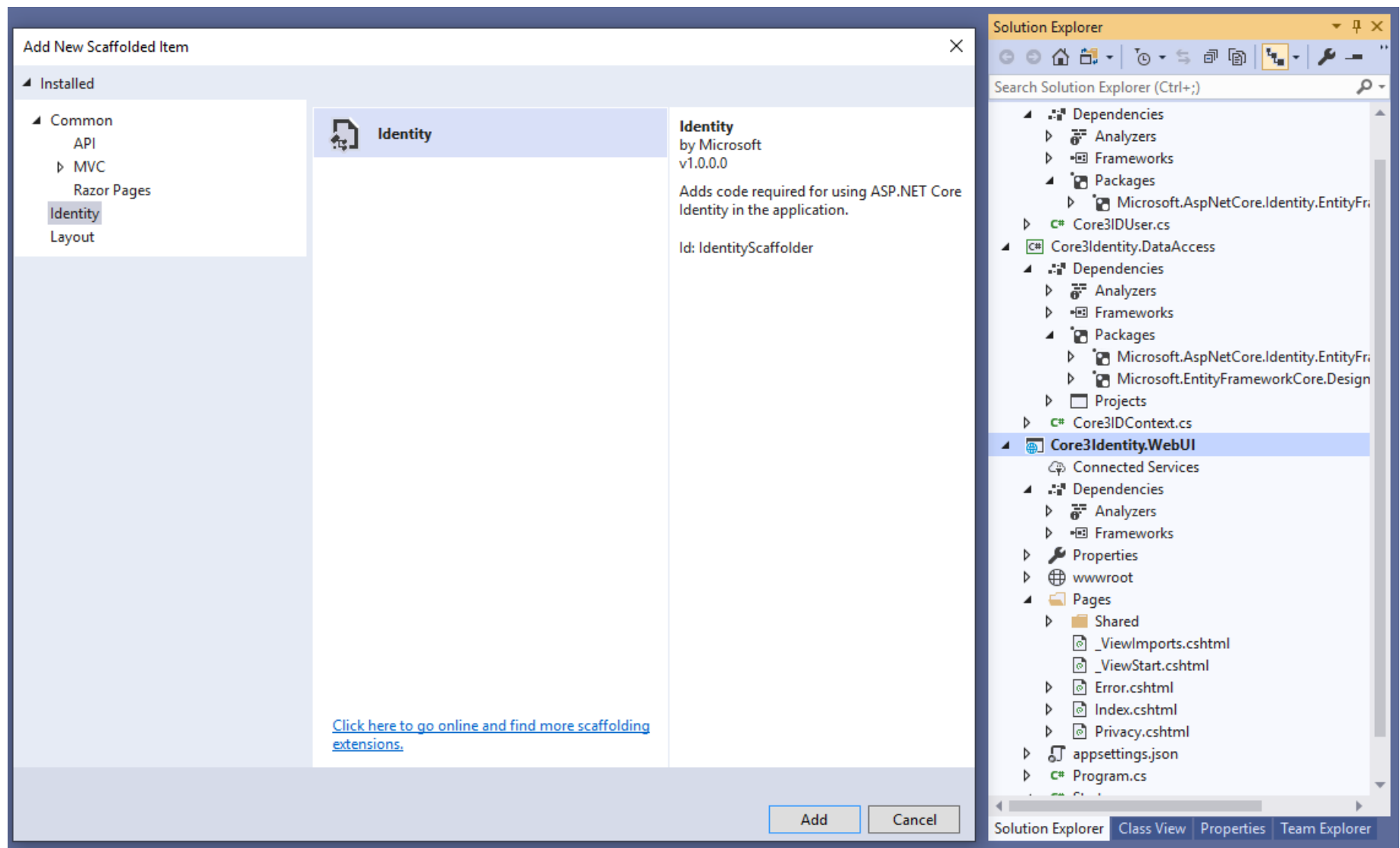
11/02/2019 16:22 – Server Explorer Connection Properties for Core3IDDB

```
{
  "Logging": {
    "LogLevel": {
      "Default": "Information",
      "Microsoft": "Warning",
      "Microsoft.Hosting.Lifetime": "Information"
    }
  },
  "AllowedHosts": "*",
  "ConnectionStrings": {
    "DefaultConnection": "Data Source=SIMBA\\MSSQLSERVER17;Initial Catalog=Core3IDDB;Integrated Security=True"
  }
}
```

**11/02/2019 16:32 – Modified ~\Core3IdentitySolution1\Core3.WebUI\appsettings.json**

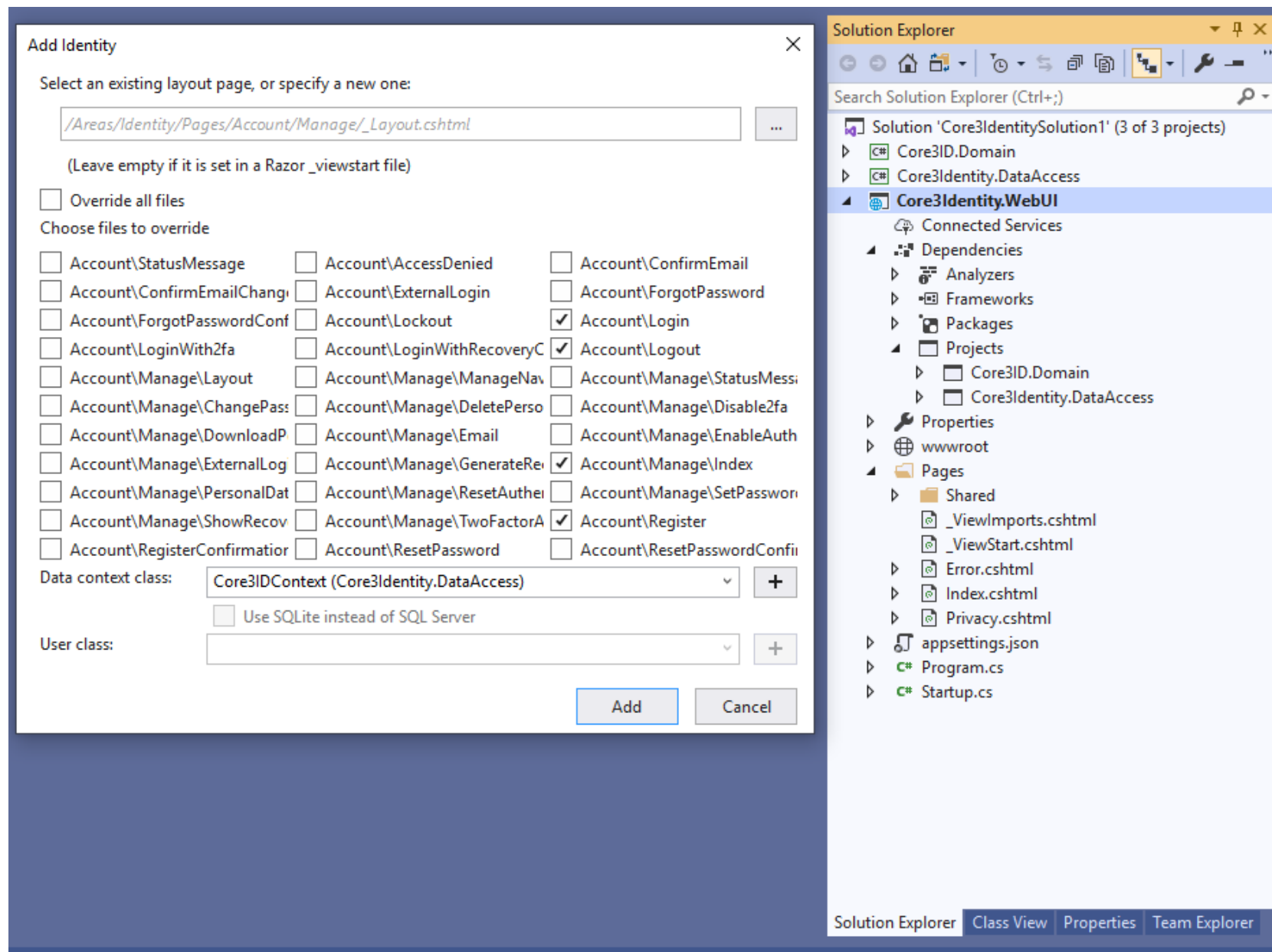


11/02/2019 16:45 – Set Core3Identity.WebUI references to Core3ID.Domain & Core3Identity.DataAccess

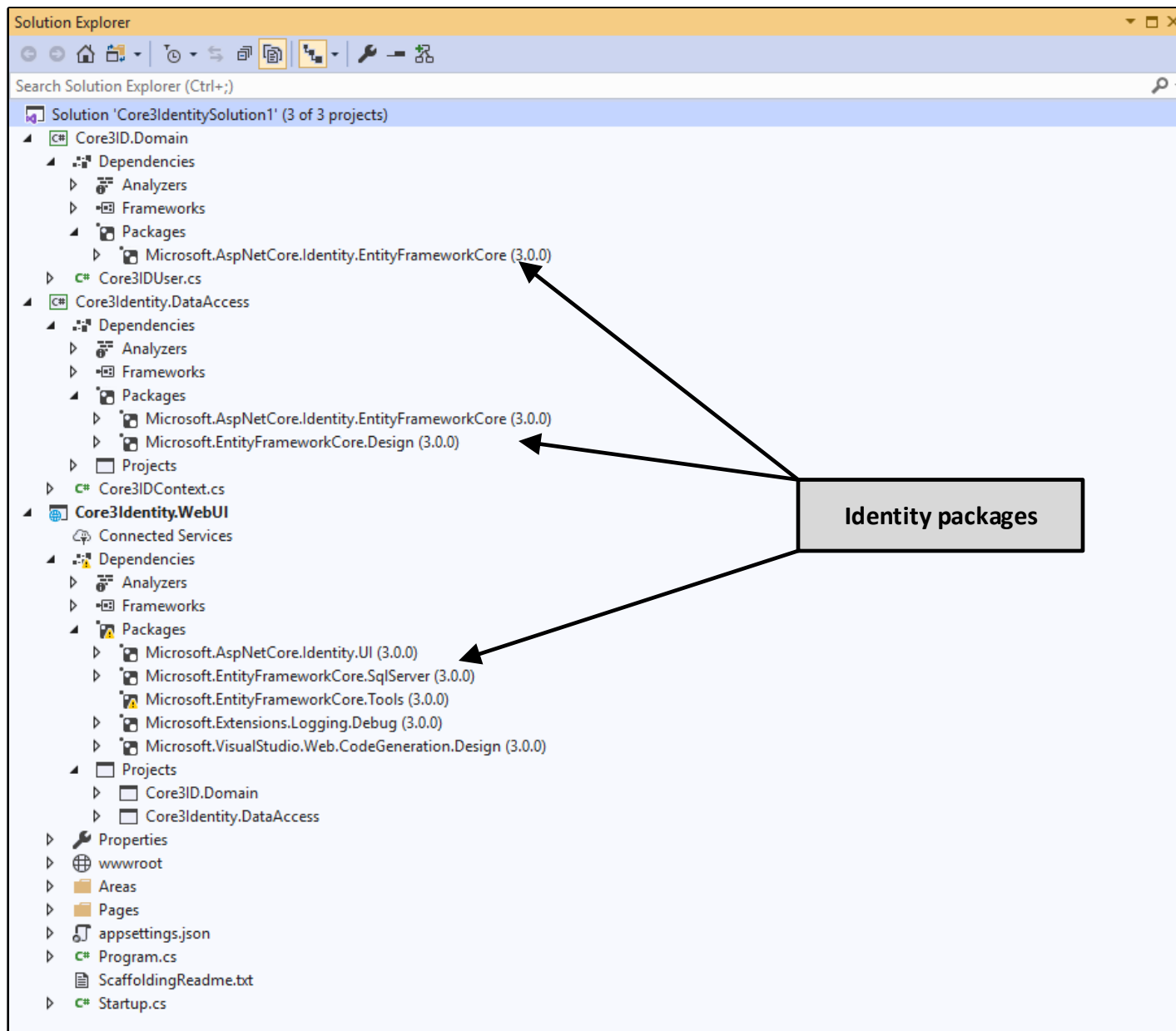


11/02/2019 16:47 – Scaffold Identity for ~\Core3IdentitySolution1\Core3Identity.WebUI





11/02/2019 16:54 – Scaffold Identity Area for ~\Core3IdentitySolution1\Core3Identity.WebUI



11/02/2019 17:00 – Core3IdentitySolution1 project packages

```

<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="utf-8" />
  <meta name="viewport" content="width=device-width, initial-scale=1.0" />
  <title>@ViewData["Title"] - Core3Identity.WebUI</title>
  <link rel="stylesheet" href="~/lib/bootstrap/dist/css/bootstrap.min.css" />
  <link rel="stylesheet" href="~/css/site.css" />
</head>
<body>
  <header>
    <nav class="navbar navbar-expand-sm navbar-toggleable-sm navbar-light bg-white border-bottom box-shadow mb-3">
      <div class="container">
        <a class="navbar-brand" asp-area="" asp-page="/Index">Core 3.0 Identity</a>
        <button class="navbar-toggler" type="button" data-toggle="collapse" data-target=".navbar-collapse"
          aria-controls="navbarSupportedContent" aria-expanded="false" aria-label="Toggle navigation">
          <span class="navbar-toggler-icon"></span>
        </button>
        <div class="navbar-collapse collapse d-sm-inline-flex flex-sm-row" style="display: flex;
          justify-content: space-around">
          <div class="navbar">
            <a class="nav-link text-dark" asp-area="" asp-page="/Index">Home</a>
            <a class="nav-link text-dark" asp-area="" asp-page="/RoleAdmin/Index">RoleAdmin</a>
          </div>
          <partial name="_LoginPartial" />
        </div>
      </div>
    </nav>
  </header>
  <div class="container">
    <main role="main" class="pb-3">
      @RenderBody()
    </main>
  </div>

```

. . . Continued on next page

**. . . Continued from last page**

```
<footer class="border-top footer text-muted">
  <div class="container">
    &copy; 2019 - Core3Identity.WebUI - <a asp-area="" asp-page="/Privacy">Privacy</a>
  </div>
</footer>

<script src="~/lib/jquery/dist/jquery.min.js"></script>
<script src="~/lib/bootstrap/dist/js/bootstrap.bundle.min.js"></script>
<script src="~/js/site.js" asp-append-version="true"></script>

@RenderSection("Scripts", required: false)
</body>
</html>
```

**11/02/2019 17:26 – Modified ~\Core3Identity.WebUI\Pages\Shared\\_Layout.cshtml, p. 2**

```

@using Microsoft.AspNetCore.Identity
@using Core3ID.Domain

@inject SignInManager<Core3IDUser> SignInManager
@inject UserManager<Core3IDUser> UserManager

<div class="navbar-nav" style="display: flex; justify-content: flex-end">
    @if (SignInManager.IsSignedIn(User))
    {
        <a id="manage" class="nav-link text-dark" asp-area="Identity" asp-page="/Account/Manage/Index" title="Manage">
            Hello @UserManager.GetUserName(User)!
        </a>
        <form id="logoutForm" class="form-inline" asp-area="Identity" asp-page="/Account/Logout"
            asp-route-returnUrl="@Url.Page("/Index", new { area = "" })">
            <button id="logout" type="submit" class="nav-link btn btn-link text-dark">Logout</button>
        </form>
    }
    else
    {
        <a class="nav-link text-dark" id="register" asp-area="Identity" asp-page="/Account/Register">Register</a>
        <a class="nav-link text-dark" id="login" asp-area="Identity" asp-page="/Account/Login">Login</a>
    }
</div>

```

11/02/2019 17:29 – Modified ~\Core3Identity.WebUI\Pages\Shared\\_LoginPartial.cshtml

```

using System.Collections.Generic;
using System.ComponentModel.DataAnnotations;
using System.Linq;
using System.Text;
using System.Text.Encodings.Web;
using System.Threading.Tasks;
using Microsoft.AspNetCore.Authentication;
using Microsoft.AspNetCore.Authorization;
using Core3ID.Domain;
using Microsoft.AspNetCore.Identity;
using Microsoft.AspNetCore.Identity.UI.Services;
using Microsoft.AspNetCore.Mvc;
using Microsoft.AspNetCore.Mvc.RazorPages;
using Microsoft.AspNetCore.WebUtilities;
using Microsoft.Extensions.Logging;

namespace Core3Identity.WebUI.Areas.Identity.Pages.Account
{
    [AllowAnonymous]
    public class RegisterModel : PageModel
    {
        private readonly SignInManager<Core3IDUser> _signInManager;
        private readonly UserManager<Core3IDUser> _userManager;
        private readonly ILogger<RegisterModel> _logger;
        private readonly IEmailSender _emailSender;

        public RegisterModel(
            UserManager<Core3IDUser> userManager,
            SignInManager<Core3IDUser> signInManager,
            ILogger<RegisterModel> logger,
            IEmailSender emailSender)
        {

```

**. . . Continued on next page**

**. . . Continued from last page**

```
_userManager = userManager;
_signInManager = signInManager;
_logger = logger;
_emailSender = emailSender;
} // end public RegisterModel(...)

[BindProperty]
public InputModel Input { get; set; }

public string returnUrl { get; set; }

public IList<AuthenticationScheme> ExternalLogins { get; set; }

public class InputModel
{
    /// <summary>
    /// User's Last Name
    /// </summary>
    [Required]
    [Display(Name = "Last Name")]
    public string LastName { get; set; }

    /// <summary>
    /// User's First Name
    /// </summary>
    [Required]
    [Display(Name = "First Name")]
    public string FirstName { get; set; }

    /// <summary>
    /// User's Phone Number
    /// </summary>
    [Required]
    [Display(Name = "Phone Number")]
    public string PhoneNumber { get; set; }
```

**. . . Continued on next page**

**. . . Continued from last page**

```
[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; }
} // end public class InputModel

public async Task OnGetAsync(string returnUrl = null)
{
    ReturnUrl = returnUrl;
    ExternalLogins = (await _signInManager.GetExternalAuthenticationSchemesAsync()).ToList();
} // end public async Task OnGetAsync(string returnUrl = null)

public async Task<IActionResult> OnPostAsync(string returnUrl = null)
{
    returnUrl = returnUrl ?? Url.Content("~/");
    ExternalLogins = (await _signInManager.GetExternalAuthenticationSchemesAsync()).ToList();
    if (ModelState.IsValid)
    {
        var user = new Core3IDUser { UserName = Input.Email, Email = Input.Email, LastName = Input.LastName,
            FirstName = Input.FirstName, PhoneNumber = Input.PhoneNumber };
        var result = await _userManager.CreateAsync(user, Input.Password);
        if (result.Succeeded)
        {
```

**. . . Continued on next page**



**. . . Continued from last page**

```
_logger.LogInformation("User created a new account with password.");

var code = await _userManager.GenerateEmailConfirmationTokenAsync(user);
code = WebEncoders.Base64UrlEncode(Encoding.UTF8.GetBytes(code));
var callbackUrl = Url.Page(
    "/Account/ConfirmEmail",
    pageHandler: null,
    values: new { area = "Identity", userId = user.Id, code = code },
    protocol: Request.Scheme);

await _emailSender.SendEmailAsync(Input.Email, "Confirm your email",
    $"Please confirm your account by <a href='{HtmlEncoder.Default.Encode(callbackUrl)}'>
        clicking here
    </a>.");

if (_userManager.Options.SignIn.RequireConfirmedAccount)
{
    return RedirectToPage("RegisterConfirmation", new { email = Input.Email });
}
else
{
    await _signInManager.SignInAsync(user, isPersistent: false);
    return LocalRedirect(returnUrl);
} // endif (_userManager.Options.SignIn.RequireConfirmedAccount)
} // endif (result.Succeeded)
foreach (var error in result.Errors)
{
    ModelState.AddModelError(string.Empty, error.Description);
} // end foreach (var error in result.Errors)
} // endif (ModelState.IsValid)
```

**. . . Continued on next page**

**. . . Continued from last page**

```
        // If we got this far, something failed, redisplay form
        return Page();
    } // end public async Task<IActionResult> OnPostAsync(string returnUrl = null)
} // end public class RegisterModel : PageModel
} // end namespace Core3Identity.WebUI.Areas.Identity.Pages.Account
```

**11/02/2019 18:15 – Modified ~\Areas\Identity\Pages\Account\Register.cshtml.cs, p. 5**

```

@page
@model RegisterModel
@{
    ViewData["Title"] = "Register";
}

<h1>@ViewData["Title"]</h1>

<div class="row">
    <div class="col-md-4">
        <form asp-route-returnUrl="@Model.ReturnUrl" method="post">
            <h4>Create a new account.</h4>
            <hr />
            <div asp-validation-summary="All" class="text-danger"></div>
            <div class="form-group">
                <label asp-for="Input.LastName"></label>
                <input asp-for="Input.LastName" class="form-control" />
                <span asp-validation-for="Input.LastName" class="text-danger"></span>
            </div>
            <div class="form-group">
                <label asp-for="Input.FirstName"></label>
                <input asp-for="Input.FirstName" class="form-control" />
                <span asp-validation-for="Input.FirstName" class="text-danger"></span>
            </div>
            <div class="form-group">
                <label asp-for="Input.PhoneNumber"></label>
                <input asp-for="Input.PhoneNumber" class="form-control" />
                <span asp-validation-for="Input.PhoneNumber" class="text-danger"></span>
            </div>
            <div class="form-group">
                <label asp-for="Input.Email"></label>
                <input asp-for="Input.Email" class="form-control" />
                <span asp-validation-for="Input.Email" class="text-danger"></span>
            </div>
        </form>
    </div>
</div>

```

. . . Continued on next page

11/02/2019 18:25 – Modified ~\Areas\Identity\Pages\Account\Register.cshtml, p. 1

. . . Continued from last page

```
<div class="form-group">
  <label asp-for="Input.Password"></label>
  <input asp-for="Input.Password" class="form-control" />
  <span asp-validation-for="Input.Password" class="text-danger"></span>
</div>
<div class="form-group">
  <label asp-for="Input.ConfirmPassword"></label>
  <input asp-for="Input.ConfirmPassword" class="form-control" />
  <span asp-validation-for="Input.ConfirmPassword" class="text-danger"></span>
</div>
<button type="submit" class="btn btn-primary">Register</button>
</form>
</div>
<div class="col-md-6 col-md-offset-2">
  <section>
    <h4>Use another service to register.</h4>
    <hr />
    @{
      if ((Model.ExternalLogins?.Count ?? 0) == 0)
      {
        <div>
          <p>
            There are no external authentication services configured. See
            <a href="https://go.microsoft.com/fwlink/?LinkID=532715">this article</a>
            for details on setting up this ASP.NET application to support logging in via external services.
          </p>
        </div>
      }
    }
  </section>
</div>
```

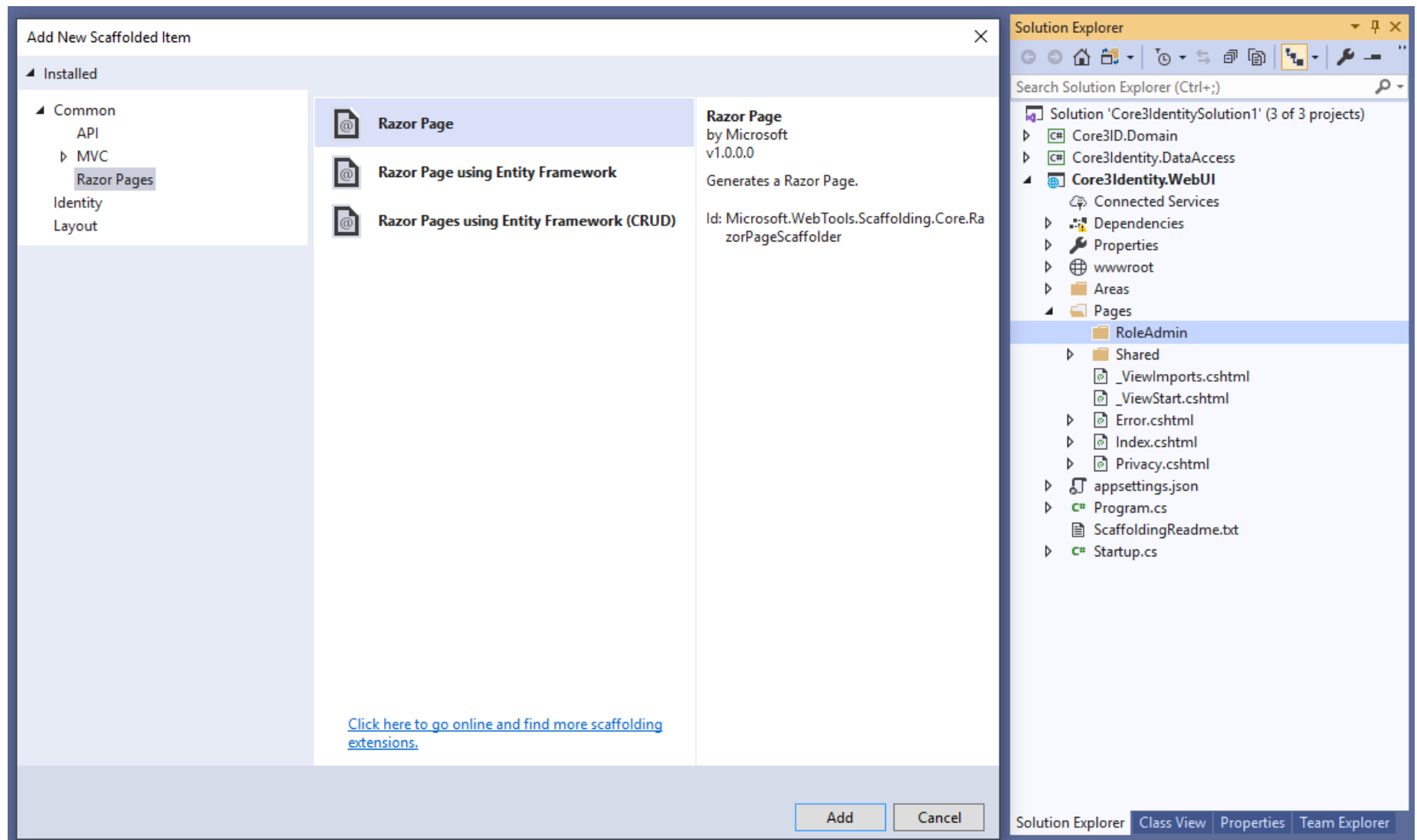
. . . Continued on next page

. . . Continued from last page

```
        else
        {
            <form id="external-account" asp-page="./ExternalLogin" asp-route-returnUrl="@Model.ReturnUrl"
                method="post" class="form-horizontal">
                <div>
                    <p>
                        @foreach (var provider in Model.ExternalLogins)
                        {
                            <button type="submit" class="btn btn-primary" name="provider" value="@provider.Name"
                                title="Log in using your @provider.DisplayName account">
                                @provider.DisplayName
                            </button>
                        }
                    </p>
                </div>
            </form>
        }
    }
</section>
</div>
</div>

@section Scripts {
    <partial name="_ValidationScriptsPartial" />
}
```

11/02/2019 18:25 – Modified ~\Areas\Identity\Pages\Account\Register.cshtml, p. 3



11/02/2019 18:56 – Scaffold a ~\Core3Identity.WebUI\Pages\RoleAdmin page

Add Razor Page

Razor Page name:

Index

Options:

☒Generate PageModel class

☐Create as a partial view

☒Reference script libraries

☒Use a layout page:

...

(Leave empty if it is set in a Razor \_viewstart file)

Add

Cancel

11/02/2019 18:58 – Scaffold ~\Core3Identity.WebUI\Pages\RoleAdmin\Index page

```

using System.Collections.Generic;
using System.Linq;
using Microsoft.AspNetCore.Identity;
using Microsoft.AspNetCore.Mvc.RazorPages;
using Core3ID.Domain;

namespace Core3Identity.WebUI.Pages.RoleAdmin
{
    public class IndexModel : PageModel
    {
        private RoleManager<IdentityRole> roleManager;
        private UserManager<Core3IDUser> userManager;

        public IEnumerable<Core3IDUser> VerityUsers { get; set; }
        public IEnumerable<IdentityRole> IdentityRoles { get; set; }

        public IndexModel(RoleManager<IdentityRole> roleMgr, UserManager<Core3IDUser> usrMgr)
        {
            roleManager = roleMgr;
            userManager = usrMgr;
        } // end public IndexModel(UserManager<VerityUser> usrMgr, ...

        public void OnGet()
        {
            IdentityRoles = roleManager.Roles.OrderBy(r => r.Name);
        } // end public void OnGet()
    } // end public class IndexModel : PageModel
} // end namespace Core3Identity.WebUI.Pages.RoleAdmin

```

**11/02/2019 19:17 – Initial ~\Core3Identity.WebUI\Pages\RoleAdmin\Index.cshtml.cs**



```

@page
@model Core3Identity.WebUI.Pages.RoleAdmin.IndexModel
@{
    ViewData["Title"] = "Index";
}

<div class="bg-primary m-1 p-1"><h4>Roles</h4></div>
<br />
<a class="btn btn-primary" asp-page="./Create">Create</a>
<br />
<br />
<div class="text-danger" asp-validation-summary="ModelOnly"></div>

<table class="table table-sm table-bordered table-bordered">
    <tr><th>ID</th><th>Name</th><th></th></tr>
    @if (Model.IdentityRoles.Count() == 0)
    {
        <tr><td colspan="4" class="text-center">No Roles</td></tr>
    }
    else
    {
        foreach (var role in Model.IdentityRoles)
        {
            <tr>
                <td>@role.Id</td>
                <td>@role.Name</td>
                <td identity-role="@role.Id"></td>
                <td>
                    <a class="btn btn-sm btn-primary" asp-page="./ManageUsers" asp-route-id="@role.Id">Manage Users</a>
                    <a class="btn btn-sm btn-primary" asp-page="./Delete" asp-route-id="@role.Id">Delete</a>
                </td>
            </tr>
        }
    }
</table>

```

11/02/2019 19:18 – Initial ~\Core3Identity.WebUI\Pages\RoleAdmin\Index.cshtml

Add Razor Page

Razor Page name:

Create

Options:

☒Generate PageModel class

☐Create as a partial view

☒Reference script libraries

☒Use a layout page:

...

(Leave empty if it is set in a Razor \_viewstart file)

Add

Cancel

11/02/2019 19:23 – Scaffold ~\Core3Identity.WebUI\Pages\RoleAdmin\Create page

```

using System.Threading.Tasks;
using Microsoft.AspNetCore.Identity;
using Microsoft.AspNetCore.Mvc;
using Microsoft.AspNetCore.Mvc.RazorPages;

namespace Core3Identity.WebUI.Pages.RoleAdmin
{
    public class CreateModel : PageModel
    {
        private RoleManager<IdentityRole> roleManager;

        public IdentityRole IdentityRole;

        public CreateModel(RoleManager<IdentityRole> roleMgr)
        {
            roleManager = roleMgr;
        } // end public CreateModel(RoleManager<IdentityRole> roleMgr)

        public IActionResult OnGet()
        {
            return Page();
        } // end public IActionResult OnGet()

        public async Task<IActionResult> OnPostAsync(string name)
        {
            if (!ModelState.IsValid)
            {
                return Page();
            }
            IdentityResult result = await roleManager.CreateAsync(new IdentityRole(name));

            return RedirectToPage("/RoleAdmin/Index");
        } // end public async Task<IActionResult> OnPostAsync()
    } // end public class CreateModel : PageModel
} // end namespace Core3Identity.WebUI.Pages.RoleAdmin

```

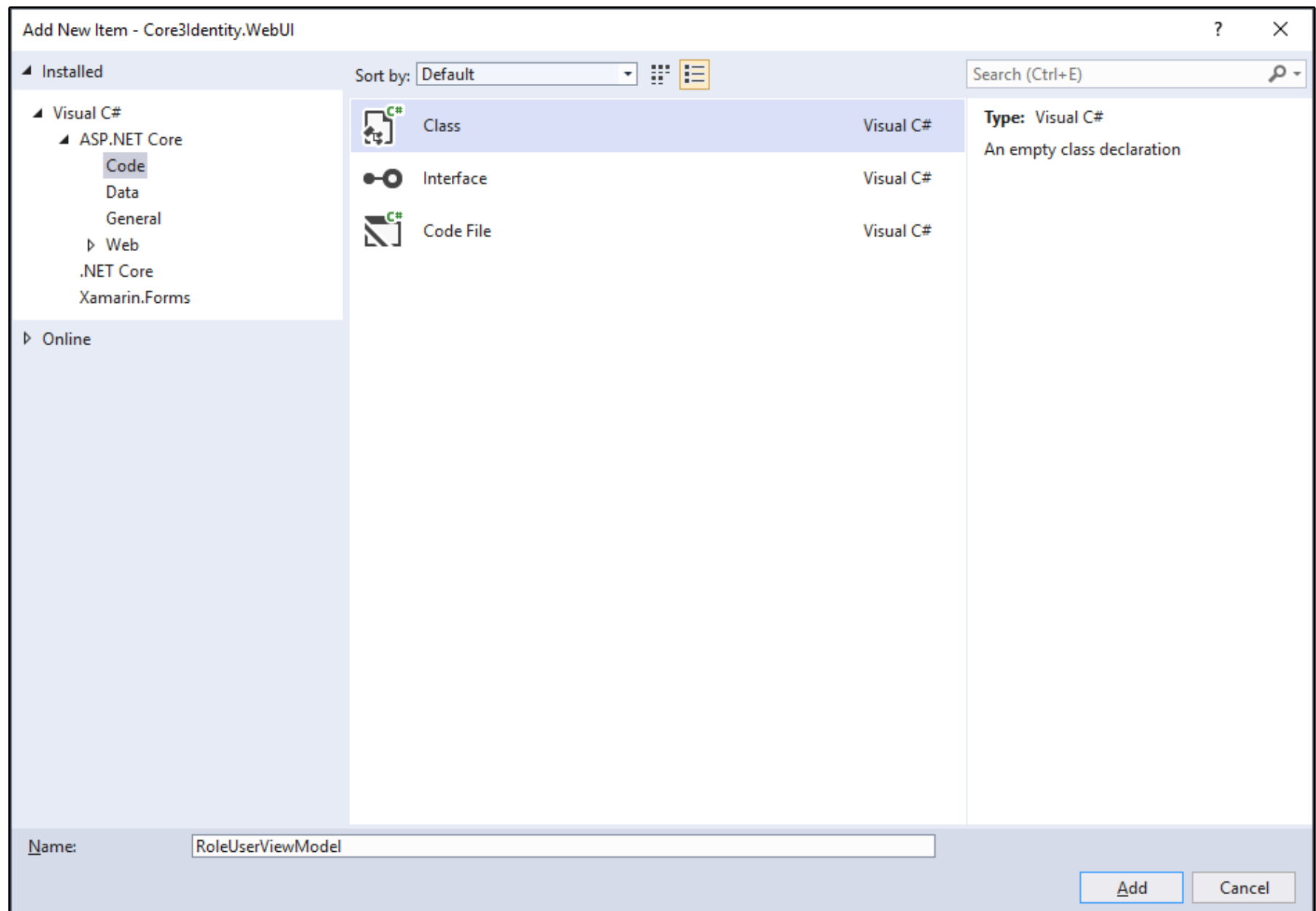
**11/02/2019 19:48 – Initial ~\Core3Identity.WebUI\Pages\RoleAdmin\Create.cshtml.cs**

```
@page
@model Core3Identity.WebUI.Pages.RoleAdmin.CreateModel
@{
    ViewData["Title"] = "Create";
}

<h1>Create Role</h1>

<p>
    Enter role name.
</p>
<div asp-validation-summary="All"></div>
<form method="post">
    <div class="form-group">
        <label for="name"></label>
        <input name="name" class="form-control" />
    </div>
    <button type="submit" class="btn btn-primary">Create</button>
    <a asp-action="Index" class="btn btn-secondary">Cancel</a>
</form>
```

11/02/2019 19:51 – Initial ~\Core3Identity.WebUI\Pages\RoleAdmin\Create.cshtml

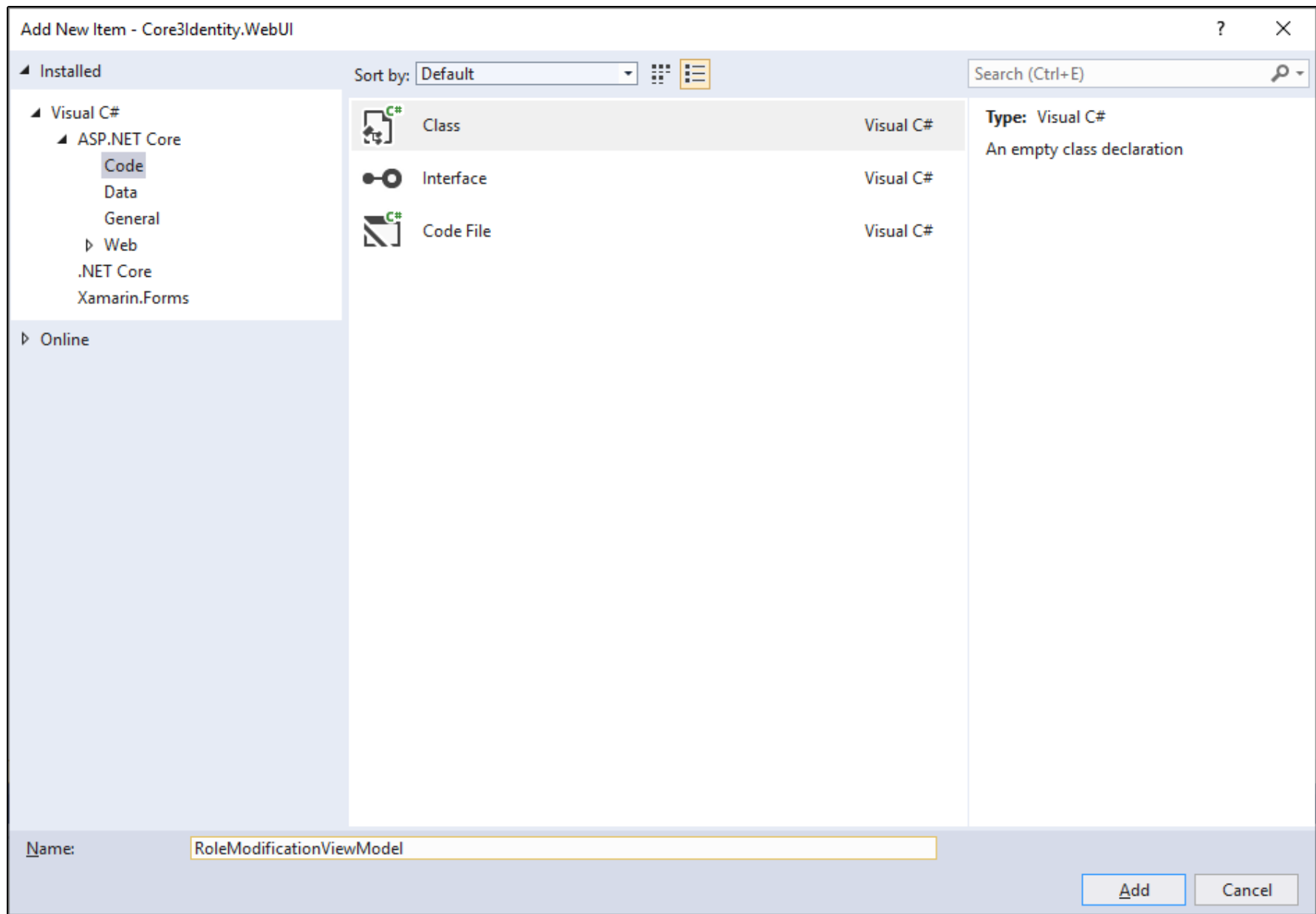


11/02/2019 20:11 - Create ~\Core3Identity.WebUI\ViewModels\RoleUserViewModel.cs

```
using System.Collections.Generic;
using Microsoft.AspNetCore.Identity;
using Core3ID.Domain;

namespace Core3Identity.WebUI.ViewModels
{
    public class RoleUserViewModel
    {
        public IdentityRole Role { get; set; }
        public IEnumerable<Core3IDUser> Members { get; set; }
        public IEnumerable<Core3IDUser> NonMembers { get; set; }
    } // end public class RoleUserViewModel
} // end namespace Core3Identity.WebUI.ViewModels
```

**11/02/2019 20:14 – Initial ~\Core3Identity.WebUI\ViewModels\RoleUserViewModel.cs**



11/02/2019 20:20 - Create ~\Core3Identity.WebUI\ViewModels\RoleModificationViewModel.cs

```

using System;
using System.Collections.Generic;
using System.Threading.Tasks;
using Microsoft.AspNetCore.Identity;
using Microsoft.AspNetCore.Mvc;
using Microsoft.AspNetCore.Mvc.RazorPages;
using Core3ID.Domain;
using Core3Identity.WebUI.ViewModels;

namespace Core3Identity.WebUI.Pages.RoleAdmin
{
    public class ManageUsersModel : PageModel
    {
        private RoleManager<IdentityRole> roleManager;
        private UserManager<Core3IDUser> userManager;

        public RoleModificationViewModel RoleModificationViewModel;
        [BindProperty]
        public RoleUserViewModel RoleUserViewModel { get; set; }

        public ManageUsersModel(RoleManager<IdentityRole> roleMgr, UserManager<Core3IDUser> userMgr)
        {
            roleManager = roleMgr;
            userManager = userMgr;
        } // end public ManageUsersModel(RoleManager<IdentityRole> roleMgr, UserManager<VerityUser> userMgr)

        public async Task<IActionResult> OnGetAsync(string id)
        {
            IdentityRole role = await roleManager.FindByIdAsync(id);
            List<Core3IDUser> members = new List<Core3IDUser>();
            List<Core3IDUser> nonMembers = new List<Core3IDUser>();
            foreach (Core3IDUser user in userManager.Users)
            {
                var list = await userManager.IsInRoleAsync(user, role.Name) ? members : nonMembers;
                list.Add(user);
            } // end foreach (Core3IDUser user in userManager.Users)
        }
    }
}

```

**. . . Continued on next page**

**11/02/2019 20:23 – Initial ~\Core3Identity.WebUI\ViewModels\RoleModificationViewModel.cs, p. 1**



**. . . Continued on next page**

```
RoleUserViewModel = new RoleUserViewModel
{
    Role = role,
    Members = members,
    NonMembers = nonMembers
};

return Page();
} // end public async Task<IActionResult> OnGetAsync(string id)
public async Task<IActionResult> OnPostAsync([FromForm]RoleModificationViewModel model)
{
    IdentityResult result;

    if (ModelState.IsValid)
    {
        if (model.IdsToAdd != null) // Avoid null exception
        {
            foreach (string addId in model.IdsToAdd)
            {
                Core3IDUser user = await userManager.FindByIdAsync(addId);
                if (user == null)
                {
                    return BadRequest(
                        String.Format("User with Id, {0}, was not found and cannot be added to role, {1}.", addId,
                            model.RoleName));
                }
                else
                {
                    result = await userManager.AddToRoleAsync(user, model.RoleName);
                    if (!result.Succeeded)
                    {
                        AddErrorsFromResult(result);
                    } // endif (!result.Succeeded)
                }
            }
        }
    }
}
```

**. . . Continued on next page**

**. . . Continued on next page**

```
        } // endif (user == null)
    } // end foreach (string addId in model.IdsToAdd)
} // endif (model.IdsToAdd != null)

if (model.IdsToDelete != null) // Avoid null exception
{
    foreach (string deleteId in model.IdsToDelete)
    {
        Core3IDUser user = await userManager.FindByIdAsync(deleteId);
        if (user == null)
        {
            return BadRequest(String.Format(
                "User with Id, {0}, was not found and cannot be deleted from role, {1}.",
                deleteId, model.RoleName));
        }
        else
        {
            result = await userManager.RemoveFromRoleAsync(user, model.RoleName);
            if (!result.Succeeded)
            {
                AddErrorsFromResult(result);
            } // endif (!result.Succeeded)
        } // endif (user == null)
    } // end foreach (string deleteId in model.IdsToDelete)
} // endif (model.IdsToDelete != null)
} // endif (ModelState.IsValid)

return RedirectToPage("/RoleAdmin/ManageUsers", new { id = model.RoleId });
} // end public async Task<IActionResult> Edit(RoleModificationViewModel model)
```

**. . . Continued on next page**

**11/02/2019 20:23 – Initial ~\Core3Identity.WebUI\ViewModels\RoleModificationViewModel.cs, p. 3**

**. . . Continued on next page**

```
private void AddErrorsFromResult(IdentityResult result)
{
    foreach (IdentityError error in result.Errors)
    {
        ModelState.AddModelError("", error.Description);
    } // end foreach (IdentityError error in result.Errors)
} // end private void AddErrorsFromResult(IdentityResult result)

} // end public class ManageUsersModel : PageModel

} // end namespace Core3Identity.WebUI.Pages.RoleAdmin
```

**11/02/2019 20:23 – Initial ~\Core3Identity.WebUI\ViewModels\RoleModificationViewModel.cs, p. 4**

Add Razor Page

Razor Page name:

ManageUsers

Options:

☒ Generate PageModel class

☐ Create as a partial view

☒ Reference script libraries

☒ Use a layout page:

...

(Leave empty if it is set in a Razor \_viewstart file)

Add

Cancel

11/02/2019 20:38 – Scaffold ~\Core3Identity.WebUI\Pages\RoleAdmin\ManageUsers page

```

using System;
using System.Collections.Generic;
using System.Threading.Tasks;
using Microsoft.AspNetCore.Identity;
using Microsoft.AspNetCore.Mvc;
using Microsoft.AspNetCore.Mvc.RazorPages;
using Core3ID.Domain;
using Core3Identity.WebUI.ViewModels;

namespace Core3Identity.WebUI.Pages.RoleAdmin
{
    public class ManageUsersModel : PageModel
    {
        private RoleManager<IdentityRole> roleManager;
        private UserManager<Core3IDUser> userManager;

        public RoleModificationViewModel RoleModificationViewModel;
        [BindProperty]
        public RoleUserViewModel RoleUserViewModel { get; set; }

        public ManageUsersModel(RoleManager<IdentityRole> roleMgr, UserManager<Core3IDUser> userMgr)
        {
            roleManager = roleMgr;
            userManager = userMgr;
        } // end public ManageUsersModel(RoleManager<IdentityRole> roleMgr, UserManager<VerityUser> userMgr)

        public async Task<IActionResult> OnGetAsync(string id)
        {
            IdentityRole role = await roleManager.FindByIdAsync(id);
            List<Core3IDUser> members = new List<Core3IDUser>();
            List<Core3IDUser> nonMembers = new List<Core3IDUser>();

```

**. . . Continued on next page**

**11/02/2019 20:48 – Initial ~\Core3Identity.WebUI\Pages\RoleAdmin\ManageUsers.cshtml.cs, p. 1**

**. . . Continued from last page**

```
foreach (Core3IDUser user in userManager.Users)
{
    var list = await userManager.IsInRoleAsync(user, role.Name) ? members : nonMembers;
    list.Add(user);
} // end foreach (Core3IDUser user in userManager.Users)

RoleUserViewModel = new RoleUserViewModel
{
    Role = role,
    Members = members,
    NonMembers = nonMembers
};

return Page();
} // end public async Task<IActionResult> OnGetAsync(string id)
public async Task<IActionResult> OnPostAsync([FromForm]RoleModificationViewModel model)
{
    IdentityResult result;

    if (ModelState.IsValid)
    {
        if (model.IdsToAdd != null) // Avoid null exception
        {
            foreach (string addId in model.IdsToAdd)
            {
                Core3IDUser user = await userManager.FindByIdAsync(addId);
                if (user == null)
                {
                    return BadRequest(String.Format(
                        "User with Id, {0}, was not found and cannot be added to role, {1}.",
                        addId, model.RoleName));
                }
            }
        }
    }
}
```

**. . . Continued on next page**

**. . . Continued from last page**

```
        else
        {
            result = await userManager.AddToRoleAsync(user, model.RoleName);
            if (!result.Succeeded)
            {
                AddErrorsFromResult(result);
            } // endif (!result.Succeeded)
        } // endif (user == null)
    } // end foreach (string addId in model.IdsToAdd)
} // endif (model.IdsToAdd != null)

if (model.IdsToDelete != null) // Avoid null exception
{
    foreach (string deleteId in model.IdsToDelete)
    {
        Core3IDUser user = await userManager.FindByIdAsync(deleteId);
        if (user == null)
        {
            return BadRequest(String.Format(
                "User with Id, {0}, was not found and cannot be deleted from role, {1}.",
                deleteId, model.RoleName));
        }
        else
        {
            result = await userManager.RemoveFromRoleAsync(user, model.RoleName);
            if (!result.Succeeded)
            {
                AddErrorsFromResult(result);
            } // endif (!result.Succeeded)
        } // endif (user == null)
    } // end foreach (string deleteId in model.IdsToDelete)
} // endif (model.IdsToDelete != null)
} // endif (ModelState.IsValid)
```

**. . . Continued on next page**

**. . . Continued from last page**

```
        return RedirectToPage("/RoleAdmin/ManageUsers", new { id = model.RoleId });
    } // end public async Task<IActionResult> Edit(RoleModificationViewModel model)

    private void AddErrorsFromResult(IdentityResult result)
    {
        foreach (IdentityError error in result.Errors)
        {
            ModelState.AddModelError("", error.Description);
        } // end foreach (IdentityError error in result.Errors)
    } // end private void AddErrorsFromResult(IdentityResult result)

} // end public class ManageUsersModel : PageModel

} // end namespace Core3Identity.WebUI.Pages.RoleAdmin
```

**11/02/2019 20:48 – Initial ~\Core3Identity.WebUI\Pages\RoleAdmin\ManageUsers.cshtml.cs, p. 4**



```

@page
@model Core3Identity.WebUI.Pages.RoleAdmin.ManageUsersModel
@{
    ViewData["Title"] = "Manage User Roles";
}

<div class="bg-primary m-1 p-1 text-white"><h4>Edit Role</h4></div>

<div asp-validation-summary="All" class="text-danger"></div>

<form method="post">
    <input type="hidden" name="roleName" value="@Model.RoleUserViewModel.Role.Name" />
    <input type="hidden" name="roleId" value="@Model.RoleUserViewModel.Role.Id" />

    <h6 class="bg-info p-1 text-white">Add To @Model.RoleUserViewModel.Role.Name</h6>
    <table class="table table-bordered table-sm">
        @if (Model.RoleUserViewModel.NonMembers.Count() == 0)
        {
            <tr><td colspan="2">All Users Are Members</td></tr>
        }
        else
        {
            @foreach (Core3IDUser user in Model.RoleUserViewModel.NonMembers)
            {
                <tr>
                    <td>@user.UserName</td>
                    <td>
                        <input type="checkbox" name="IdsToAdd" value="@user.Id">
                    </td>
                </tr>
            }
        }
    </table>

```

**. . . Continued on next page**

**. . . Continued from last page**

```
<h6 class="bg-info p-1 text-white">Remove From @Model.RoleUserViewModel.Role.Name</h6>
<table class="table table-bordered table-sm">
  @if (Model.RoleUserViewModel.Members.Count() == 0)
  {
    <tr><td colspan="2">No Users Are Members</td></tr>
  }
  else
  {
    @foreach (Core3IDUser user in Model.RoleUserViewModel.Members)
    {
      <tr>
        <td>@user.UserName</td>
        <td>
          <input type="checkbox" name="IdsToDelete" value="@user.Id">
        </td>
      </tr>
    }
  }
</table>
<button type="submit" class="btn btn-primary">Save</button>
<a asp-page="/RoleAdmin/Index" class="btn btn-secondary">Cancel</a>
</form>
```

**11/02/2019 20:56 – Initial ~\Core3Identity.WebUI\Pages\RoleAdmin\ManageUsers.cshtml, p. 2**

```
using Microsoft.AspNetCore.Builder;
using Microsoft.AspNetCore.Hosting;
using Microsoft.AspNetCore.Identity;
using Microsoft.EntityFrameworkCore;
using Microsoft.Extensions.Configuration;
using Microsoft.Extensions.DependencyInjection;
using Microsoft.Extensions.Hosting;
using Core3ID.Domain;
using Core3Identity.DataAccess;

namespace Core3Identity.WebUI
{
    public class Startup
    {
        public Startup(IConfiguration configuration)
        {
            Configuration = configuration;
        }

        public IConfiguration Configuration { get; }

        // This method gets called by the runtime. Use this method to add services to the container.
        public void ConfigureServices(IServiceCollection services)
        {
            services.AddDbContextPool<Core3IDContext>(options =>
            {
                options.UseSqlServer(Configuration.GetConnectionString("DefaultConnection"));
            });
        }
    }
}
```

**. . . Continued on next page**

**11/03/2019 16:08 – Initial ~\Core3Identity.WebUI\Startup.cshtml, p. 1**

**. . . Continued from last page**

```
services.AddIdentity<Core3IDUser, IdentityRole>(opts => {
    opts.User.RequireUniqueEmail = true;
    opts.Password.RequiredLength = 6;
    opts.Password.RequireNonAlphanumeric = true;
    opts.Password.RequireLowercase = true;
    opts.Password.RequireUppercase = true;
    opts.Password.RequireDigit = true;
}).AddEntityFrameworkStores<Core3IDContext>()
.AddDefaultUI()
.AddDefaultTokenProviders();
services.AddRazorPages();
} // end public void ConfigureServices(IServiceCollection services)

// This method gets called by the runtime. Use this method to configure the HTTP request pipeline.
public void Configure(IApplicationBuilder app, IWebHostEnvironment env)
{
    if (env.IsDevelopment())
    {
        app.UseDeveloperExceptionPage();
    }
    else
    {
        app.UseExceptionHandler("/Error");
        // The default HSTS value is 30 days. You may want to change this for production scenarios,
        // see https://aka.ms/aspnetcore-hsts.
        app.UseHsts();
    } // endif (env.IsDevelopment())

    app.UseHttpsRedirection();
    app.UseStaticFiles();

    app.UseRouting();
}
```

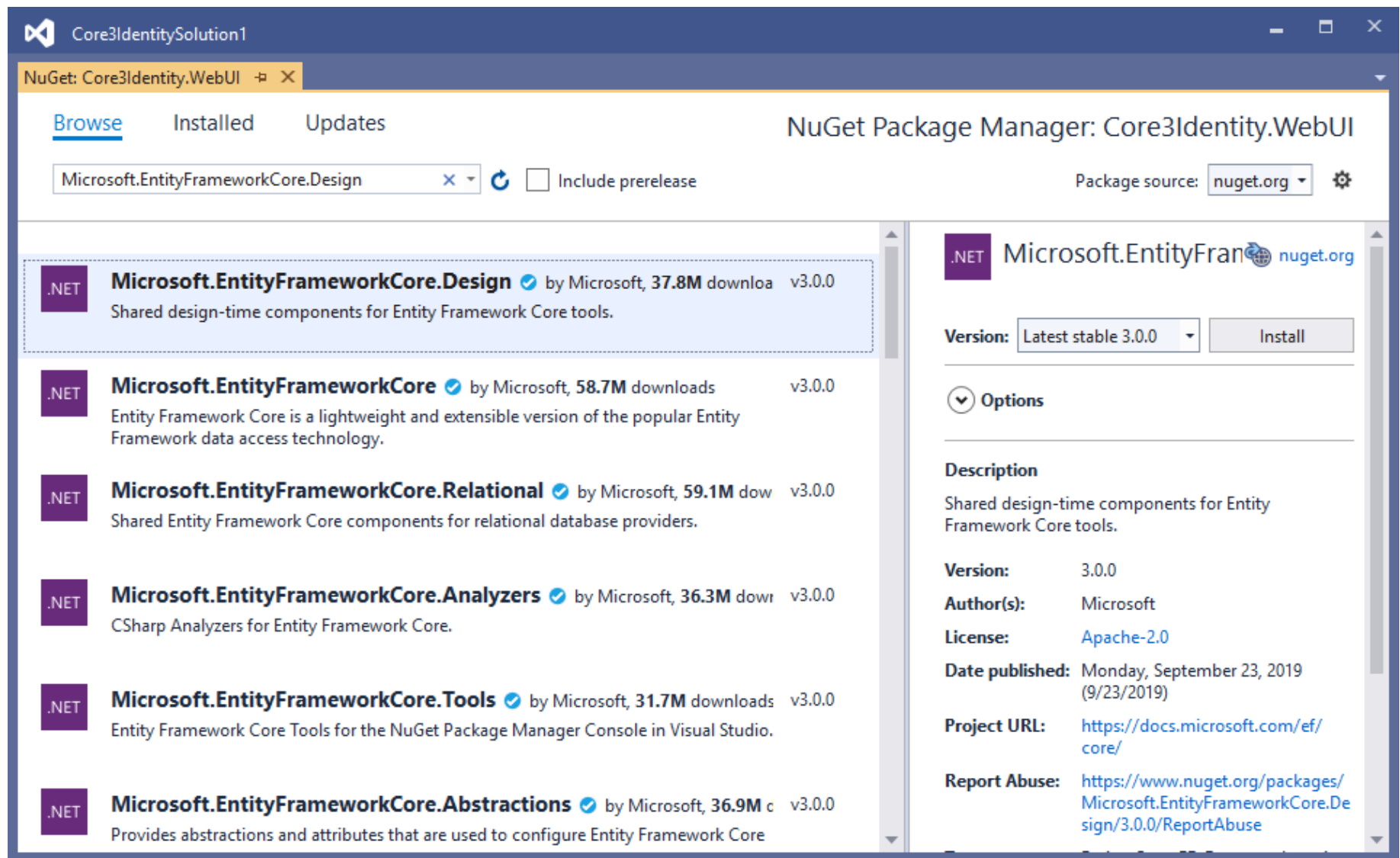
**. . . Continued on next page**

**. . . Continued from last page**

```
    app.UseAuthorization();

    app.UseEndpoints(endpoints =>
    {
        endpoints.MapRazorPages();
    });
}
} // end public class Startup
} // end namespace Core3Identity.WebUI
```

**11/03/2019 16:08 – Initial ~\Core3Identity.WebUI\Startup.cshtml, p. 3**



11/03/2019 16:40 – Add NuGet pkg, Microsoft.EntityFrameworkCore.Design 3.0.0 to Core3Identity.WebUI

Core3IdentitySolution1 - NuGet: Core3Identity.WebUI

NuGet: Core3Identity.WebUI

Browse Installed Updates

Microsoft.EntityFrameworkCore.Design x ↻ ☐ Include prerelease Package source: nuget.org ⚙

**Microsoft.EntityFrameworkCore.Relational** ✓ by Microsoft, 59.1M dow v3.0.0  
Shared Entity Framework Core components for relational database providers.

**Microsoft.EntityFrameworkCore.Analyzers** ✓ by Microsoft, 36.3M dow v3.0.0  
CSharp Analyzers for Entity Framework Core.

**Microsoft.EntityFrameworkCore.Tools** ✓ by Microsoft, 31.7M downloads v3.0.0  
Entity Framework Core Tools for the NuGet Package Manager Console in Visual Studio.

**Microsoft.EntityFrameworkCore.Abstractions** ✓ by Microsoft, 36.9M d v3.0.0  
Provides abstractions and attributes that are used to configure Entity Framework Core

**Microsoft.EntityFrameworkCore.SqlServer** ✓ by Microsoft, 39.1M dow v3.0.0  
Microsoft SQL Server database provider for Entity Framework Core.

**Microsoft.EntityFrameworkCore.InMemory** ✓ by Microsoft, 27.8M dow v3.0.0  
In-memory database provider for Entity Framework Core (to be used for testing)

**Microsoft.EntityFr** nuget.org

Version: 3.0.0 Install

Options

**Description**  
Entity Framework Core Tools for the NuGet Package Manager Console in Visual Studio.

Enables these commonly used commands:  
Add-Migration  
Drop-Database  
Get-DbContext  
Scaffold-DbContext  
Script-Migrations  
Update-Database

Version: 3.0.0  
Author(s): Microsoft  
License: Apache-2.0  
Date published: Monday, September 23, 2019

11/03/2019 16:43 – Add NuGet pkg, Microsoft.EntityFrameworkCore.Tools 3.0.0 to Core3Identity.WebUI

```
PM> Add-Migration Initial-Migration-w-Identity
Microsoft.EntityFrameworkCore.Infrastructure[10403]
    Entity Framework Core 3.0.0 initialized 'Core3IDContext' using provider 'Microsoft.EntityFrameworkCore.SqlServer' with
options: MaxPoolSize=128
To undo this action, use Remove-Migration.
PM>
```

**11/03/2019 16:50 – PMC: Add-Migration Initial-Migration-w-Identity**



```

using System;
using Microsoft.EntityFrameworkCore.Migrations;

namespace Core3Identity.DataAccess.Migrations
{
    public partial class InitialMigrationwIdentity : Migration
    {
        protected override void Up(MigrationBuilder migrationBuilder)
        {
            migrationBuilder.CreateTable(
                name: "AspNetRoles",
                columns: table => new
                {
                    Id = table.Column<string>(nullable: false),
                    Name = table.Column<string>(maxLength: 256, nullable: true),
                    NormalizedName = table.Column<string>(maxLength: 256, nullable: true),
                    ConcurrencyStamp = table.Column<string>(nullable: true)
                },
                constraints: table =>
                {
                    table.PrimaryKey("PK_AspNetRoles", x => x.Id);
                });

            migrationBuilder.CreateTable(
                name: "AspNetUsers",
                columns: table => new
                {
                    Id = table.Column<string>(nullable: false),
                    UserName = table.Column<string>(maxLength: 256, nullable: true),
                    NormalizedUserName = table.Column<string>(maxLength: 256, nullable: true),
                    Email = table.Column<string>(maxLength: 256, nullable: true),
                    NormalizedEmail = table.Column<string>(maxLength: 256, nullable: true),
                    EmailConfirmed = table.Column<bool>(nullable: false),
                    PasswordHash = table.Column<string>(nullable: true),

```

**. . . Continued on next page**

**. . . Continued from last page**

```
        SecurityStamp = table.Column<string>(nullable: true),
        ConcurrencyStamp = table.Column<string>(nullable: true),
        PhoneNumberConfirmed = table.Column<bool>(nullable: false),
        TwoFactorEnabled = table.Column<bool>(nullable: false),
        LockoutEnd = table.Column<DateTimeOffset>(nullable: true),
        LockoutEnabled = table.Column<bool>(nullable: false),
        AccessFailedCount = table.Column<int>(nullable: false),
        LastName = table.Column<string>(maxLength: 50, nullable: false),
        FirstName = table.Column<string>(maxLength: 50, nullable: false),
        PhoneNumber = table.Column<string>(nullable: true)
    },
    constraints: table =>
    {
        table.PrimaryKey("PK_AspNetUsers", x => x.Id);
    });

migrationBuilder.CreateTable(
    name: "AspNetRoleClaims",
    columns: table => new
    {
        Id = table.Column<int>(nullable: false)
            .Annotation("SqlServer:Identity", "1, 1"),
        RoleId = table.Column<string>(nullable: false),
        ClaimType = table.Column<string>(nullable: true),
        ClaimValue = table.Column<string>(nullable: true)
    },
    constraints: table =>
    {
```

**. . . Continued on next page**

**. . . Continued from last page**

```
        table.PrimaryKey("PK_AspNetRoleClaims", x => x.Id);
        table.ForeignKey(
            name: "FK_AspNetRoleClaims_AspNetRoles_RoleId",
            column: x => x.RoleId,
            principalTable: "AspNetRoles",
            principalColumn: "Id",
            onDelete: ReferentialAction.Cascade);
    });

    migrationBuilder.CreateTable(
        name: "AspNetUserClaims",
        columns: table => new
        {
            Id = table.Column<int>(nullable: false)
                .Annotation("SqlServer:Identity", "1, 1"),
            UserId = table.Column<string>(nullable: false),
            ClaimType = table.Column<string>(nullable: true),
            ClaimValue = table.Column<string>(nullable: true)
        },
        constraints: table =>
        {
            table.PrimaryKey("PK_AspNetUserClaims", x => x.Id);
            table.ForeignKey(
                name: "FK_AspNetUserClaims_AspNetUsers_UserId",
                column: x => x.UserId,
                principalTable: "AspNetUsers",
                principalColumn: "Id",
                onDelete: ReferentialAction.Cascade);
        }
    });
```

**. . . Continued on next page**

**. . . Continued from last page**

```
migrationBuilder.CreateTable(
    name: "AspNetUserLogins",
    columns: table => new
    {
        LoginProvider = table.Column<string>(nullable: false),
        ProviderKey = table.Column<string>(nullable: false),
        ProviderDisplayName = table.Column<string>(nullable: true),
        UserId = table.Column<string>(nullable: false)
    },
    constraints: table =>
    {
        table.PrimaryKey("PK_AspNetUserLogins", x => new { x.LoginProvider, x.ProviderKey });
        table.ForeignKey(
            name: "FK_AspNetUserLogins_AspNetUsers_UserId",
            column: x => x.UserId,
            principalTable: "AspNetUsers",
            principalColumn: "Id",
            onDelete: ReferentialAction.Cascade);
    });

migrationBuilder.CreateTable(
    name: "AspNetUserRoles",
    columns: table => new
    {
        UserId = table.Column<string>(nullable: false),
        RoleId = table.Column<string>(nullable: false)
    },
    constraints: table =>
    {
        table.PrimaryKey("PK_AspNetUserRoles", x => new { x.UserId, x.RoleId });
    });
```

**. . . Continued on next page**

**. . . Continued from last page**

```
        table.ForeignKey(
            name: "FK_AspNetUserRoles_AspNetRoles_RoleId",
            column: x => x.RoleId,
            principalTable: "AspNetRoles",
            principalColumn: "Id",
            onDelete: ReferentialAction.Cascade);
        table.ForeignKey(
            name: "FK_AspNetUserRoles_AspNetUsers_UserId",
            column: x => x.UserId,
            principalTable: "AspNetUsers",
            principalColumn: "Id",
            onDelete: ReferentialAction.Cascade);
    });

    migrationBuilder.CreateTable(
        name: "AspNetUserTokens",
        columns: table => new
        {
            UserId = table.Column<string>(nullable: false),
            LoginProvider = table.Column<string>(nullable: false),
            Name = table.Column<string>(nullable: false),
            Value = table.Column<string>(nullable: true)
        },
        constraints: table =>
        {
            table.PrimaryKey("PK_AspNetUserTokens", x => new { x.UserId, x.LoginProvider, x.Name });
            table.ForeignKey(
                name: "FK_AspNetUserTokens_AspNetUsers_UserId",
                column: x => x.UserId,
                principalTable: "AspNetUsers",
                principalColumn: "Id",
                onDelete: ReferentialAction.Cascade);
        }
    );
```

**. . . Continued on next page**

**. . . Continued from last page**

```
migrationBuilder.CreateIndex(  
    name: "IX_AspNetRoleClaims_RoleId",  
    table: "AspNetRoleClaims",  
    column: "RoleId");  
  
migrationBuilder.CreateIndex(  
    name: "RoleNameIndex",  
    table: "AspNetRoles",  
    column: "NormalizedName",  
    unique: true,  
    filter: "[NormalizedName] IS NOT NULL");  
  
migrationBuilder.CreateIndex(  
    name: "IX_AspNetUserClaims_UserId",  
    table: "AspNetUserClaims",  
    column: "UserId");  
  
migrationBuilder.CreateIndex(  
    name: "IX_AspNetUserLogins_UserId",  
    table: "AspNetUserLogins",  
    column: "UserId");  
  
migrationBuilder.CreateIndex(  
    name: "IX_AspNetUserRoles_RoleId",  
    table: "AspNetUserRoles",  
    column: "RoleId");  
  
migrationBuilder.CreateIndex(  
    name: "EmailIndex",  
    table: "AspNetUsers",  
    column: "NormalizedEmail");
```

**. . . Continued on next page**

**11/03/2019 16:52 – Generated ~\Core3Identity.DataAccess\Migrations\20191103224810\_Initial-Migration-w-Identity.cs, p. 6**

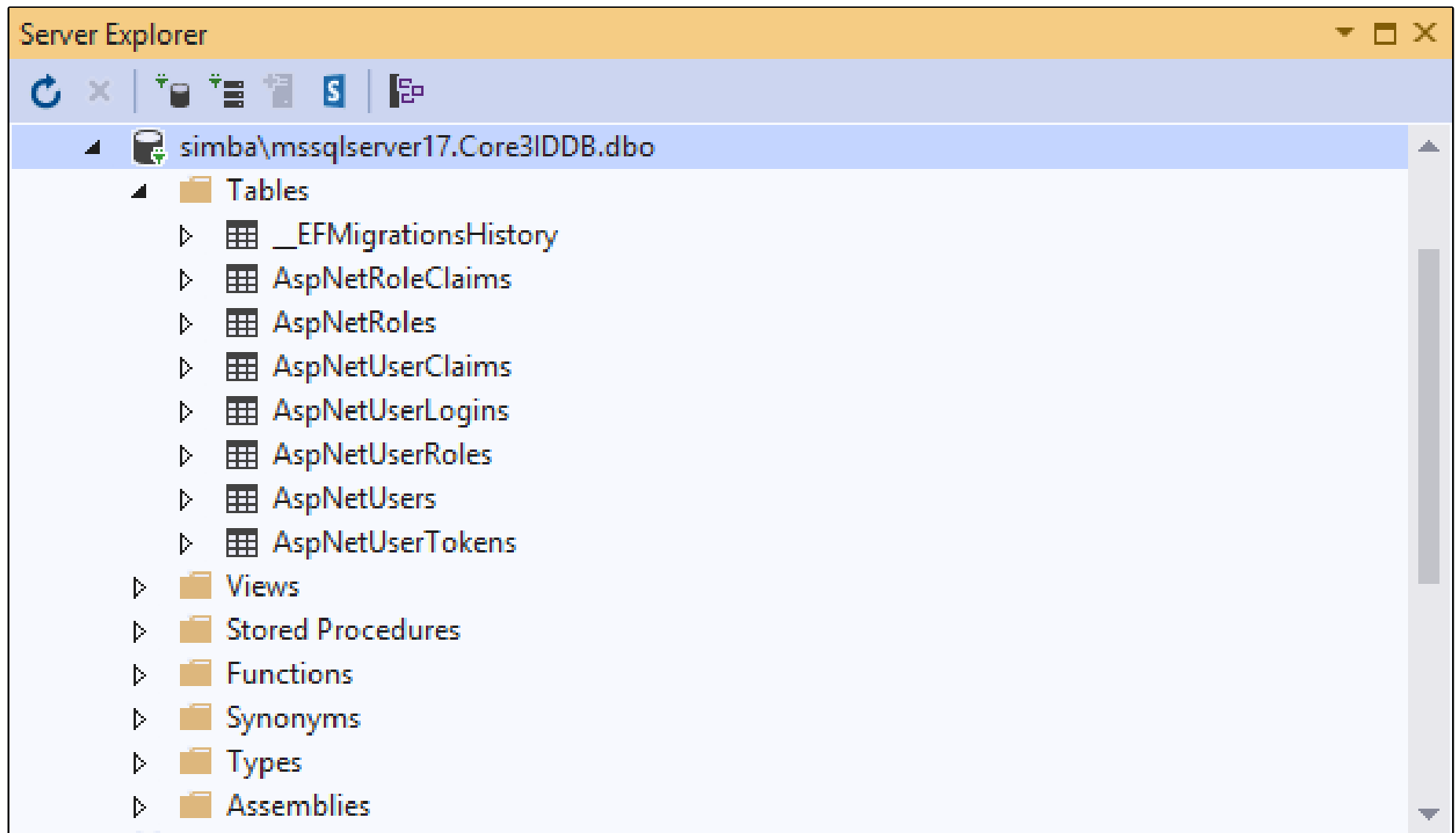
**. . . Continued from last page**

```
migrationBuilder.CreateIndex(  
    name: "UserNameIndex",  
    table: "AspNetUsers",  
    column: "NormalizedUserName",  
    unique: true,  
    filter: "[NormalizedUserName] IS NOT NULL");  
}  
  
protected override void Down(MigrationBuilder migrationBuilder)  
{  
    migrationBuilder.DropTable(  
        name: "AspNetRoleClaims");  
  
    migrationBuilder.DropTable(  
        name: "AspNetUserClaims");  
  
    migrationBuilder.DropTable(  
        name: "AspNetUserLogins");  
  
    migrationBuilder.DropTable(  
        name: "AspNetUserRoles");  
  
    migrationBuilder.DropTable(  
        name: "AspNetUserTokens");  
  
    migrationBuilder.DropTable(  
        name: "AspNetRoles");  
  
    migrationBuilder.DropTable(  
        name: "AspNetUsers");  
    }  
}
```

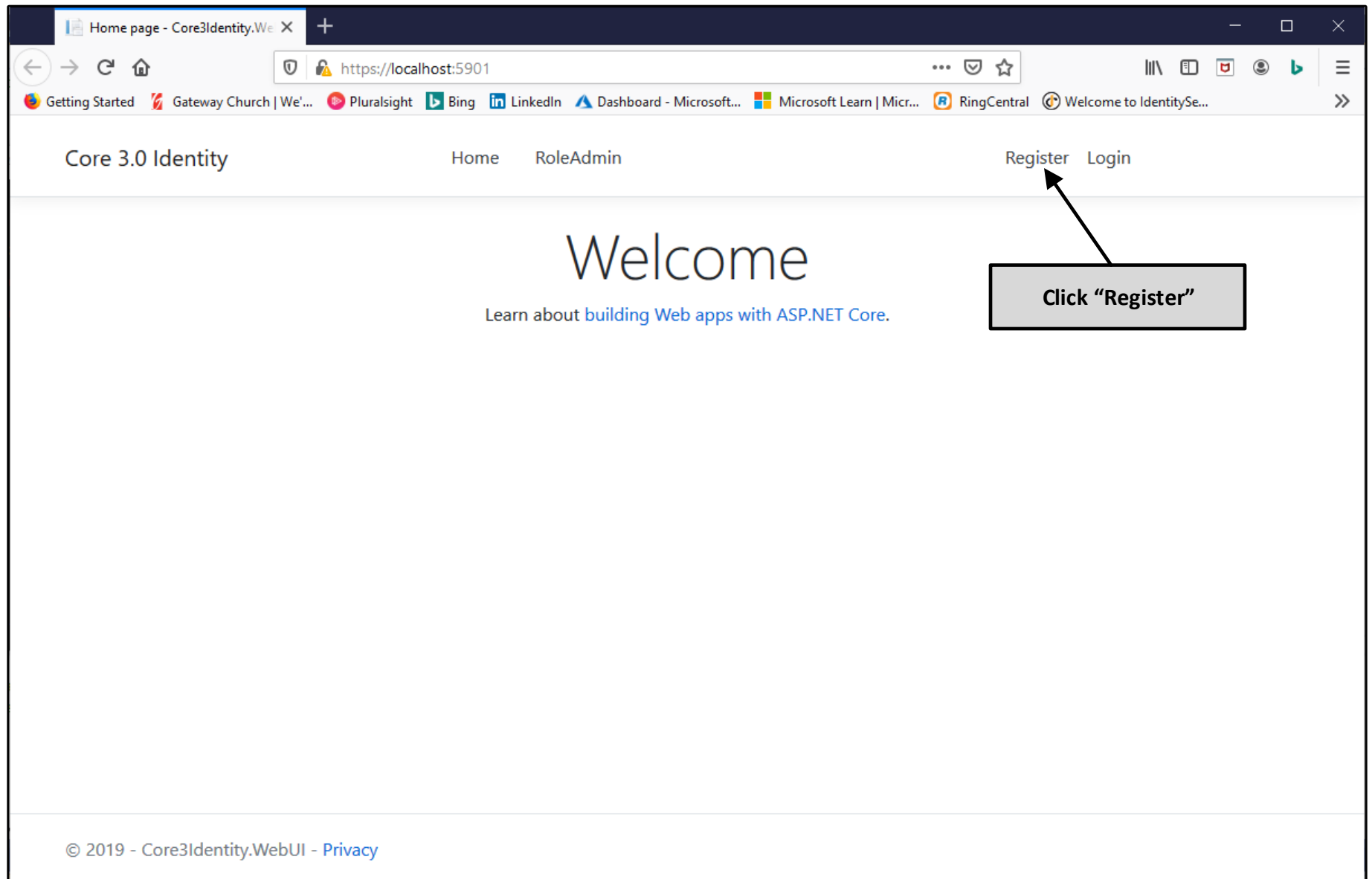
```
PM> Update-Database
Microsoft.EntityFrameworkCore.Infrastructure[10403]
    Entity Framework Core 3.0.0 initialized 'Core3IDContext' using provider 'Microsoft.EntityFrameworkCore.SqlServer' with
options: MaxPoolSize=128
. . . Lines omitted
Microsoft.EntityFrameworkCore.Database.Command[20100]
    Executing DbCommand [Parameters=[], CommandType='Text', CommandTimeout='30']
    INSERT INTO [__EFMigrationsHistory] ([MigrationId], [ProductVersion])
    VALUES (N'20191103224810_Initial-Migration-w-Identity', N'3.0.0');
Done.
```

**11/03/2019 17:21 – PMC: Update-Database for Initial-Migration-w-Identity**





11/03/2019 17:26 – Server Explorer: Identity tables added to Core3IDDB



Run 11/03/2019 18:15 – https://localhost:5901/

Register - Core3Identity.WebUI

https://localhost:5901/Identity/Account/Register

Core 3.0 Identity Home RoleAdmin Register Login

## Register

Create a new account.

Use another service to register.

There are no external authentication services configured. See [this article](#) for details on setting up this ASP.NET application to support logging in via external services.

Last Name

James

First Name

Leonard

Phone Number

4692358213

Email

lenj@lionharttech.com

Password

.....

Confirm password

.....

Register

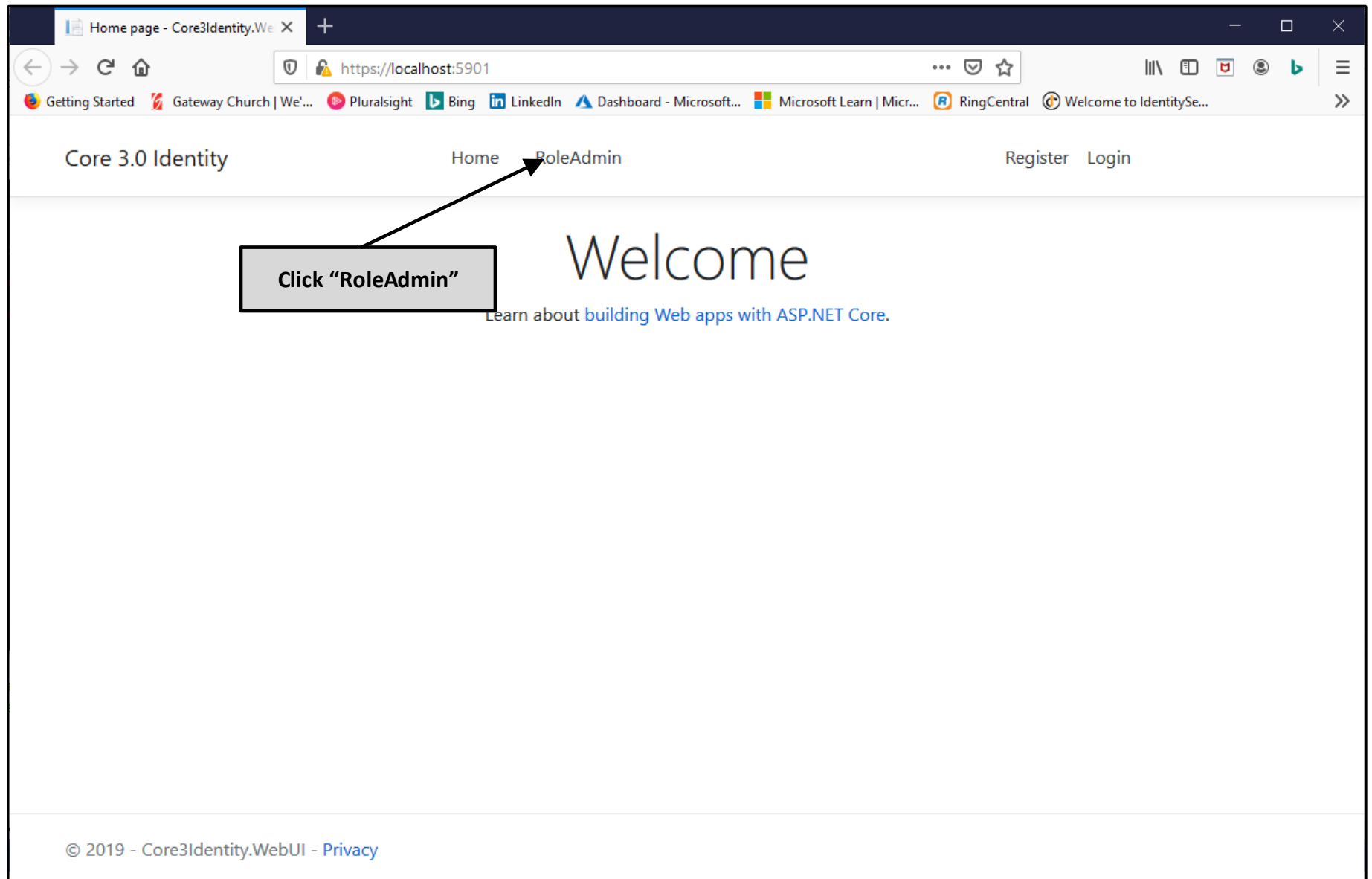
Click "Register"

© 2019 - Core3Identity.WebUI - [Privacy](#)

Run 11/03/2019 18:22 – https://localhost:5901/Identity/Account/Register

<b>Core3IdentitySolution1 Test Accounts</b>				
<b>Email/UserId</b>	<b>Last Name</b>	<b>First Name</b>	<b>Phone</b>	<b>Password</b>
<a href="mailto:lenj@lionharttech.com">lenj@lionharttech.com</a>	James	Leonard	4692358213	T121%ctcSkwr
<a href="mailto: johndoe@doefamily.com">johndoe@doefamily.com</a>	Doe	John	9725552587	T121%ctcSkwr
<a href="mailto: janedoe@doefamily.com">janedoe@doefamily.com</a>	Doe	Jane	9725552587	T121%ctcSkwr
<a href="mailto: larrydoe@doefamily.com">larrydoe@doefamily.com</a>	Doe	Larry	9725552587	T121%ctcSkwr

11/03/2019 18:38 – Core3IdentitySolution Test Accounts



Run 11/03/2019 18:44 – https://localhost:5901/

Index - Core3Identity.WebUI x +

https://localhost:5901/RoleAdmin

Getting Started Gateway Church | We... Pluralsight Bing LinkedIn Dashboard - Microsoft... Microsoft Learn | Micr... RingCentral Welcome to IdentitySe... Visual Studio IDE, Cod...

Core 3.0 Identity Home RoleAdmin Register Login

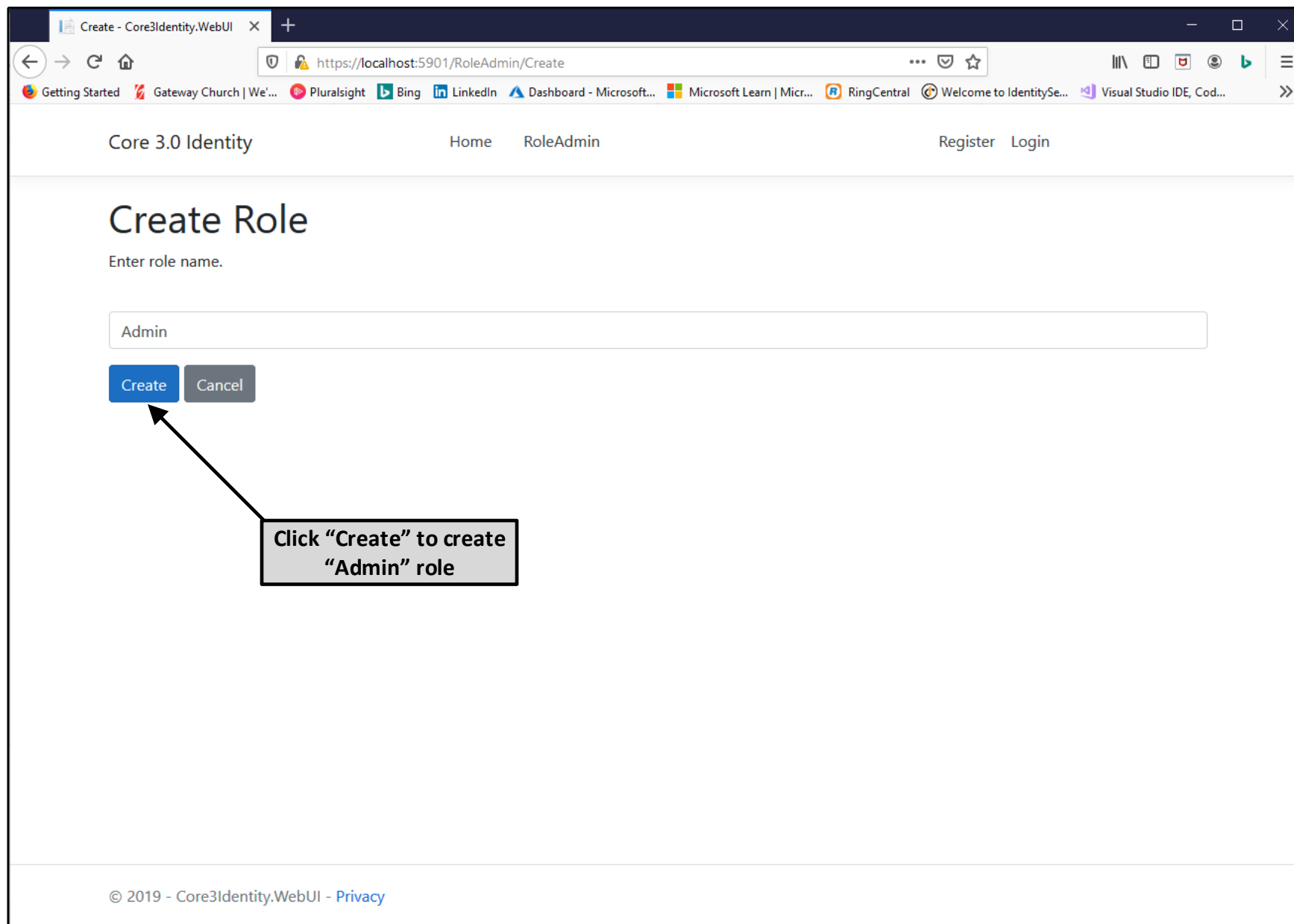
## Roles

Create ← Click "Create"

ID	Name
No Roles	

© 2019 - Core3Identity.WebUI - [Privacy](#)

Run 11/03/2019 18:48 – https://localhost:5901/RoleAdmin



Run 11/03/2019 18:49 – https://localhost:5901/RoleAdmin/Create

Index - Core3Identity.WebUI

+

https://localhost:5901/RoleAdmin

Getting Started Gateway Church | We'... Pluralsight Bing LinkedIn Dashboard - Microsoft... Microsoft Learn | Micr... RingCentral Welcome to IdentitySe... Visual Studio IDE, Cod...

Core 3.0 Identity

Home RoleAdmin

Register Login

Roles

Create

ID	Name	
e928b96b-864a-423d-945c-cf776a4b3482	Admin	<div>Manage UsersDelete</div>

Create Manager, Foreman,  
Employee, & Customer Roles

© 2019 - Core3Identity.WebUI - Privacy

Run 11/03/2019 18:52 – https://localhost:5901/RoleAdmin



Index - Core3Identity.WebUI

+

https://localhost:5901/RoleAdmin

Getting Started Gateway Church | We'... Pluralsight Bing LinkedIn Dashboard - Microsoft... Microsoft Learn | Micr... RingCentral Welcome to IdentitySe... Visual Studio IDE, Cod...

Core 3.0 Identity

Home RoleAdmin

Register Login

Roles

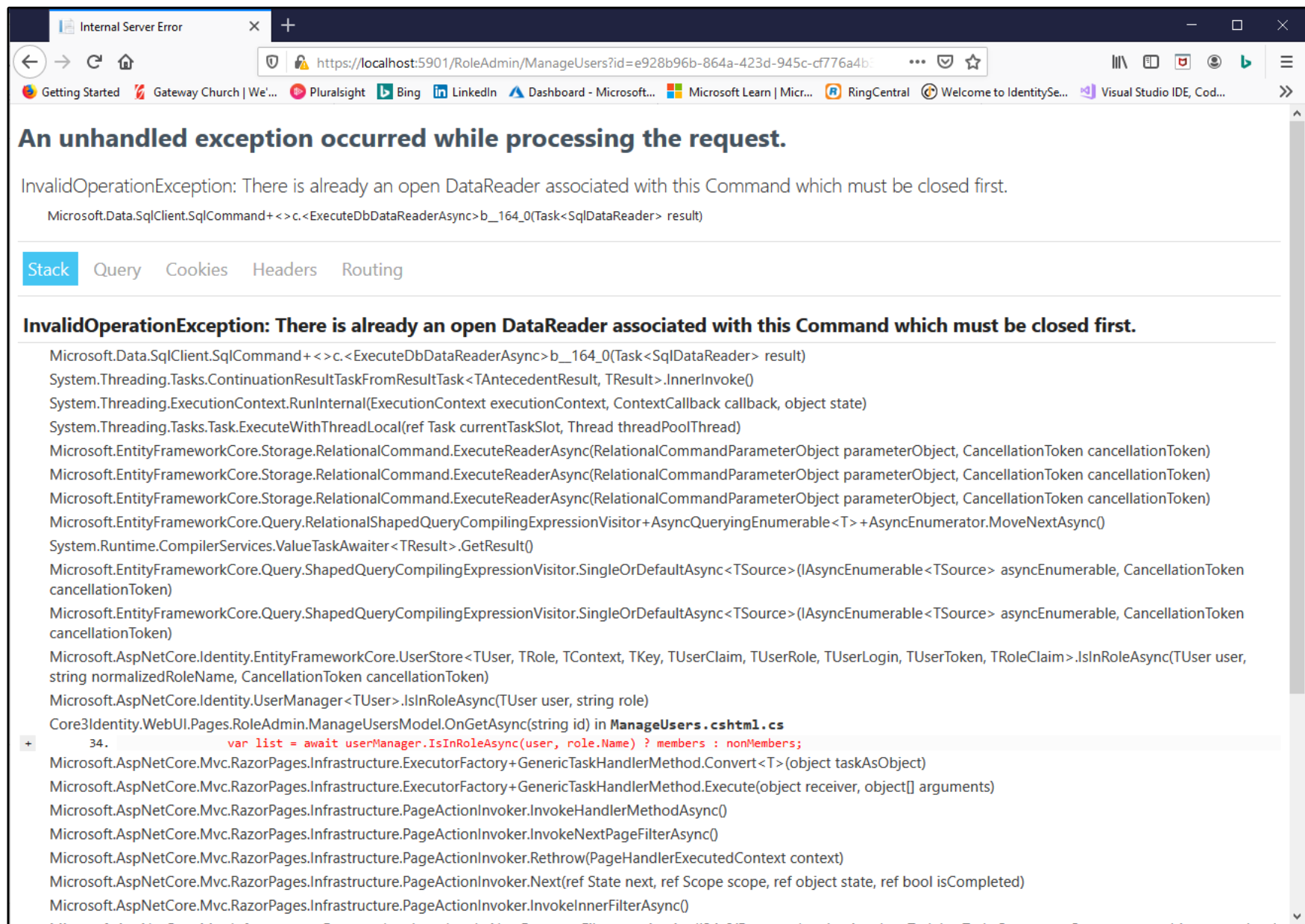
Create

Click "Manage Users" to manage "Admin" role

ID	Name	
e928b96b-864a-423d-945c-cf776a4b3482	Admin	<div>Manage UsersDelete</div>
1d2bdaa1-29fb-498d-949d-edcaa83aba86	Customer	<div>Manage UsersDelete</div>
8615194c-aa68-43ef-8390-56bd8e9360c9	Employee	<div>Manage UsersDelete</div>
7528b730-1b35-42e3-9e58-699211c11a86	Foreman	<div>Manage UsersDelete</div>
2c954f1d-7c5e-42cd-a20f-ae1a4a7ac2cf	Manager	<div>Manage UsersDelete</div>

© 2019 - Core3Identity.WebUI - Privacy

Run 11/03/2019 18:56 – https://localhost:5901/RoleAdmin



Run 11/03/2019 18:59 – Got InvalidOperationException in ManageUsers.cshtml.cs

Back Up Database - Core3IDDB

Select a page

- General
- Media Options
- Backup Options

Script Help

Source

Database: Core3IDDB

Recovery model: FULL

Backup type: Full

☐ Copy-only backup

Backup component:

☒ Database

☐ Files and filegroups:

Destination

Back up to: Disk

L:\Samples\ASPNET-Identity\DBBackups\Core3IDDB\Core3IDDB\_Initial-with-Users-n-Roles\_2019-11-03-1910.bak

Add...

Remove

Contents

Connection

Server: SIMBA\MSSQLSERVER17

Connection: SIMBA\venj

[View connection properties](#)

Progress

Ready

OK Cancel

11/03/2019 19:10 – SSMS: Create DB backup, Core3IDDB\_Initial-with-Users-n-Roles\_2019-11-03-1910.bak

Team Explorer - Commit Details

Commit Details | Core3IdentitySolution1

Commit 03f2f738

Leonard E. James <lenj@lionharttech.com>  
11/3/2019 7:05:50 PM

Parent: a8dcf9e0

Add project files.

[Revert](#) | [Amend Message](#) | [Reset](#) | [Create Tag](#) | [Actions](#)

Changes (87)

Core3ID.Domain

Core3ID.Domain.csproj [add]

Core3IDUser.cs [add]

Core3Identity.DataAccess

Migrations

Core3IDContext.cs [add]

Core3Identity.DataAccess.csproj [add]

Core3Identity.WebUI

Areas\Identity

Pages

Properties

ViewModels

wwwroot

appsettings.Development.json [add]

appsettings.json [add]

Core3Identity.WebUI.csproj [add]

Program.cs [add]

ScaffoldingReadme.txt [add]

Startup.cs [add]

Core3IdentitySolution1.sln [add]

GitId: 03f2f738



# **NETCore3 Identity Solution #1**

**Starting GitId: 03f2f738**

**Add project files**

**Push to remote repository, Core3IdentitySolution1, Next**

**Base Folder**

**L:\Samples\ASPNET-Identity\**

**Source Folder**

**~\Core3IdentitySolution1**

**Git-Archives**

**~\Git-Archives\Core3IdentitySolution1\  
2019-11-02-1906\_03f2f738\_Add project files.zip**

**11/03/2019 19:06**

**Page 85**