**ASP.NET Core Identity**



Implementing login/logout, Scaffolding



**SoftUni Team**

**Technical Trainers**

**Software University**



[**https://softuni.bg**](https://softuni.bg/)



**Table of Contents**

##### Authentication vs. Authorization

1. ASP.NET Core Identity



1. Scaffolding Identity



**Have a Question?**

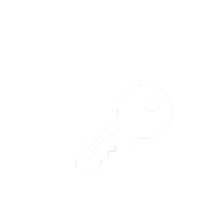
# sli.do



**#csharp-web**



## Authentication vs. Authorization



**Authentication vs. Authorization**

#### Authentication

* + The process of verifying the identity of a user or computer
  + Prerequisite for authorization



* + Questions: **Who are you**? How you prove it?
  + Credentials can be password, smart card, external token, etc.

#### Authorization

* + The process of determining what a user is permitted to do on a

computer or network

* + Questions: **What are you allowed to do**? Can you see this page?

##### You can't authorize a user before authenticating this user

5



**Authentication vs. Authorization (2)**



## ASP.NET Core Identity



**ASP.NET Core Identity (1)**

* The **ASP.NET Core Identity** system
  + Authentication and authorization system for ASP.NET Core



* + Supports ASP.NET Core MVC, Pages, Web API (JWT), SignalR
  + Handles **Users**, **User Profiles**, **Login** / **Logout**, **Roles**, etc.
  + Handles cookie consent and GDPR



* + Supports external login providers
    - Facebook, Google, Twitter, etc.
  + Supports database, Azure, Active Directory, Windows Users, etc.



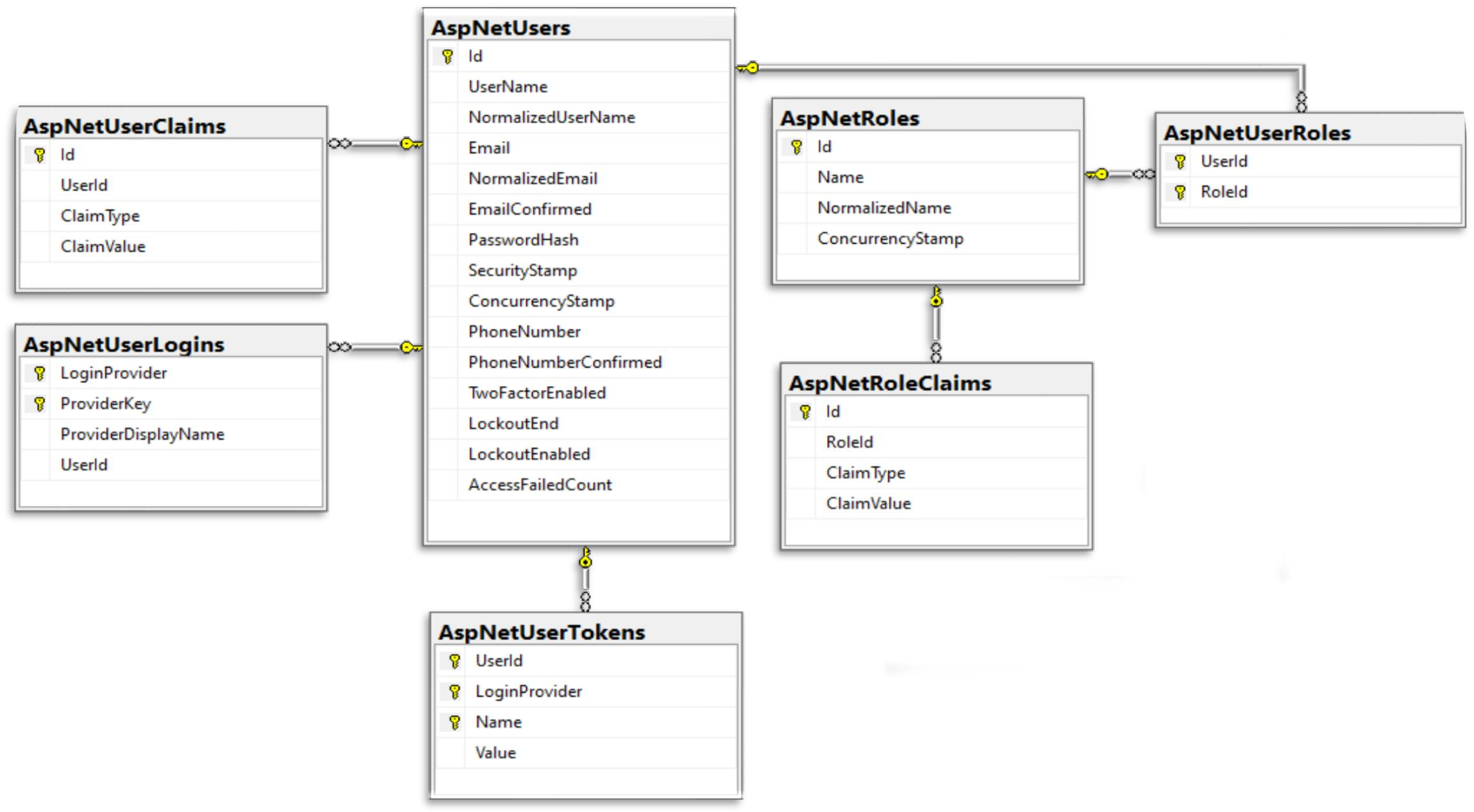
**ASP.NET Core Identity (2)**

* Typically, the **ASP.NET Core** identity data is stored in relational database



* + Data is persisted using **Entity Framework Core**
  + You have some control over the internal database schema

### Internal Database Schema



**ASP.NET Core Identity System Setup**

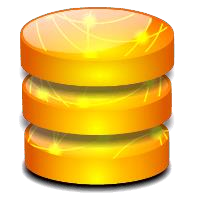
* Setup **ASP.NET Identity**
  + Using the ASP.NET **project templates** from Visual Studio



* + - And then customize it
  + **By hand**
    - Install NuGet packages, manual configuration, create

EF mappings (models), view models, controllers, views, etc.

##### Required NuGet package

* + **Microsoft.AspNetCore.Identity.EntityFrameworkCore**
* **ApplicationDbContext.cs**
  + Holds the



EF data context



**ASP.NET Core Project Template Authentication (1)**

* + Provides access to the application's data using model objects

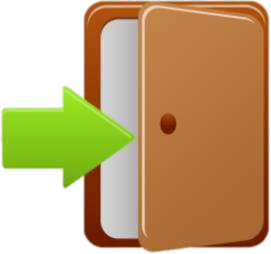
#### Program.cs

* + Can configure cookie-based (or JWT) authentication
  + May enable external login (e.g. Facebook login)
  + Can change default identity settings
    - More on those in the next course



**ASP.NET Core Project Template Authentication (2)**

* **Password settings** – can be defined in **Program.cs**



**builder.Services.AddDefaultIdentity<IdentityUser>(options =>**

**{**

**// Password, lockout, emails, etc. options.SignIn.RequireConfirmedAccount = false; options.Password.RequireDigit = false;**

**})**

**.AddEntityFrameworkStores<ApplicationDbContext>();**



**Authorization**

* Use the [**Authorize**] and [**AllowAnonymous**] attributes to configure

**Authorized** / **Anonymous access** for **Controller** / **Action**

**[Authorize]**



**public class AccountController : Controller**

**{**

**// GET: /Account/Login (anonymous)**

**[AllowAnonymous]**

**public async Task<IActionResult> Login(string returnUrl) { … }**

**// POST: /Account/LogOff (for logged-in users only)**

**[HttpPost]**

**public async Task<IActionResult> Logout() { … }**

**}**



### Check the Currently Logged-In User

**// GET: /Account/Roles (for logged-in users only)**

**[Authorize]**

**public ActionResult Roles()**

**{**

**var currentUser = await userManager.GetUserAsync(this.User);**

**var roles = await userManager.GetRolesAsync(currentUser);**

**...**

**}**

**// GET: /Account/Data (for logged-in users only) [Authorize]**

**public ActionResult Data()**

**{**

**var currentUser = await userManager.GetUserAsync(this.User);**

**var currentUserUsername = await userManager.GetUserNameAsync(currentUser);**

**var currentUserId = await userManager.GetUserIdAsync(currentUser);**

**...**

**}**



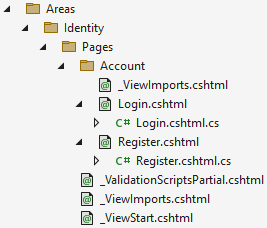
**ASP.NET Core User Manager**

|  |  |  |
| --- | --- | --- |
| * **UserManager<TU**   **Category** | **ser>** - APIs for man | aging users in a persistence store |
|  |  |  |
|  |  |  |
| **AddClaimsAsync(…)** | **FindByEmailAsync(…)** | **GenerateChangePhoneNumberTokenAsync(…)** |
| **AddToRoleAsync(…)** | **FindByIdAsync(…)** | **GenerateEmailConfirmationTokenAsync(…)** |
| **IsInRoleAsync(…)** | **FindByNameAsync(…)** | **GeneratePasswordResetTokenAsync(…)** |
| **GetUserId(…)** | **GetClaimsAsync(…)** | **GetAuthenticationTokenAsync(…)** |
| **ChangeEmailAsync(…)** | **GetEmailAsync(…)** | **IsEmailConfirmedAsync(…)** |
|  |  |  |
| **ConfirmEmailAsync(…)** | **GetRolesAsync(…)** | **CreateSecurityTokenAsync(…)** |
| **CreateAsync(…)** | **GetUserAsync(…)** | **ResetPasswordAsync(…)** |
| **DeleteAsync(…)** | **CheckPasswordAsync(…)** | **RemoveFromRoleAsync(…)** |
| **Dispose(…)** | **UpdatePassword(…)** | **RemoveClaimsAsync(…)**  16 |



## Scaffolding Identity

* Since **ASP.NET Core 2.2**, **Identity** is provided as a **Razor Class Library**



* The **scaffolder** can be configured to generate source code
  + If you need to modify the code and change the behavior
* Most of the necessary code is generated by the **scaffolder**

**Scaffolded**

**Account.Register**



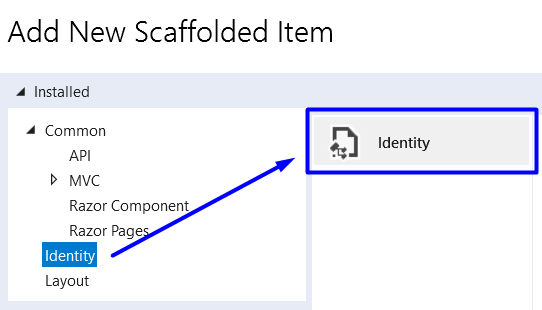
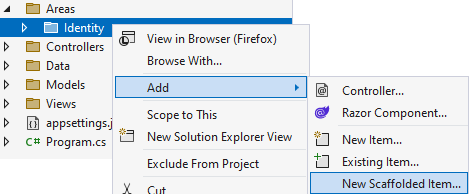
**Scaffolding ASP.NET Core Identity**

**and Account.Login**



**Scaffolding ASP.NET Core Identity in VS**

* Scaffold Identity pages by adding a **new scaffolded identity item**



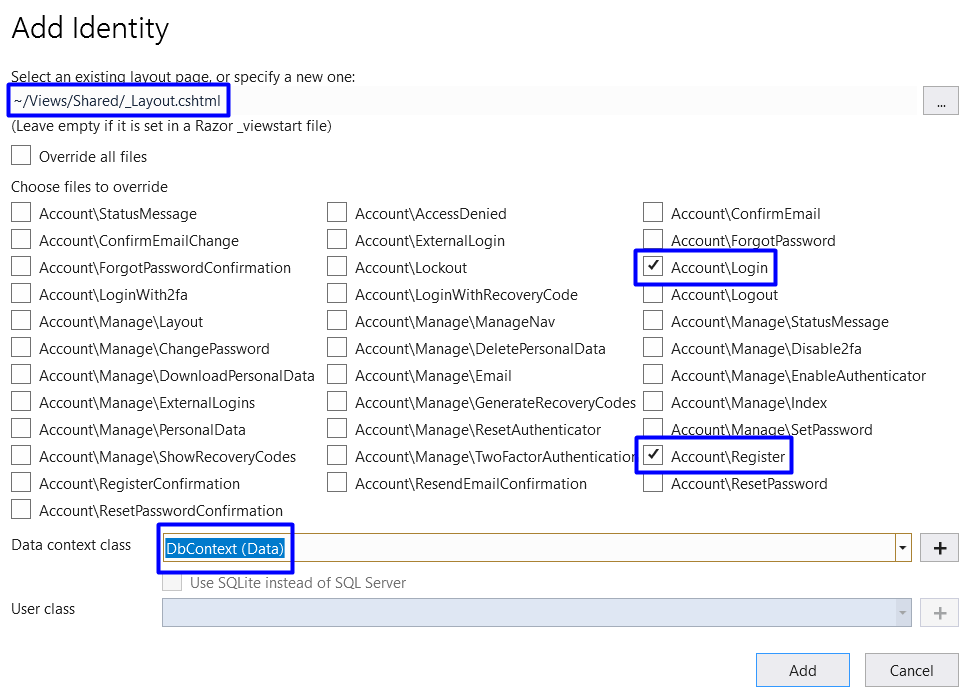
### Scaffolding ASP.NET Core Identity in VS (2)

* **Scaffolded pages** will be part of the "**/Areas/Identity**" folder

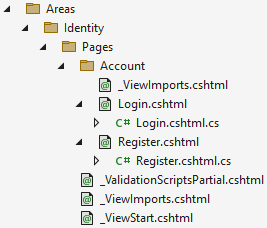
**Select the**

**layout page**

**Select pages to be scaffolded**



**Select the db context**



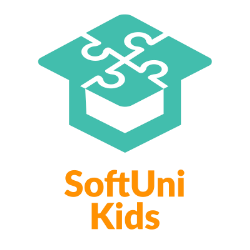
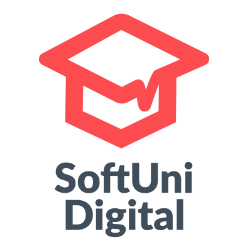
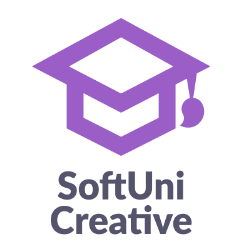
**Summary**

* 1. Authentication vs. Authorization



* 2. **ASP.NET Core Identity**

3. **Scaffolding** Identity

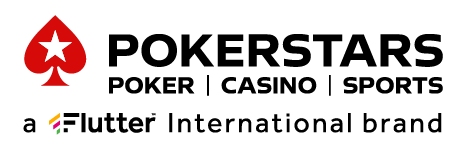
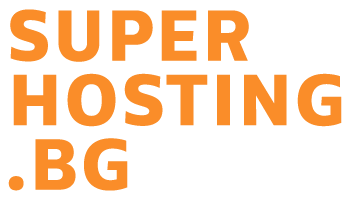




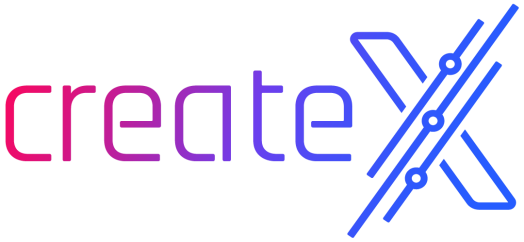
# A picture containing text, sign, vector graphics Description automatically generatedQuestions?

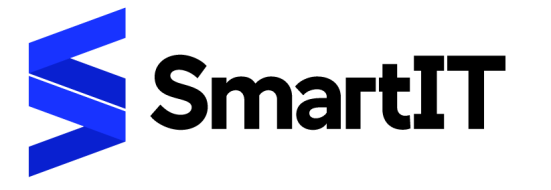






**SoftUni Diamond Partners**







**Educational Partners**



**Trainings @ Software University (SoftUni)**

* Software University – High-Quality Education, Profession and Job for Software Developers



* + [softuni.bg](https://softuni.bg/), [about.softuni.bg](https://about.softuni.bg/)



* Software University Foundation
  + [softuni.foundation](https://softuni.foundation/)
* Software University @ Facebook
  + [facebook.com/SoftwareUniversity](https://www.facebook.com/SoftwareUniversity)
* Software University Forums
  + [forum.softuni.bg](https://forum.softuni.bg/)



**License**

##### This course (slides, examples, demos, exercises, homework,

documents, videos and other assets) is **copyrighted content**



##### Unauthorized copy, reproduction or use is illegal

* © SoftUni – [https://about.softuni.bg](https://about.softuni.bg/)
* © Software University – [https://softuni.bg](https://softuni.bg/)