**Plan: State Management**

**​Objective**

In this lecture, the students should be introduced to ASP.NET Core Identity, differences between Authorization and Authentication and Scaffolding Identity

**Motivation**

It's important to know how to scaffold Identity in order to use the default identity users for basic actions like register, login and logout.

**Content**

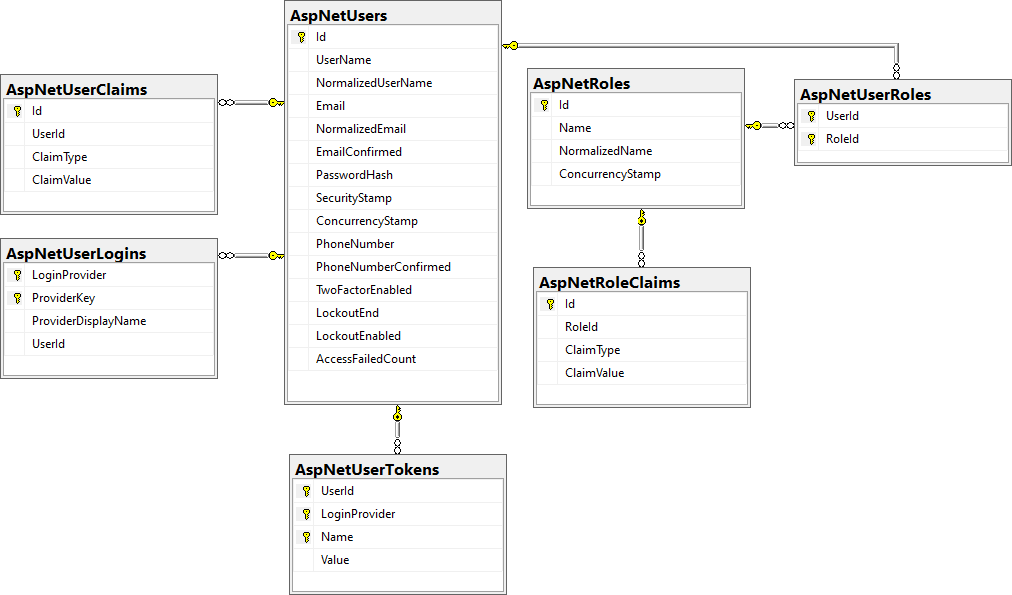
### Authentication vs Authorization (~25 min)

* **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?

### Q&A [Sli.do] (~10 min)

### ASP.NET Core Identity (~25 min)

* **ASP.NET Core Identity**
  + 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.
  + 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**



### BREAK (15 min)

### ASP.NET Core Identity (~40 min)

* **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**
* **ASP.NET Core Project Template Authentication**
  + **ApplicationDbContext.cs**
    - Holds the EF data context
    - 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
  + **Password settings** 
    - Can be defined in **Program.cs**

### Q&A [Sli.do] (~10 min)

### BREAK: 15 min

### ASP.NET Core Identity (~40 min)

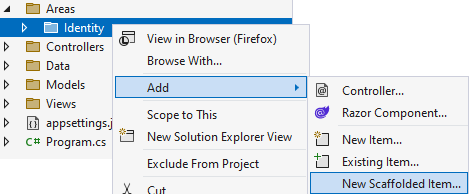
* **Authorization**
  + - Use the [**Authorize**] and [**AllowAnonymous**] attributes to configure **Authorized** / **Anonymous** **access** for **Controller** / **Action**
* **Check the Currently Logged-in User**
* **ASP.NET Core User Manager**
  + **UserManager<TUser>** - APIs for managing users in a persistence store

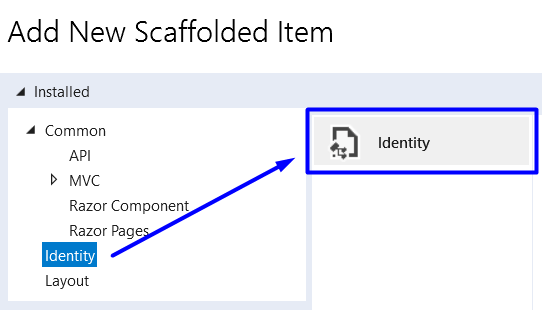
### Q&A [Sli.do] (~10 min)

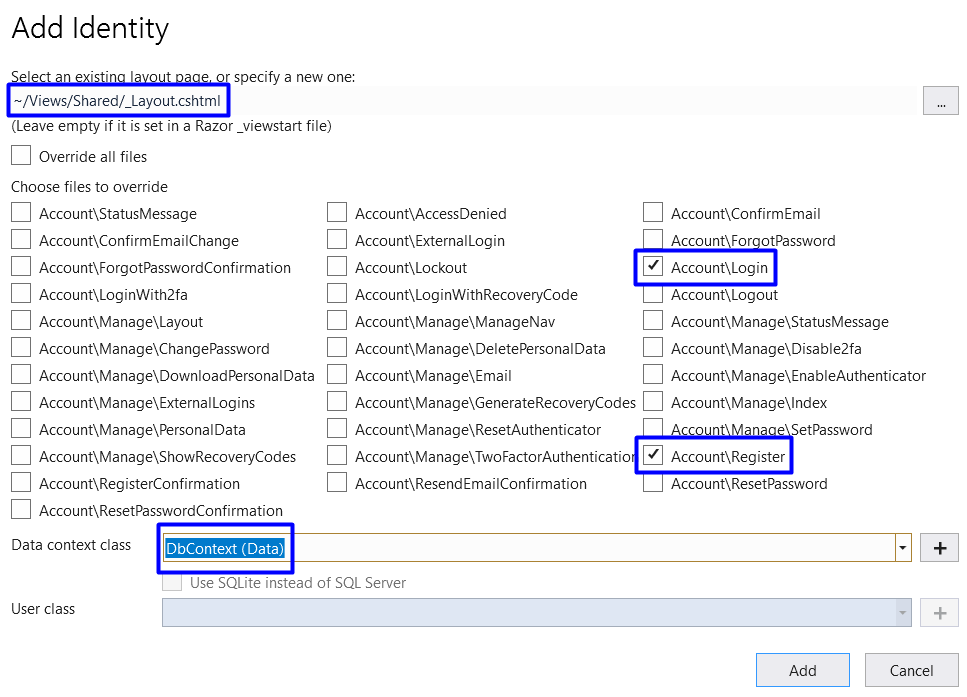
### BREAK: 10 min

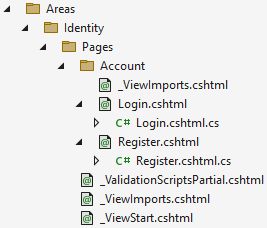
### Scaffolding Identity (~30 min)

* **Scaffolding ASP.NET Core 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**
* **Scaffolding ASP.NET Core Identity in VS**
  + Scaffold Identity pages by adding a **new** **scaffolded** **identity item**









### Q&A [Sli.do] (~10 min)

**Exercise**

Give tasks and examples that include creating a user, log in/logout/register and different views for logged in and logged out users.

**Evaluation & Exam**

The lecture will be included in the exam.