ASP Core Authentication

With ASP.NET **identity**

AARHUS UNIVERSITY
SCHOOL OF ENGINEERING

Definitions

- **Authentication**
  is the process of verifying that "you are who you say you are“

- **Authorization**
  is the process of verifying that "you are permitted to do what you are trying to do"

What is ASP.NET Identity?

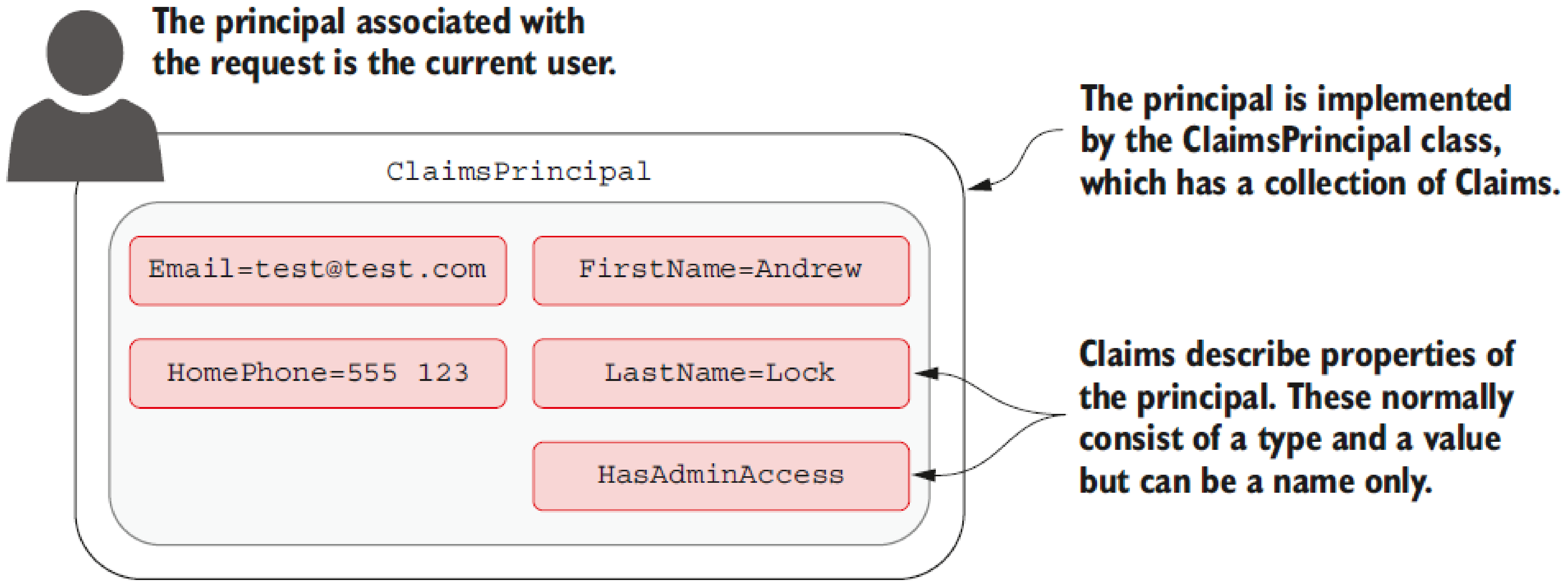
- Identity is a membership system which allows you to add login functionality to your application
- Users can create an account and login with a user name and password
- Or they can use an external login providers such as Facebook, Google, Microsoft Account, Twitter and more

- You can configure ASP.NET Core Identity to use a SQL Server database to store user names, passwords, and profile data. Alternatively, you can use your own persistent store, for example Azure Table Storage

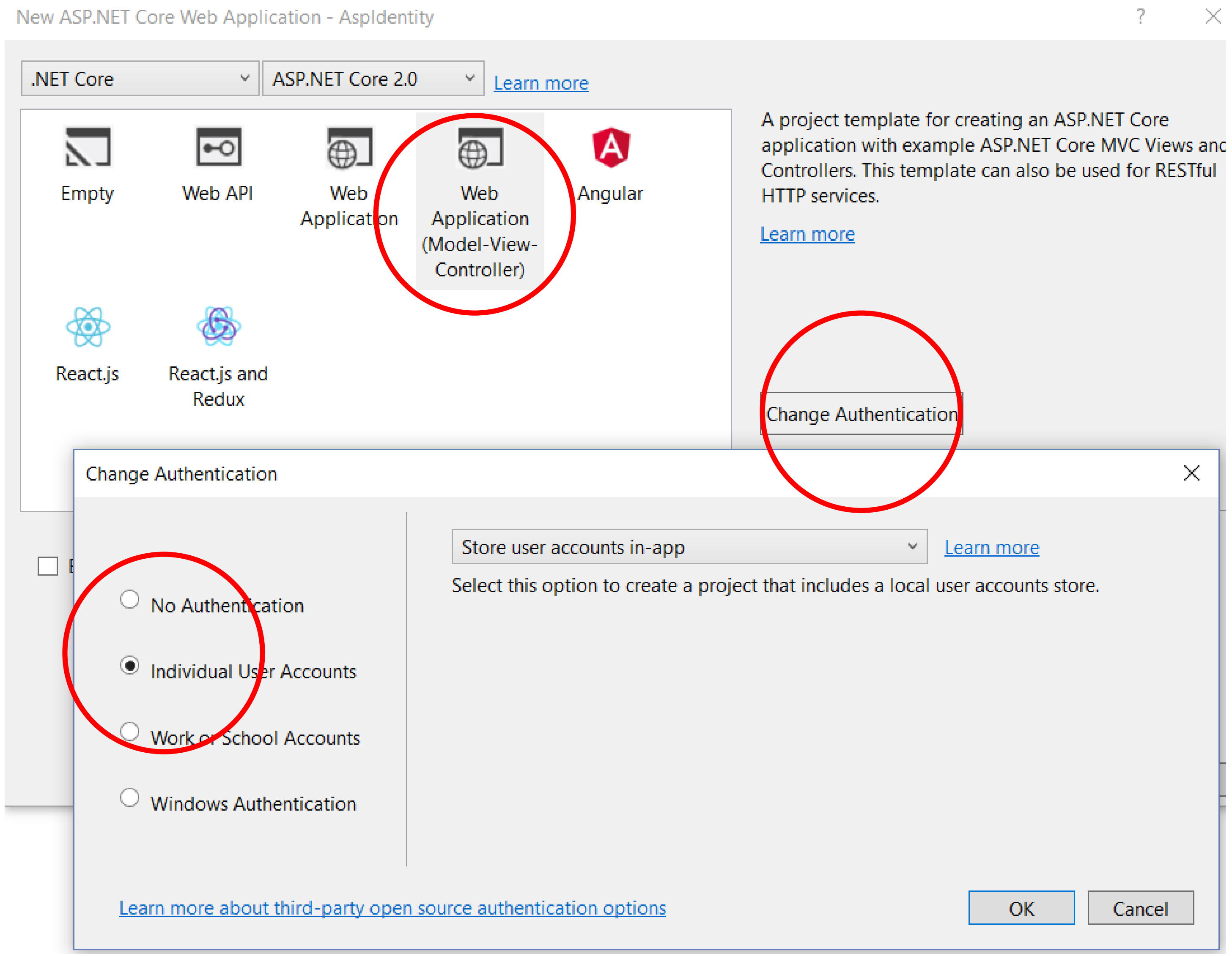
*Use the **Identity Package** to implement A&A in ASP.Net Core*

The principal is the current user

- The HttpContext object exposes the current *principal* for a request as the User property


The principal associated with the request is the current user.
ClaimsPrincipal
Email=test@test.com
FirstName=Andrew
HomePhone=555 123
LastName=Lock
HasAdminAccess
The principal is implemented by the ClaimsPrincipal class, which has a collection of Claims.
Claims describe properties of the principal. These normally consist of a type and a value but can be a name only.

Select Individual User Accounts for a new project



New ASP.NET Core Web Application - AspIdentity

.NET Core | ASP.NET Core 2.0 | Learn more

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.

Learn more

Change Authentication

Change Authentication

- No Authentication
- [✓ Individual User Accounts]
- Work or School Accounts
- Windows Authentication

Store user accounts in-app | Learn more

Select this option to create a project that includes a local user accounts store.

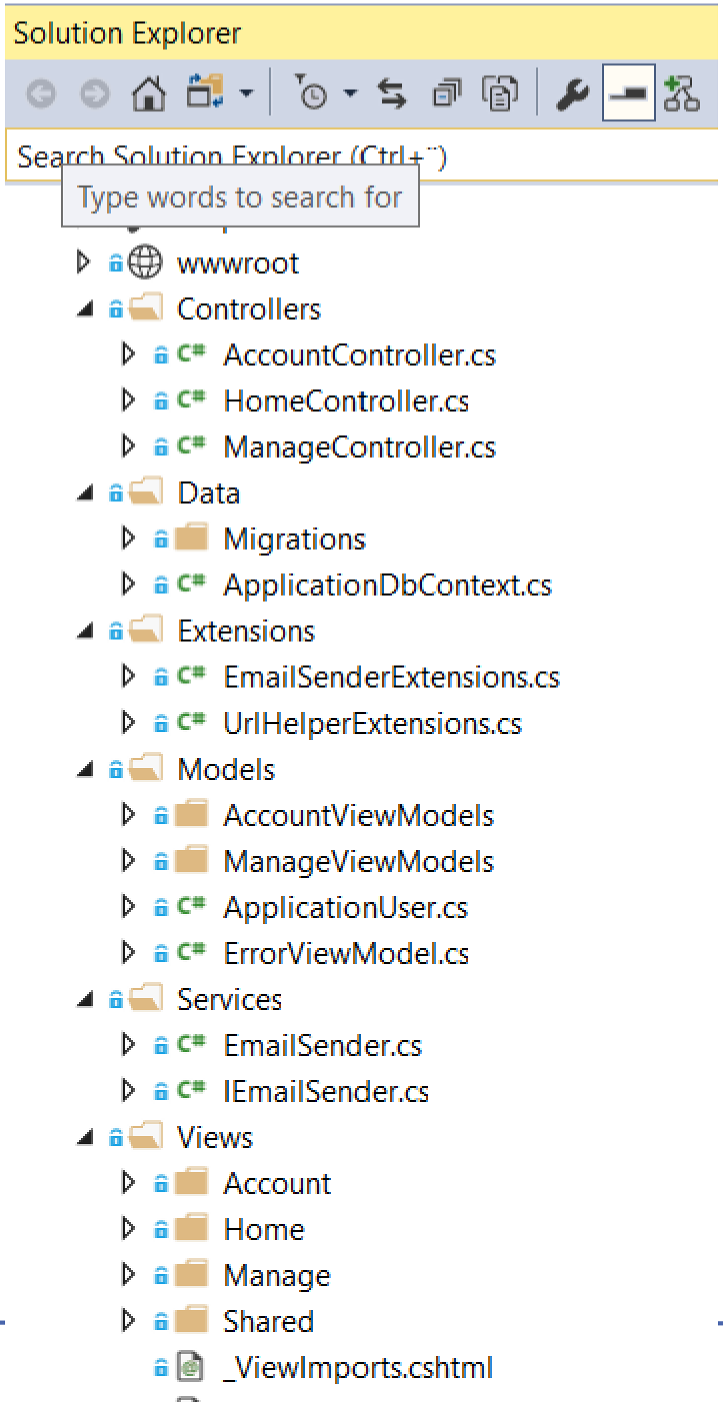
Learn more about third-party open source authentication options

OK | Cancel

AARHUS UNIVERSITY
SCHOOL OF ENGINEERING

The wizard generates a lot of files

- Controllers
  - AccountController for registration, login and logout
  - ManageController for managing the user account
- Data
  - ApplicationDbContext and Migrations
- Extensions
  - Extension for email verification
- Models
  - ApplicationUser
  - Account- and ManageViewModels
- Services
  - EmailSender
- Views
  - Views for AccountController and ManageController



Solution Explorer

Search Solution Explorer (Ctrl+¨)

Type words to search for

- wwwroot
- Controllers
  - AccountController.cs
  - HomeController.cs
  - ManageController.cs
- Data
  - Migrations
  - ApplicationDbContext.cs
- Extensions
  - EmailSenderExtensions.cs
  - UrlHelperExtensions.cs
- Models
  - AccountViewModels
  - ManageViewModels
  - ApplicationUser.cs
  - ErrorViewModel.cs
- Services
  - EmailSender.cs
  - IEmailSender.cs
- Views
  - Account
  - Home
  - Manage
  - Shared
  - _ViewImports.cshtml

Create the Identity Database

- To use EF Core to store user date in MS SQL Server:
  - Add class file called ApplicationDbContext

Application class used to represent users

```
public class ApplicationDbContext : IdentityDbContext<ApplicationUser> {

    public ApplicationDbContext(DbContextOptions<ApplicationDbContext> options)
            : base(options) { }
    }
```

AARHUS UNIVERSITY
SCHOOL OF ENGINEERING

Define the Connection String

- In appsettings.json:

```
{
 "ConnectionStrings": {
    "DefaultConnection": "Server=(localdb)\\mssqllocaldb;Database=aspnet-WebApplication3-A2A2F18E-16B9-4AA0-995B-37232E9AA416;Trusted_Connection=True;MultipleActiveResultSets=true"
  },}
```

You are welcome to rename the database

Configure services

- In the `Startup` class:

```
public void ConfigureServices(IServiceCollection services) {
  services.AddDbContext<ApplicationDbContext>(options =>
    options.UseSqlServer(Configuration.GetConnectionString(
      "DefaultConnection")));

  services.AddIdentity<ApplicationUser, IdentityRole>()
    .AddEntityFrameworkStores<ApplicationDbContext>()
    .AddDefaultTokenProviders();

  // Add application services.
  services.AddTransient<IEmailSender, EmailSender>();
  services.AddMvc();
}
```

AARHUS UNIVERSITY
SCHOOL OF ENGINEERING

Configure middleware

- In the Startup class:

```
public void Configure(IApplicationBuilder app, IHostingEnvironment env)
{
  if (env.IsDevelopment())
  {
    app.UseBrowserLink();
    app.UseDeveloperExceptionPage();
    app.UseDatabaseErrorPage();
  }
  else
  {
    app.UseExceptionHandler("/Home/Error");
  }

  app.UseStaticFiles();
  app.UseAuthentication();
  app.UseMvc(routes =>
    {
      routes.MapRoute(
        name: "default",
        template: "{controller=Home}/{action=Index}/{id?}");
      });
    }
```

Defining the Seed Data (Optionel)

- The IdentitySeedData.cs File

```
public static class IdentitySeedData {
        private const string adminUser = "Admin";
        private const string adminPassword = "Secret123$";

        public static async void EnsurePopulated(IApplicationBuilder app) {

            UserManager<IdentityUser> userManager = app.ApplicationServices
                .GetRequiredService<UserManager<IdentityUser>>();

            IdentityUser user = await userManager.FindByIdAsync(adminUser);
            if (user == null) {
                user = new IdentityUser("Admin");
                await userManager.CreateAsync(user, adminPassword);
            }
        }
    }
```

Apply the Database Migration

- In the Package Manager Console or in Powershell

```
Add-Migration Initial -Context AppIdentityDbContext
```

```
Update-Database -Context AppIdentityDbContext
```

Applying a Basic Authorization Policy

```
using Microsoft.AspNetCore.Authorization;

namespace SportsStore.Controllers {

    public class OrderController : Controller {
        private IOrderRepository repository;
        private Cart cart;

        public OrderController(IOrderRepository repoService, Cart cartService) {
            repository = repoService;
            cart = cartService;
        }

        [HttpPost]
        [Authorize]
        public IActionResult MarkShipped(int orderID) {
```

AARHUS UNIVERSITY
SCHOOL OF ENGINEERING

Applying a Basic Authorization Policy

```
[Authorize]
public class AccountController : Controller {
    private UserManager<IdentityUser> userManager;
    private SignInManager<IdentityUser> signInManager;

    public AccountController(UserManager<IdentityUser> userMgr,
            SignInManager<IdentityUser> signInMgr) {
        userManager = userMgr;
        signInManager = signInMgr;
    }

    [AllowAnonymous]
    public ViewResult Login(string returnUrl) {
    . . .
```

How to require a Role?

Define your roles

```
public enum Role
{
    Student,
    Staff,
    Admin
}
```

```
[Authorize(Roles = "Admin")]
public class ComponentTypeController : Controller
{
```

When initializing `IdentityDbContext`

```
 //Create Roles if they do not exist
 foreach (var role in roles)
 {
     if (!RoleManager.RoleExists(role))
     {
         RoleManager.Create(new IdentityRole(role));
     }
}
```

AARHUS UNIVERSITY
SCHOOL OF ENGINEERING

Oauth Specification

- Is an open standard for authorization
- It specifies a process for resource owners to authorize third-party access to their server resources without sharing their credentials
- Focuses on client developer simplicity while providing specific authorization flows for web applications, desktop applications, mobile phones, and living room devices
- Is being developed within the IETF OAuth WG
- **OAuth is commonly used as a way for Internet users to log into third party websites using their Microsoft, Google, Facebook or Twitter accounts without exposing their password**
- The OAuth 2.0 framework was published as RFC 6749 in October 2012

References & Links

- Secure a Web Api in ASP.NET Core
  http://www.blinkingcaret.com/2017/09/06/secure-web-api-in-asp-net-core/
- **Identity by K. Scott Allen**
  http://pluralsight.com/training/Player?author=scott-allen&name=aspdotnet-mvc5-fundamentals-m6-ef6&mode=live&clip=0&course=aspdotnet-mvc5-fundamentals
- OAuth
  http://oauth.net/2/