Bringing in Authentication and Authorization

Gill Cleeren
CTO Xpirit Belgium

@gillcleeren – xpirit.com/gill

Module overview

Understanding ASP.NET Core Identity

Adding authentication to the site

Using authorization

Understanding ASP.NET Core Identity

Introducing ASP.NET Core Identity

- Membership system
- Supports UI login functionality
- Authentication and authorization
- Supports external providers

Introducing ASP.NET Core Identity

- Manage users, roles, claims, passwords, email confirmation...
- Scaffolding support

Adding ASP.NET Identity

- SQL Server support built-in
  - Can work with other databases too
  - IdentityDbContext

```
builder.Services.AddDefaultIdentity<IdentityUser>()
    .AddEntityFrameworkStores<BethanysPieShopDbContext>();
```

Adding Support for ASP.NET Core Identity

```
builder.Services.AddDefaultIdentity<IdentityUser>(options =>
{
    options.Password.RequireDigit = true;
    options.Password.RequiredLength = 8;
    options.Password.RequireNonAlphanumeric = true;
    options.User.RequireUniqueEmail = true;
})
.AddEntityFrameworkStores<BethanysPieShopDbContext>();
```

Configuration Options

Password length, user options...

Demo

Adding ASP.NET Core Identity to the application

Adding Authentication to the Site

Manual approach

- Specific controller with Login, Logout... actions
- Specific views

Scaffolding

- Visual Studio tooling or through scaffolder for CLI
- Include items to project to customize, otherwise default is used
- Based on Razor Class Library

Razor Class Library

- Reuse functionality
  - Views, pages, controllers, view components...
- Can be overridden

Steps to Add Authentication

Use built-in RCL

Scaffold item(s)

Layout changes

Important Classes in ASP.NET Core Identity

UserManager<IdentityUser>

SignInManager<IdentityUser>

Demo

Scaffolding Login and Register functionality

Layout changes

Using Authorization

```
[Authorize]
public class OrderController : Controller
{
}
```

Authorizing Users

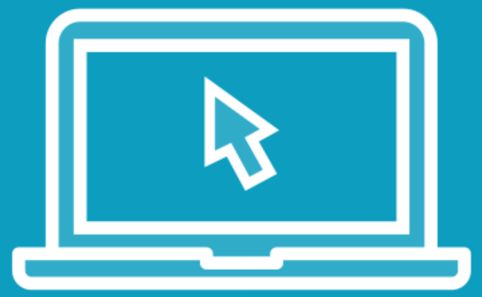
```
public class OrderController : Controller
{
    [Authorize]
    public IActionResult Checkout()
    { }
}
```

Action Level

```
[Authorize(Roles = "Administrator")]
public class OrderController : Controller
{
}
```

Combining with Roles

Demo

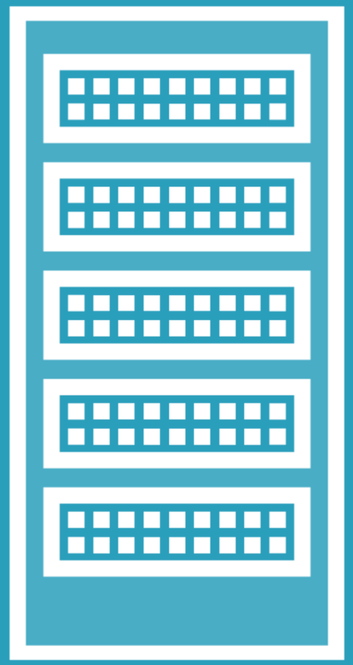


Adding authorization

Summary

ASP.NET Core Identity allows us to perform authentication

Using [Authorize], we can restrict access to resources



Up next:
Deploying our site to Azure