

Adding security using ASP.NET Identity



Gill Cleeren

ARCHITECT

@gillcleeren www.snowball.be



Overview



Introducing ASP.NET Core Identity

Configuring the application for Identity

Adding the login page

Adding authorization



Introducing ASP.NET Identity



ASP.NET Core Identity is a membership system which allows us to manage users and create login functionality



ASP.NET Core Identity

Authentication and
authorization

Membership-based

Support for external providers

SQL Server support



Configuring the Application for Identity



```
{  
  "dependencies": {  
    ...  
    "Microsoft.AspNetCore.Identity.EntityFrameworkCore": "1.0.0"  
  }  
}
```

Adding the Required Package



```
public class AppDbContext : IdentityDbContext<IdentityUser>
{
    public AppDbContext(DbContextOptions<AppDbContext> options)
        : base(options)
    {
    }
}
```

Creating the Identity Database




```
public class PieShopUser : IdentityUser {  
    //other properties can be added here  
}
```

Working with Our Own User Class



Configuration Changes

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

    services.AddIdentity<IdentityUser, IdentityRole>()
        .AddEntityFrameworkStores<AppDbContext>();
}
```



Configuration Changes

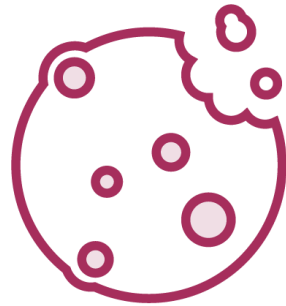
```
public void Configure(IApplicationBuilder app,  
    IHostingEnvironment env, ILoggerFactory loggerFactory)  
{  
    app.UseDeveloperExceptionPage();  
    app.UseStatusCodePages();  
    app.UseStaticFiles();  
  
    app.UseSession();  
    app.UseIdentity();  
}
```



Further Configuration Options



Password



Cookies



User



Lockout



Configuration Options

```
services.Configure<IdentityOptions>(options =>
{
    options.Password.RequireDigit = true;
    options.Password.RequiredLength = 8;
    options.Password.RequireNonAlphanumeric = true;
    options.User.RequireUniqueEmail = true;
});
```



Demo



Adding ASP.NET Core Identity

Configuring ASP.NET Core Identity



Adding the Login Page



Allowing Users to Log In

Login view

Account controller

Model changes



AccountController

```
public class AccountController : Controller
{
    public IActionResult Login(string returnUrl) {}
    public async Task<IActionResult>
        Login(LoginViewModel loginViewModel) {}

    public IActionResult Register() {}
    public async Task<IActionResult> Logout() {}
}
```



Important Classes in Identity

UserManager<IdentityUser>

SignInManager<IdentityUser>



LoginViewModel Class

```
public class LoginViewModel
{
    [Required]
    [Display(Name="User name")]
    public string UserName { get; set; }

    [Required]
    [DataType(DataType.Password)]
    public string Password { get; set; }
}
```



Demo



Adding the AccountController

Adding the Login View



Adding Authorization



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

The Authorize Attribute



```
[AllowAnonymous]  
public IActionResult Login(string returnUrl)  
{  
}
```

Allowing Access to Specific Methods



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

Adding Roles



Demo



Adding authorization to our site



Summary



ASP.NET Identity is a very complete package

- Authentication
- Authorization

Can be configured to work together with EFCore

