



# What's new with ABP 8.0?



Engincan VESKE

Software Engineer at Volosoft



# ABP Framework 8.0 Highlights

<https://abp.io/b/v80>

- Upgraded to .NET 8.0
- Upgraded to Angular 17
- Dynamic Claims
- CDN Support for Bundling & Minification system
- Read-Only Repositories: Don't track changes by default
- Account Module: Set username after social login

# Upgraded to .NET 8.0

## Migration Guides

- [Microsoft's Migrate from ASP.NET Core 7.0 to 8.0 documentation](#)
- [ABP Framework 7.x to 8.0](#)
- [ABP Commercial 7.x to 8.0](#)

# Upgraded to Angular 17

- Angular 17 [has been released on November 8](#)
- ABP 8.0 Angular UI is based on Angular 17

# Dynamic Claims

**Dynamic Claims** allows you to **get the latest user claims** without need to re-login in each time.

The **Dynamic Claims** feature is used to dynamically generate claims for the user in each request. This was needed because claims-based authentication is using in the ASP.NET Core, and because it stores the claims in the cookie or token, and they are being static, they don't change until a next re-login.

**It's already been documented:** <https://docs.abp.io/en/abp/8.0/Dynamic-Claims>

# Dynamic Claims

Configure the **AbpClaimsPrincipalFactoryOptions** and set the **IsDynamicClaimsEnabled** option as true 🙌

Add the **DynamicClaims** middleware into the request pipeline 🙌

```
public override void ConfigureServices(ServiceConfigurationContext context)
{
    context.Services.Configure<AbpClaimsPrincipalFactoryOptions>(options =>
    {
        options.IsDynamicClaimsEnabled = true;
    });
}
```

```
public override void OnApplicationInitialization(ApplicationInitializationContext context)
{
    // Add this line before UseAuthorization.
    app.UseDynamicClaims();
    app.UseAuthorization();
    //...
}
```

# CDN Support for Bundling & Minification System

```
Configure<AbpBundlingOptions>(options =>
{
    options.StyleBundles
        .Add( bundleName: "MyStyleBundles", configureAction: config: BundleConfiguration =>
            {
                config
                    .AddFiles("/styles/my-style-1.css")
                    .AddFiles("/styles/my-style-2.css")
                    .AddFiles("https://cdn.abp.io/bootstrap.css");
            });

    options.ScriptBundles
        .Add( bundleName: "MyScriptBundles", configureAction: config: BundleConfiguration =>
            {
                config
                    .AddFiles("/scripts/my-script-1.js")
                    .AddFiles("/scripts/my-script-2.js")
                    .AddFiles("https://cdn.abp.io/bootstrap.js");
            });
});
```

In this version on, ABP Framework's MVC / Razor Pages UI's [Bundling System](#) provides **CDN Support**.

The system automatically recognizes the **external/CDN files** and places them as script tags into the page along with the bundled CSS/JS files.

```
<link rel="stylesheet" href="/__bundles/MyStyleBundles.EA8C28419DCA43363E9670973D4C0D15.css?v=638331889644609730" />
<link rel="stylesheet" href="https://cdn.abp.io/bootstrap.css" />

<script src="/__bundles/MyScriptBundles.C993366DF8840E08228F3EE685CB08E8.js?v=638331889644937120"></script>
<script src="https://cdn.abp.io/bootstrap.js"></script>
```



# Read-Only Repositories: Don't track changes by default

```
public class MyService
{
    private readonly IRepository<Book, Guid> _bookRepository;
    private readonly IReadOnlyRepository<Book, Guid> _bookReadOnlyRepository;

    public async Task MyMethod()
    {
        //ReadOnly Repository -> GetListAsync(), GetAsync() vs. read-only methods
        //not change tracking involved
        var books:List<Book> = await _bookReadOnlyRepository.GetListAsync();

        //Repository -> All methods including InsertAsync(), UpdateAsync() etc.
        var newBook = await _bookRepository.InsertAsync(entity: new Book());
        var newBook2 = await _bookReadOnlyRepository.InsertAsync(new Book()); //not exists
    }
}
```

In this version, ABP Framework provides read-only repository interfaces (**IReadOnlyRepository<T>** or **IReadOnlyBasicRepository<T>**) to explicitly indicate that your purpose is to query data, but not change it.



# Account Module: Set username after social login

Email 'admin@abp.io' is already taken. ×

TENANT  
Not selected switch

## Register

Already registered? [Login](#)

Username \*

Email address \*

Register

# Other News

- **LDAP over SSL (LDAPS)** setting has been added and recommended to establish a secure connection.
- **Object Mapping Enhancements** (supports mapping collection of objects for custom object mappers)

Once you implement `IObjectMapper<User, UserDto>`, ABP can automatically convert a collection of `User` objects to a collection of `UserDto` objects. The following generic collection types are supported:

- `IEnumerable<T>`
- `ICollection<T>`
- `Collection<T>`
- `IList<T>`
- `List<T>`
- `T[]` (array)

- **Sending attachments** with `ISender.QueueAsync()` method

# ABP Commercial 8.0 Highlights

- Suite: Generating **Master/Detail** Relationship
- Get **profile picture** from **social/external logins**
- Switch from **Ocelot** to **YARP** for the API Gateway (Microservice Solution Template)
- **Password complexity indicators** for MVC & Blazor UIs
- **Export & Import** Users as Excel/CSV

# Suite: Generating Master/Detail Relationship

The screenshot displays a web application interface for managing orders. The left sidebar shows a navigation menu with options: Home, Dashboard, SaaS, Administration, Products, and Orders. The main content area is titled 'Orders' and includes a search bar and an 'Advanced filters' link. Below this, there are two main sections:

- Master / Parent Grid:** This section displays a table of orders. It has columns for 'ACTIONS', 'NAME', and 'CODE'. The table contains two rows: one for 'ORD-2' with code '10644' and another for 'ORD-1' with code '10643'. Each row has an 'Actions' button.
- Child / Detail Grid:** This section displays a table of order lines for the selected order (ORD-1). It has columns for 'ACTIONS', 'QUANTITY', 'TOTAL PRICE', and 'PRODUCT'. The table contains five rows of order lines, each with an 'Actions' button.

At the bottom of the page, there is a footer with the text '2023 © Lepton Theme by Volosoft' and a language selector set to 'EN'. The bottom right corner features a logo for '10 YEARS OF abp'.

ACTIONS	NAME	CODE
Actions	ORD-2	10644
Actions	ORD-1	10643

ACTIONS	QUANTITY	TOTAL PRICE	PRODUCT
Actions	2	\$ 28.59	Book - Social Animal: A Story of How Success Happens
Actions	1	\$ 21.09	Book - Mastering ABP Framework: Build maintainable .NET solutions by implementing software development best practices
Actions	1	\$ 44.99	Book - Domain-Driven Design: Tackling Complexity in the Heart of Software
Actions	4	\$ 120.00	Book - Clean Code: A Handbook of Agile Software Craftsmanship
Actions	2	\$ 40.20	Book - Refactoring: Improving the Design of Existing Code



# THANKS FOR WATCHING

