XMAS DEV 2025

Missione Natale Protetto: Autenticazione moderna in Blazor con OpenID Connect, Passwordless e FIDO2

Andrea Dottor

Platinum Sponsor

Microsoft

Gold Sponsor

BC Soft
consulenza sistemi informativi

Aenduo
Get health connected

Technical Sponsor

VILLANOVA

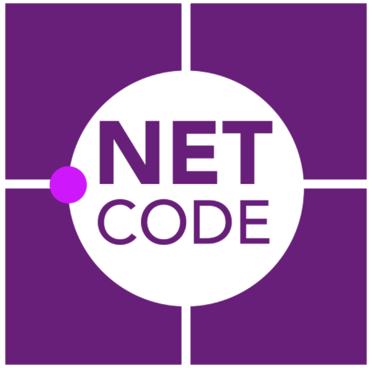
MAGNETIKODE

JET BRAINS

Packt

XMASDEV
<OH>OH</OH>

Community


NET
CODE

DotNet
Toscana

DotNetAbruzzo

dotnet
LOMBARDIA

COOKIES, IDENTITY, AND EXTERNAL IDENTITY PROVIDERS

ASP.NET Core & Blazor Server

- Authentication handled entirely **server-side**
- Uses ASP.NET Core authentication middleware
- Cookie authentication is the natural approach

Standalone Blazor WebAssembly

- Runs **client-side** in the browser
- Must use OIDC / OAuth2
- Works with real Identity Providers (Keycloak, Azure Entra ID, Auth0…)

CLAIMS-BASED AUTHORIZATION

Claims-based authorization is the core security model in ASP.NET Core and Blazor. It grants access based on information contained in the user's identity, not just roles.

A claim is a key-value pair describing something about the user

Decouple authorization from identity provider structure

ROLES-BASED AUTHORIZATION

Roles represent high-level permissions assigned to users.

Roles tell who the user is.

- Simple: "*Admin*", "*Manager*", "*User*"

⚠️ Limitations of Roles

- Not granular enough for complex scenarios
- Hard to scale when many permissions are needed

AUTHORIZATION POLICIES

Authorization Policies allow you to define rules that determine when and how a user is authorized.

Policies define *what* the user is allowed to do.

Policies operate on claims, roles, or custom requirements.

Common Policies:

- Require a claim
- Require multiple claims
- Require a role
- Combine requirements
- Custom rule with IAuthorizationHandler

LA CASETTA DEGLI ELFI (PROTEZIONE SEMPLICE)

COOKIE AUTHENTICATION

- Server maintains state, HttpOnly cookies
- Fully compatible with claims-based auth
- ✔️ Pros: simple, built-in, minimal setup

XMASDEV

<oh>oh</oh>

DEMO: SIMPLE AUTHENTICATION

L'UFFICIO POSTALE DEL POLO NORD (CONTROLLI INTERMEDI)

WHY ASP.NET CORE IDENTITY?

- Full-featured membership system: users, roles, 2FA, lockout, password resets
- UserManager and SignInManager
- Claims-based architecture
- Writes authentication cookies automatically
- Easy to extend with additional fields

IDENTITY PROS & CONS

✓ Pros

- Robust, secure, feature-rich
- Easy integration with cookie auth
- Supports external logins

✗ Cons

- Heavy if you only need simple auth
- Tightly coupled to EF Core (default)

ENABLE WEB AUTHENTICATION API (WEBAUTHN) PASSKEYS

- Passkeys provide a modern, phishing-resistant authentication method based on the Web Authentication API (WebAuthn) and FIDO2 standards. They are a secure alternative to passwords, using public key cryptography and device-based authentication.

ENABLE WEB AUTHENTICATION API (WEBAUTHN) PASSKEYS

- Passkeys provide a modern, phishing-resistant authentication method based on the Web Authentication API (WebAuthn) and FIDO2 standards. They are a secure alternative to passwords, using public key cryptography and device-based authentication.
- The **private key is stored securely on the user's device**, such as in a hardware security module, platform authenticator (examples: Windows Hello, Touch ID, Face ID), or a password manager, while the **public key is stored by the web app**.
- Key benefits of passkeys include:
  - **Phishing resistance**: Passkeys are bound to specific websites and can't be used on fake sites.
  - **No shared secrets**: The server only stores public keys, eliminating the risk of password database breaches.
  - **User convenience**: Simple biometric or PIN verification replaces complex password requirements.
  - **Cross-device synchronization**: Many passkey providers sync credentials across a user's devices.

DEMO: ASP.NET CORE IDENTITY

IL LABORATORIO SEGRETO DI BABBO NATALE (MASSIMA SICUREZZA)

WHY USE AN EXTERNAL IDENTITY PROVIDER?

- Single Sign-On across multiple applications
- Industry-standard authentication
- No password handling inside the application
- App receives ID token and access token
- Token is stored server-side
- Uses ASP.NET Core OIDC middleware

- Some well-known providers commonly used:
  - Keycloak
  - WSO2 Identity Server
  - Azure Entra ID
  - Amazon Cognito
  - Auth0
  - …

KEYCLOAK OVERVIEW

- Keycloak is an open-source Identity and Access Management (IAM) solution built for modern applications and services.
  It provides authentication, authorization, and user management without requiring custom security code.
- 🔑 Key Features
  - OpenID Connect & OAuth2 support
  - Single Sign-On (SSO) for multiple applications
  - User federation (LDAP, Active Directory)
  - Built-in login and consent pages
  - Role-based and attribute-based access control
  - Password policies, 2FA, brute-force protection
  - Administration UI + REST APIs

REALMS, CLIENTS, ROLES, USERS

- 🏰 Realms
  - A realm is an isolated security domain.
  - Each realm manages:
    - its own usersits own roles
    - its own clients
    - its own authentication settings
- 🧩 Clients
  - Applications that use Keycloak for authentication:
    - Blazor WASM / Blazor Server apps
    - Web APIs
    - SPA, mobile apps, backend services
  - Clients define:
    - Allowed redirect URIs
    - OIDC flow (Authorization Code, Hybrid…)
    - Client roles
    - Token settings (lifetime, signature)

REALMS, CLIENTS, ROLES, USERS

- 🎭 Roles
  - Roles define what a user can do.
    Two types:
    - Realm Roles
      Available globally across the realm.
    - Client Roles
      Scoped to a specific application/client.Useful for per-app permissions.
- 👤 Users
  - Users represent people or service accounts.
    A user contains:
    - Credentials (password, OTP, WebAuthn)
    - Profile attributes
    - Assigned roles
    - Groups (hierarchical role assignment)

IDENTITY BROKERING

- Identity Brokering allows Keycloak to act as an intermediary between your application and external identity providers:
  - Microsoft, Google, GitHub, Facebook
  - Microsoft Entra ID
  - SAML providers
  - Any OIDC-compliant IdP

- 🔑 What It Does
  - Delegates authentication to external providers
  - Normalizes identities into Keycloak users
  - Issues ID Token / Access Token to your apps
  - Supports both OIDC and SAML identity providers

XMASDEV

<oh>oh</oh>

DEMO: KEYCLOAK

FINAL LESSONS LEARNED

- Cookie Authentication remains the simplest for server-side scenarios
- ASP.NET Core Identity is a complete framework for user management
- Keycloak enables SSO, federation, MFA, and scalable auth flows
- Everything is claims-based in ASP.NET Core
- Use well-designed authorization rules, not custom logic in components

XMASDEV
<OH>OH</OH>

GRAZIE

```
{
     name:          Andrea Dottor
     email:         andrea@dottor.net
     bsky:          @andrea.dottor.net
     linkedin:      @andreadottor
}
```

@andreadottor

.NET IN PILLOLE
il podcast

Microsoft® Most Valuable Professional

XMASDEV

<OH>OH</OH>

PLEASE VOTE FOR THIS SESSION

Vote online at:

https://vote.dotnetdev.it/vote/xqt335vh/1022024

XMASDEV

<oh>oh</oh>

QUESTIONS & DISCUSSION