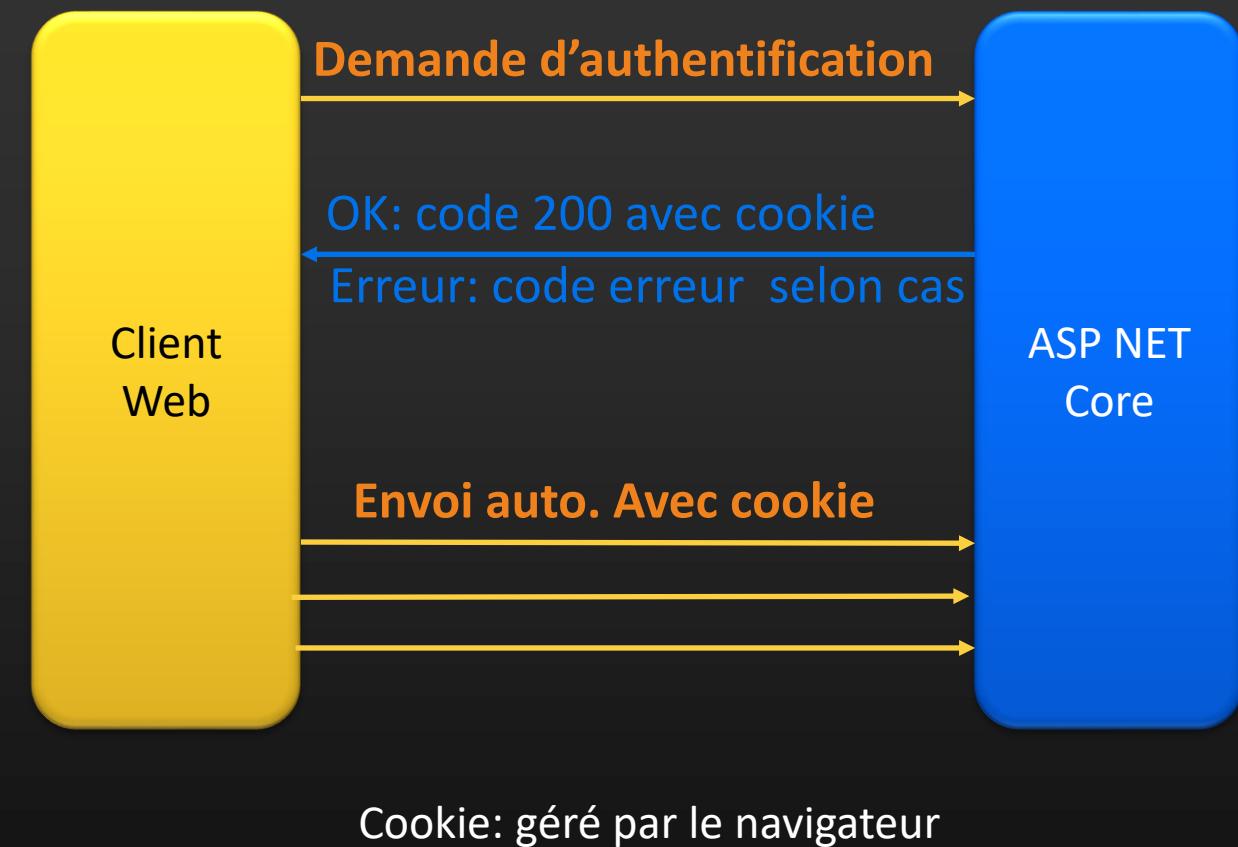


Programmation Web Transactionnelle

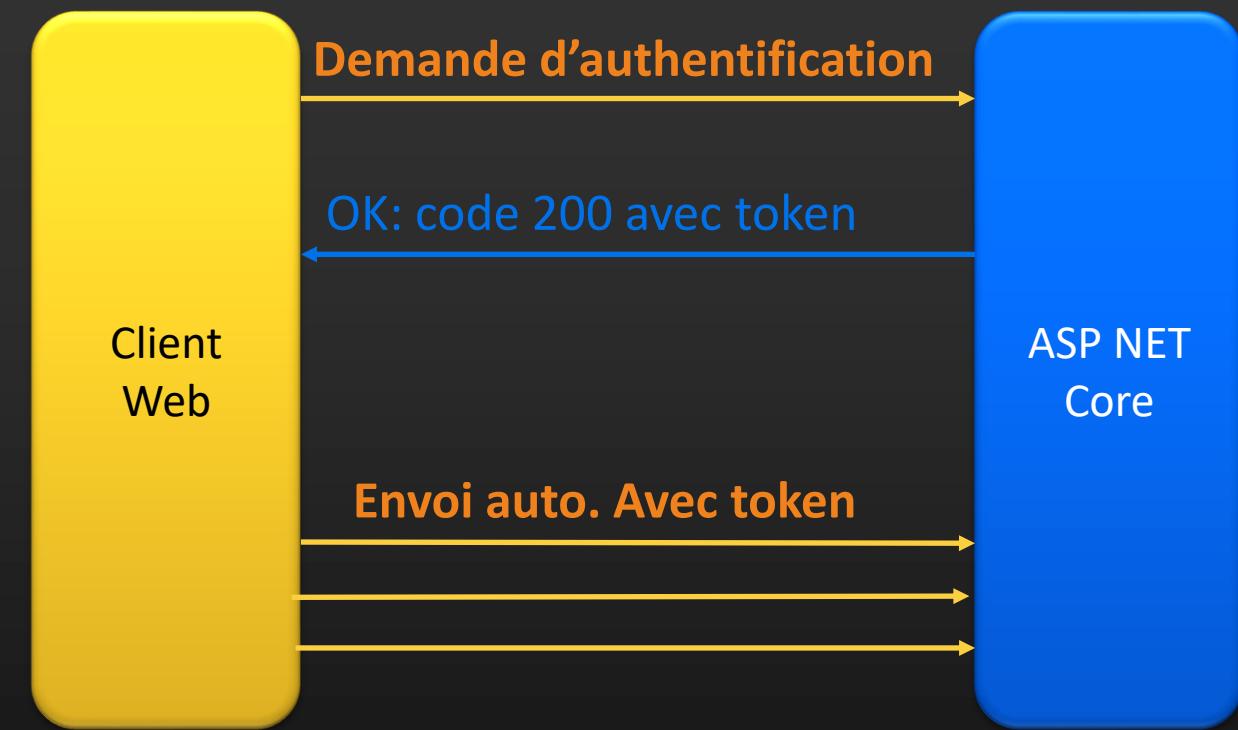
**Authentication
MS Identity**



Authentification: Cookie

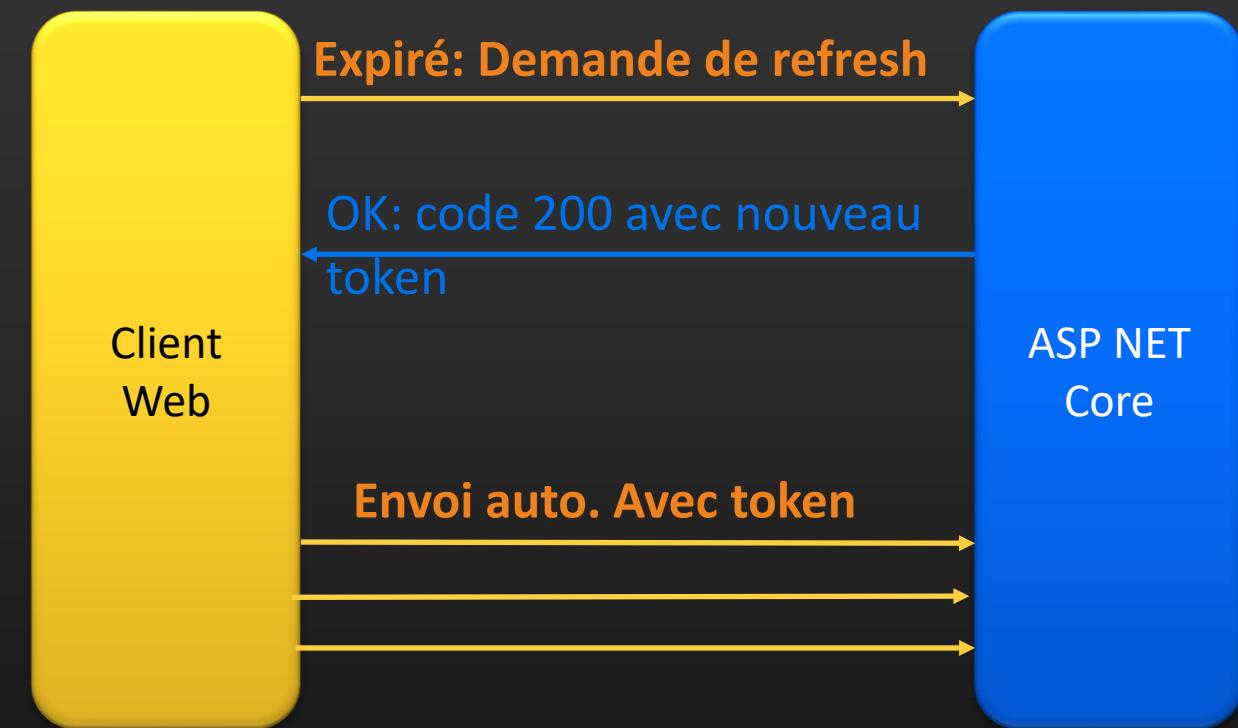


Authentification: Jeton (*Token bearer*)



token: géré par l'application (client responsable d'envoyer le token)
norme JWT (Jason Web Token)

Authentification: Durée de vie du Token



MS Identity: une librairie qui...

- ⚠ Contient les informations et code requis pour la connexion
- ⚠ Contient les modèles, views et appels => créer, modifier et identifier les utilisateurs
- ⚠ Contient l'implantation de l'attribut [Authorize] permettant de sécuriser un contrôleur ou des actions
 - ⚠ Valider l'utilisateur et les rôles associés (*détails prochaine séance*)



La sécurité: Autorisations

- Rendre accessible certaines parties de l'application/pages
- Basées sur des **rôles** et des **revendications** (*claims*) ou **caractéristiques** de l'utilisateur
- Les **privileges** doivent être octroyés via une interface réservée aux administrateurs.

Détails prochaine séance

Architecture MS Identity

- ⚠ gestionnaires (ou des **managers**): repositories pour les modèles contenus.
- ⚠ **UserManager** : chercher, créer, modifier, supprimer des utilisateurs
- ⚠ **SignInManager** : connexion, la création/suppression de cookie, etc.
- ⚠ **RoleManager** chercher, créer, modifier, supprimer des rôles.

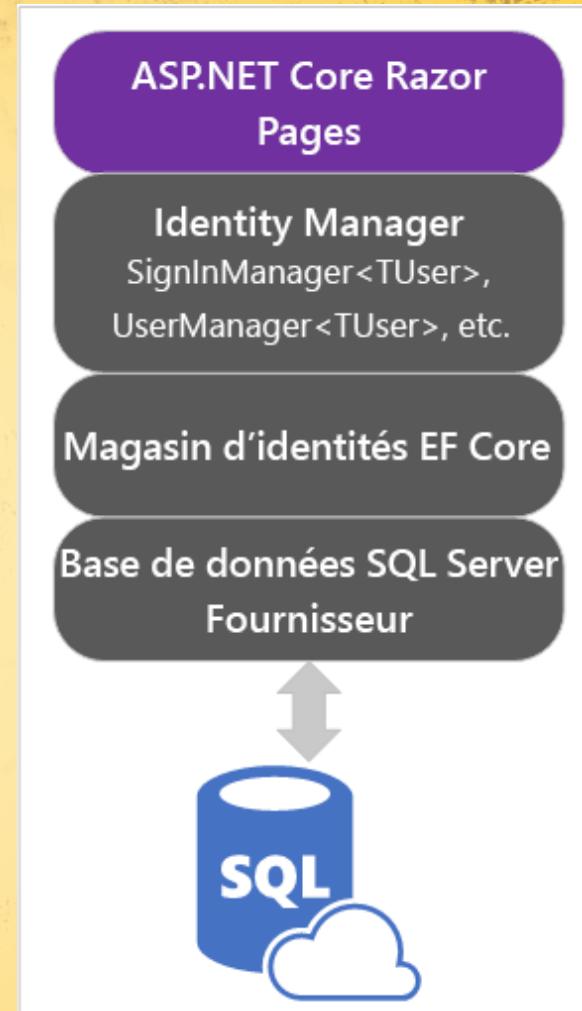
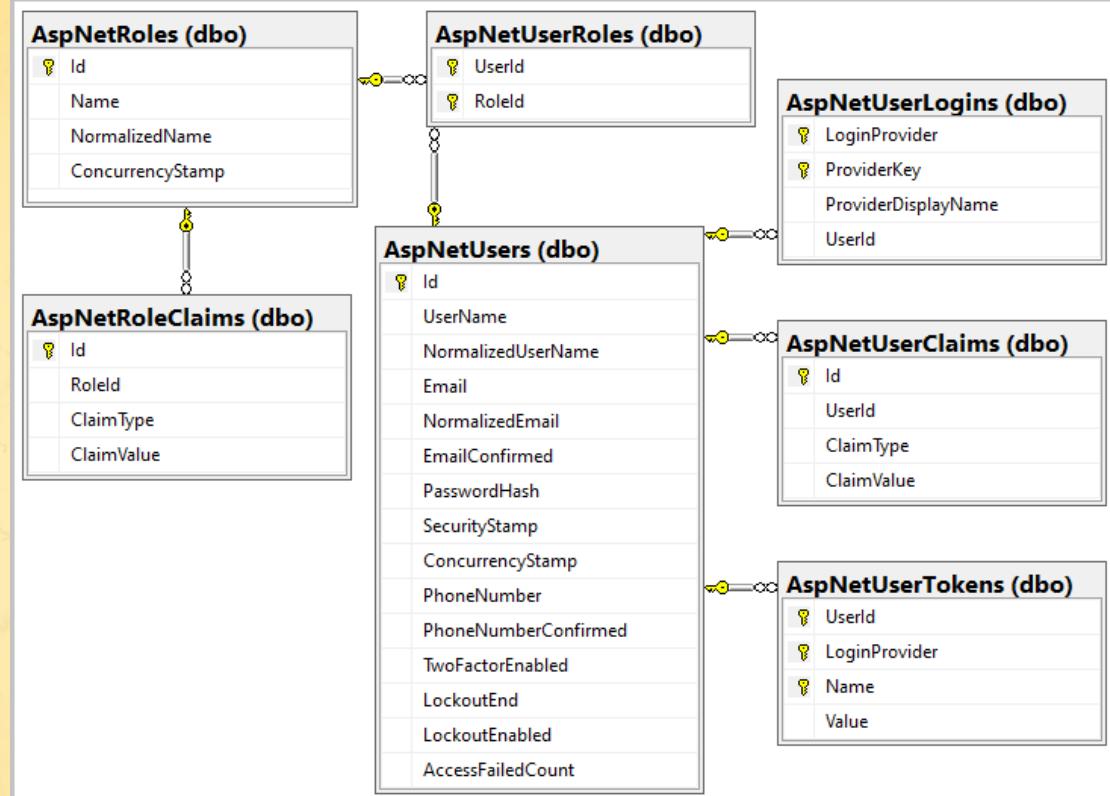


Schéma BD MS Identity

- MS Identity est fourni en différents modèles.
- L'utilisateur, qui contient ses informations
- Les rôles, pour différencier Admin de User
- Les UserLogins qui permet de se connecter par un fournisseur externe
- Les revendications (claims) clés-valeurs à l'utilisateur. (fréquemment fournisseur externe ou de politique)



Mise en place: Identity

Classe ApplicationUser

- utilisateur héritant de *IdentityUser*
- Courriel, mot de passe haché, etc.
- Propriétés pour validation

ApplicationDbContext

- Héritant de *IdentityDbContext* (tables)

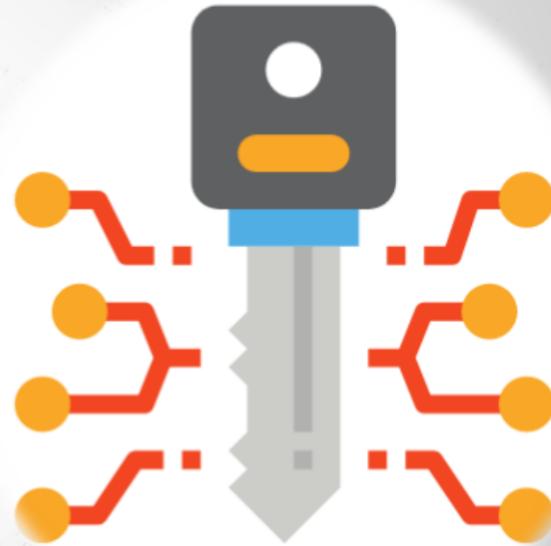
DefaultTokenProviders

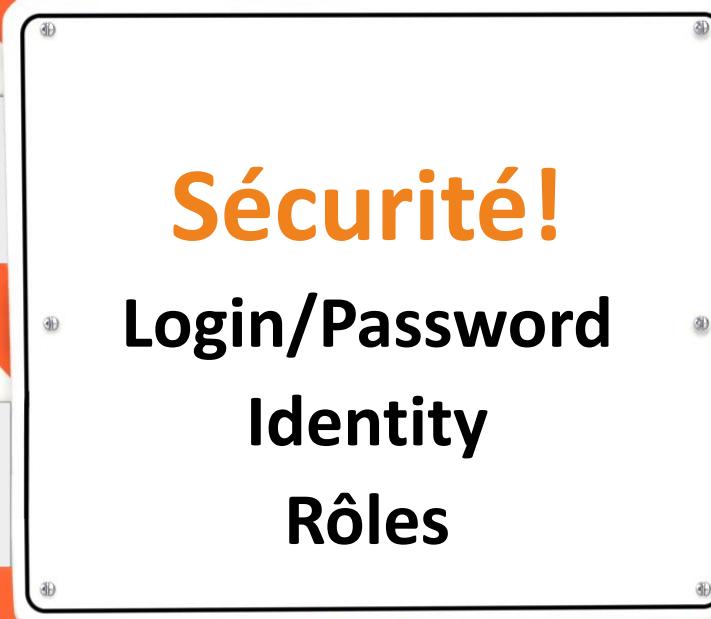
- Mécanismes tel que validation par courriel, etc

Non présenté
dans ce cours

Identity: **userManager<TUser>**

- Gestion des utilisateurs
- Gestion des mots de passe
 - Hash
 - Vérification
- Gestion des *Tokens*
- Gestion multiComptes
- Gestion des rôles





Sécurité!
Login/Password
Identity
Rôles



Installation des packages requis

- Microsoft.AspNetCore.Identity.EntityFrameworkCore

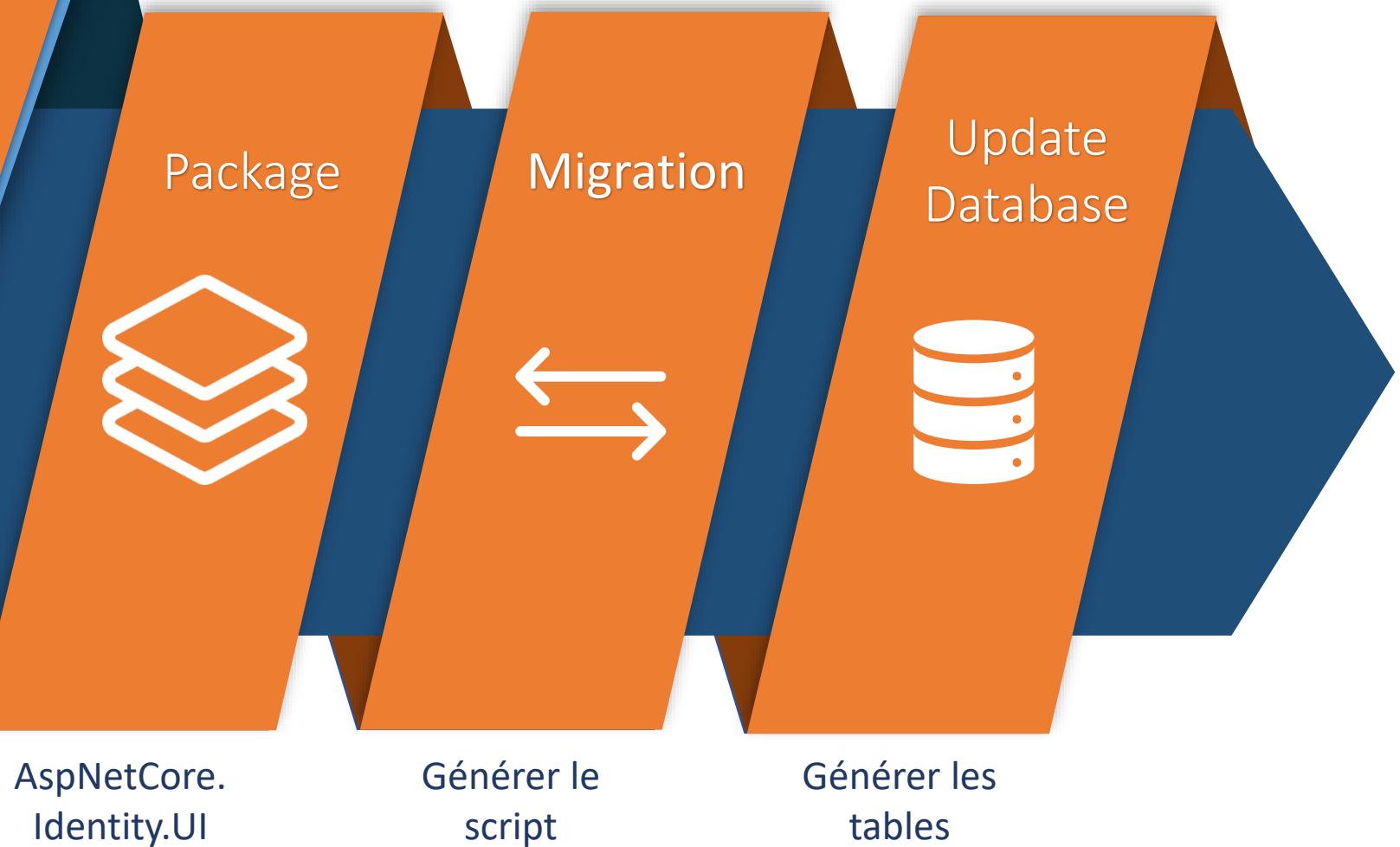
- Microsoft.AspNetCore.Identity.UI

- Microsoft.Extensions.Identity.Stores

- Automatiquement

- Si problème dans votre version
- Installation du package via PowerShell
 - `dotnet tool install --global dotnet-aspnet-codegenerator --version 6.0`
 - `dotnet aspnet-codegenerator identity –useDefaultUI`

Identity autogénéré



- Authentification

User



- Regroupement de users

Rôle



- Informations du user, ce qu'il est. Guide les autorisations

UserClaim



- Système de confirmation de l'authentification

UserToken



- Association qui fait quoi

UserRole



- Informations du role, ce qu'il est. guide les autorisations

RoleClaim



ApplicationContext

```
using Microsoft.AspNetCore.Identity.EntityFrameworkCore;
using Microsoft.EntityFrameworkCore;
using ZombieParty_SOL_Models;
...
public class ZombiePartyDbContext:IdentityDbContext
{
    public
ZombiePartyDbContext(DbContextOptions<ZombiePartyDbContext>
options) : base(options)
    {
    }
}
```

Login Partial View
AccessDenied
ForgotPassword
ResendEmailConfirmation
Etc...

Identity

Scaffolded Pages

Log in

Use a local account to log in.

Email

Password

Remember me?

Log in

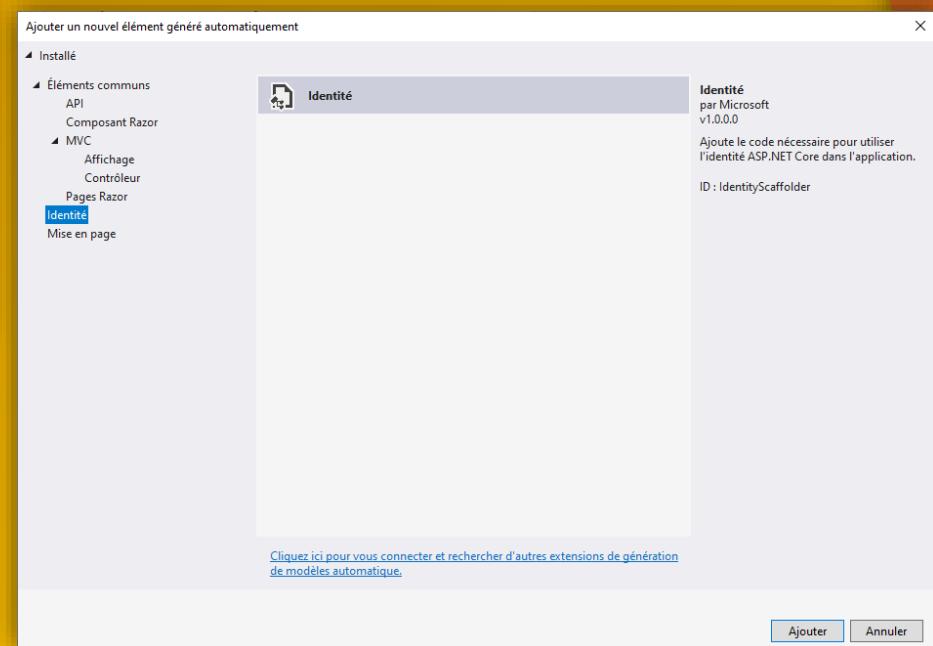
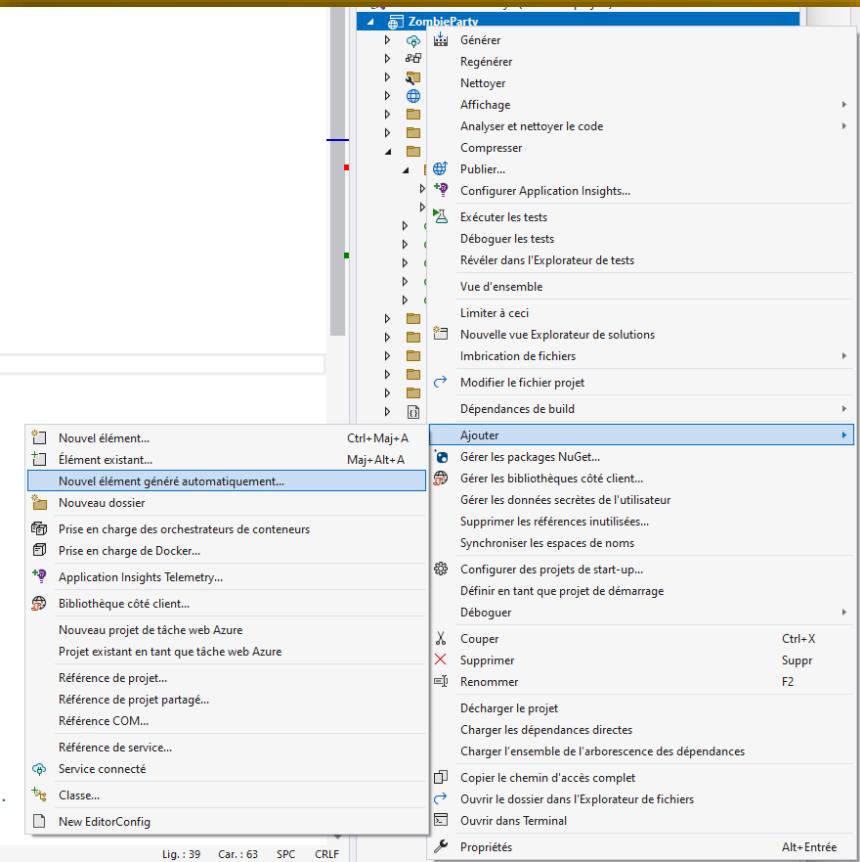
[Forgot your password?](#)

[Register as a new user](#)

[Resend email confirmation](#)

Use another service to log in.

There are no external authentication services configured. See this [article about setting up this ASP.NET application to support logging in via external services](#).



Sélectionner les pages Razor désirées

Ajouter Identité

Sélectionnez une page de disposition existante ou spécifiez-en une nouvelle :

/Areas/Identity/Pages/Account/_Layout.cshtml

(Laissez vide s'il est défini dans un fichier Razor _viewstart)

Remplacer tous les fichiers

Choisir les fichiers à remplacer

<input checked="" type="checkbox"/> Account\StatusMessage	<input checked="" type="checkbox"/> Account\AccessDenied	<input checked="" type="checkbox"/> Account\ConfirmEmail
<input checked="" type="checkbox"/> Account\ConfirmEmailChange	<input checked="" type="checkbox"/> Account\ExternalLogin	<input checked="" type="checkbox"/> Account\ForgotPassword
<input checked="" type="checkbox"/> Account\ForgotPasswordConfirmation	<input checked="" type="checkbox"/> Account\Lockout	<input checked="" type="checkbox"/> Account>Login
<input checked="" type="checkbox"/> Account>LoginWith2fa	<input checked="" type="checkbox"/> Account>LoginWithRecoveryCode	<input checked="" type="checkbox"/> Account\Logout
<input checked="" type="checkbox"/> Account\Manage\Layout	<input checked="" type="checkbox"/> Account\Manage\ManageNav	<input checked="" type="checkbox"/> Account\Manage\StatusMessage
<input checked="" type="checkbox"/> Account\Manage\ChangePassword	<input checked="" type="checkbox"/> Account\Manage\DeletePersonalData	<input checked="" type="checkbox"/> Account\Manage\Disable2fa
<input checked="" type="checkbox"/> Account\Manage\DownloadPersonalData	<input checked="" type="checkbox"/> Account\Manage\Email	<input checked="" type="checkbox"/> Account\Manage\EnableAuthenticator
<input checked="" type="checkbox"/> Account\Manage\ExternalLogins	<input checked="" type="checkbox"/> Account\Manage\GenerateRecoveryCodes	<input checked="" type="checkbox"/> Account\Manage\Index
<input checked="" type="checkbox"/> Account\Manage\PersonalData	<input checked="" type="checkbox"/> Account\Manage\ResetAuthenticator	<input checked="" type="checkbox"/> Account\Manage\SetPassword
<input checked="" type="checkbox"/> Account\Manage>ShowRecoveryCodes	<input checked="" type="checkbox"/> Account\Manage\TwoFactorAuthentication	<input checked="" type="checkbox"/> Account\Register
<input checked="" type="checkbox"/> Account\RegisterConfirmation	<input checked="" type="checkbox"/> Account\ResendEmailConfirmation	<input checked="" type="checkbox"/> Account\ResetPassword
<input checked="" type="checkbox"/> Account\ResetPasswordConfirmation		

Classe de contexte de données :

Utiliser SQLite au lieu de SQL Server

Classe utilisateur :

Si vide: voir diapo bogue.

Bogue Identity autoGénéré

Sélection du
DbContext vide

Dans le DbContext ajouter:

```
public class ZombiePartyDbContext : IdentityDbContext
{
    public
ZombiePartyDbContext(DbContextOptions<ZombiePartyDbContext> options) :
base(options)
    {

    }

protected override void OnModelCreating(ModelBuilder modelBuilder)
{
    base.OnModelCreating(modelBuilder);

    // Votre code pour le seed
}
```

Register

Create a new account.

Use another service to register.

Email

There are no external authentication services configured. See this [article about setting up this ASP.NET application to support logging in via external services.](#)

Password

Confirm password

Log in

Use a local account to log in.

Use another service to log in.

Email

There are no external authentication services configured. See this [article about setting up this ASP.NET application to support logging in via external services.](#)

Password

Remember me?

[Forgot your password?](#)

[Register as a new user](#)

[Resend email confirmation](#)

Areas	
Identity	
Pages	
Account	
Manage	
_StatusMessage.cshtml	
_ViewImports.cshtml	
+ AccessDenied.cshtml	
+ C# AccessDenied.cshtml.cs	
+ ConfirmEmail.cshtml	
+ C# ConfirmEmail.cshtml.cs	
+ ConfirmEmailChange.cshtml	
+ C# ConfirmEmailChange.cshtml.cs	
+ ExternalLogin.cshtml	
+ C# ExternalLogin.cshtml.cs	
+ ForgotPassword.cshtml	
+ C# ForgotPassword.cshtml.cs	
+ ForgotPasswordConfirmation.cshtml	
+ C# ForgotPasswordConfirmation.cshtml.cs	
+ Lockout.cshtml	
+ C# Lockout.cshtml.cs	
+ Login.cshtml	
+ C# Login.cshtml.cs	
+ LoginWith2fa.cshtml	
+ C# LoginWith2fa.cshtml.cs	
+ LoginWithRecoveryCode.cshtml	
+ C# LoginWithRecoveryCode.cshtml.cs	
+ Logout.cshtml	
+ C# Logout.cshtml.cs	
+ Register.cshtml	
+ C# Register.cshtml.cs	
+ RegisterConfirmation.cshtml	
+ C# RegisterConfirmation.cshtml.cs	
+ ResendEmailConfirmation.cshtml	
+ C# ResendEmailConfirmation.cshtml.cs	
+ ResetPassword.cshtml	
+ C# ResetPassword.cshtml.cs	
+ ResetPasswordConfirmation.cshtml	
+ C# ResetPasswordConfirmation.cshtml.cs	

dans Program.cs

Injecter la dépendance et ajouter au pipeline
User par défaut sans rôle

```
builder.Services.AddDbContext<ZombiePartyDbContext>(options =>  
options.UseSqlServer(builder.Configuration.GetConnectionString(  
"DefaultConnection"))).UseLazyLoadingProxies();
```

NOTE: s'injecte
automatiquement selon
la version

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

.....

```
app.UseRouting();
```

//Contrôle ajouté au pipeline http

```
app.UseAuthentication();
```

NOTE: s'injecte AVANT
UseAuthorization

```
app.UseAuthorization();
```

_Layout

Insérer la page de connexion _LoginPartial View dans la NavBar ou la page d'accueil

```
<button class="navbar-toggler" type="button" data-toggle="collapse" data-target=".navbar-collapse" aria-controls="navbarSupportedContent" aria-expanded="false" aria-label="Toggle navigation">
    <span class="navbar-toggler-icon"></span>
</button>
<div class="navbar-collapse collapse d-sm-inline-flex justify-content-between">
    <partial name="_LoginPartial" />
    <ul class="navbar-nav flex-grow-1">
        <li class="nav-item">
            <a class="nav-link text-dark" asp-area="" asp-controller="Home" asp-action="Index">Home</a>
        </li>
```

Ajouter dans Program.cs

Ajouter les pages Razor!

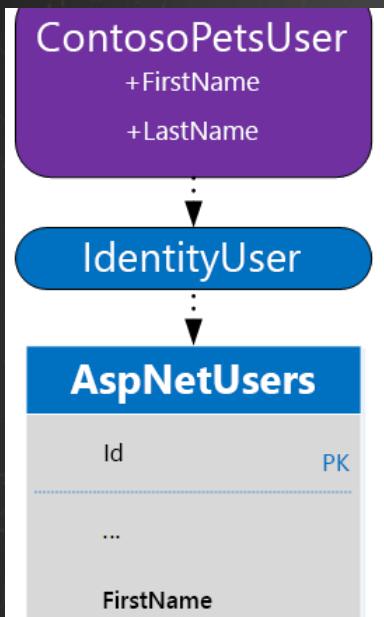
MS Identity n'utilise pas les contrôleurs, mais directement les pages Razor

NOTE: Si vous cliquez sur *Register* et qu'il ne se passe rien c'est qu'il vous manque ces éléments

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

```
app.MapControllerRoute(
    name: "default",
    pattern:
    "{controller=Home}/{action=Index}/{id?}");
app.MapRazorPages();
app.Run();
```

Modifier le profil utilisateur



- Exemple: ajouter le prénom et le nom à l'utilisateur
 - Ajouter au niveau du modèle.
 - modifier le modèle *IdentityUser*
 - modifier les pages de connexion pour avoir les nouveaux champs...

Modifier l'utilisateur *User* par défaut...

	Nom de la colonne
PK	<code>Id</code>
	<code>FirstName</code>
	<code>LastName</code>
	<code>UserName</code>
	<code>NormalizedUserName</code>
	<code>Email</code>
	<code>NormalizedEmail</code>
	<code>EmailConfirmed</code>
	<code>PasswordHash</code>
	<code>SecurityStamp</code>
	<code>ConcurrencyStamp</code>
	<code>PhoneNumber</code>
	<code>PhoneNumberConfirmed</code>
	<code>TwoFactorEnabled</code>
	<code>LockoutEnd</code>
	<code>LockoutEnabled</code>
	<code>AccessFailedCount</code>

```
// Hériter de IdentityUser permet d'ajouter des colonnes
//dans la table Users
public class ApplicationUser: IdentityUser
{
    public string NickName { get; set; }
}
```

```
public DbSet<ApplicationUser> ApplicationUser { get; set; }
```

Add-migration
Update-database

Modifier l'utilisateur *User* par défaut...

Nom de la colonne
Id
UserName
NormalizedUserName
Email
NormalizedEmail
EmailConfirmed
PasswordHash
SecurityStamp
ConcurrencyStamp
PhoneNumber
PhoneNumberConfirmed
TwoFactorEnabled
LockoutEnd
LockoutEnabled
AccessFailedCount
Discriminator
NickName

```
// Hériter de IdentityUser permet d'ajouter des colonnes
//dans la table Users
public class ApplicationUser: IdentityUser
{
    public string NickName { get; set; }
}
```

Add-migration
Update-database

Note sur les pages razor sans contrôleurs.

- Le cshtml.cs est le code derrière la vue. Elle contient un modèle et des actions : OnGet et OnPost.

comme si on faisait un contrôleur qui contient les actions [HttpGet] et [HttpPost].

- Les propriétés de type [BindProperty] sont disponibles directement dans la vue.

Au lieu de [RedirectToAction](#), on a des [RedirectToPage](#).

Mettre à jour le UI

- Ajouter les champs souhaités dans la *View* d'inscription *Register* et dans le Model approprié (*InputModel*)
- Ajuster le *OnPostAsync* sous l'action *Register* pour qu'il ajoute Les nouveaux champs à la table *User*.
- On doit faire la même chose au niveau de la gestion de profil (*Account/Manage/Index.cshtml*) et du *login par fournisseur externe*.
- *Changer le _loginPartial pour afficher les champs!*

PageModel Register

Areas/Identity/Page/Account

```
public class InputModel
{
    [Required]
    [EmailAddress]
    [Display(Name = "Email")]
    public string Email { get; set; }
    //Ajouter personnalisé
    [Required]
    [Display(Name = "NickName")]
    public string NickName { get; set; }
    //Ajouter dans la table par défaut
    [Required]
    [Display(Name = "PhoneNumber")]
    public string PhoneNumber { get; set; }
```

Page Register

Areas/Identity/Page/Account

```
<div class="form-group">
    <label asp-for="Input.Email"></label>
    <input asp-for="Input.Email" class="form-control" />
    <span asp-validation-for="Input.Email" class="text-danger"></span>
</div>
<div class="form-group">
    <label asp-for="Input.NickName"></label>
    <input asp-for="Input.NickName" class="form-control" />
    <span asp-validation-for="Input.NickName" class="text-danger"></span>
</div>
<div class="form-group">
    <label asp-for="Input.PhoneNumber"></label>
    <input asp-for="Input.PhoneNumber" class="form-control" />
    <span asp-validation-for="Input.PhoneNumber" class="text-danger"></span>
</div>
```

Bogue Identity autoGénéré

- Dans Program.cs supprimer l'injection du DbContext créé en double
- Supprimer le DbContext auto-généré

Net 7 et Net 8
Actuellement

