Cours n°12

# Programme du cours

|  |  |
| --- | --- |
| Activité | Durée |
| Introduction à l'authentification et l'autorisation dans ASP.NET Core | 20m |
| Mise en œuvre de l'authentification utilisateur | 45h |
| Mise en œuvre de l'autorisation utilisateur | 20m |
| Introduction à la gestion des Claims | 15m |
| Introduction à la gestion des erreurs avec TempData | 45m |
| Mise en œuvre de TempData pour la gestion des erreurs | 40m |

## Authentification et Autorisation en ASP.NET Core

## Objectif

L'objectif de cette section est d'introduire l'authentification et l'autorisation dans ASP.NET Core MVC et de montrer comment les mettre en œuvre en utilisant une table utilisateur existante dans la base de données.

**Vous devrez implémenter l’authentification et l’autorisation dans votre projet pour bloquer l’accès à la page utilisateur**

## Introduction à l'authentification et l'autorisation

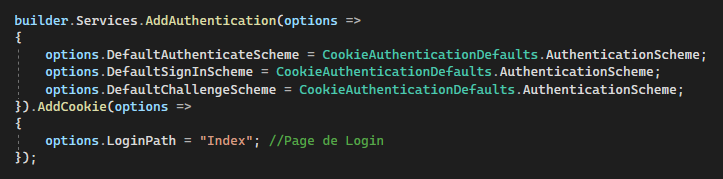
## L'authentification est le processus qui détermine l'identité d'un utilisateur.

## L'autorisation est le processus qui détermine ce qu'un utilisateur identifié est autorisé à faire.

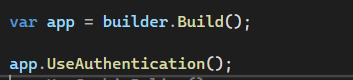
## ASP.NET Core fournit un framework pour gérer l'authentification et l'autorisation.

## Mise en œuvre de l'authentification utilisateur

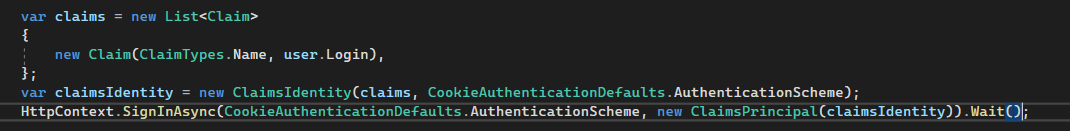
* Ajoutez les services d'authentification dans le fichier program.cs dans la méthode Main



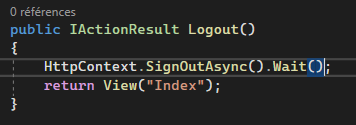
* Ajoutez le middleware d'authentification dans la méthode Configure.



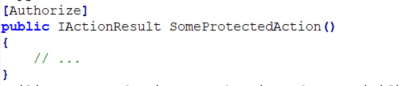
* Dans votre contrôleur de login, utilisez SignInAsync pour connecter l'utilisateur.



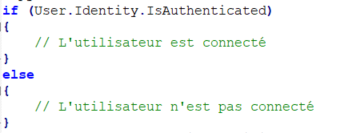
* Pour vous déconnecter, utilisez SignOutAsync.



* Utilisez l'attribut [Authorize] pour protéger vos contrôleurs ou actions.



* Utilisez User.Identity.IsAuthenticated pour vérifier si l'utilisateur est connecté.



Documentation : <https://learn.microsoft.com/fr-fr/aspnet/core/security/authentication/?view=aspnetcore-7.0>

## Introduction aux Claims

## Objectif

L'objectif de cette section est de comprendre les claims et de les utiliser pour enregistrer le login de l’utilisateur pour l’affichage du « Bonjour » sur la page utilisateur.

## Introduction

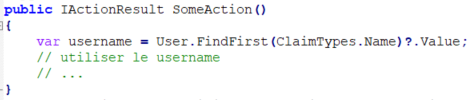
Un Claim représente une déclaration sur l'utilisateur. Il peut contenir diverses informations comme le nom d'utilisateur, l'adresse email, le rôle, etc.

Dans ASP.NET Core, l'identité d'un utilisateur est représentée sous forme d'un ensemble de Claims.

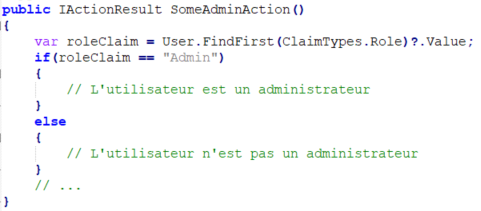
Les Claims sont stockés dans l'objet ClaimsIdentity qui fait partie de l'objet User accessible dans les contrôleurs.

## Lecture et utilisation des Claims

Vous pouvez accéder aux Claims de l'utilisateur à partir de l'objet User dans vos actions de contrôleur.



Vous pouvez également utiliser les Claims pour la logique d'autorisation personnalisée. Par exemple, si vous avez un Claim qui représente le rôle de l'utilisateur, vous pouvez l'utiliser pour autoriser l'accès à certaines actions seulement aux utilisateurs ayant un certain rôle.



# Gestion des erreurs avec TempData

## Objectif

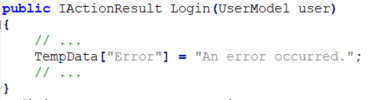
L'objectif de cette section est d'introduire TempData et comment l'utiliser pour gérer les erreurs. Utiliser les TempData pour afficher une erreur lors si l’utilisateur entre des identifiant invalide

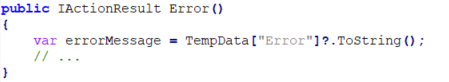
## Introduction à TempData

* TempData est un moyen de transférer des données entre les actions du contrôleur ou entre les requêtes.
* TempData utilise les cookies ou les sessions en interne pour stocker les données.
* TempData est utile pour les redirections, où vous voulez montrer des messages d'erreur ou d'autres informations temporairement.

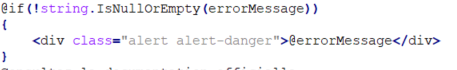
## Mise en œuvre de TempData pour la gestion des erreurs

Dans votre méthode d'action, utilisez TempData["Key"] pour définir un message d'erreur.





Affichez ce message d'erreur dans votre vue.



# Evolutions

Prenez avantage de l’authentification pour éviter de devoir passer le UserModel à chaque fois dans tous les contrôleurs