Module 11: Contrôle de l'accès aux applications Web ASP.NET MVC 5

# **Contenu:**

|  |  |
| --- | --- |
|  | [Aperçu du module](https://www.skillpipe.com/api/2.1/content/urn:uuid:4ed7876c-8864-4cf5-91ee-491726dfe322@2020-12-11T17:05:09Z/OPS/html/20486C11.html#P1) |
| **Leçon 1:** | [Implémentation de l'authentification et de l'autorisation](https://www.skillpipe.com/api/2.1/content/urn:uuid:4ed7876c-8864-4cf5-91ee-491726dfe322@2020-12-11T17:05:09Z/OPS/html/20486C11.html#P2) |
| **Leçon 2:** | [Attribution de rôles et d'appartenance](https://www.skillpipe.com/api/2.1/content/urn:uuid:4ed7876c-8864-4cf5-91ee-491726dfe322@2020-12-11T17:05:09Z/OPS/html/20486C11.html#P3) |
| **Laboratoire:** | [Contrôle de l'accès aux applications Web ASP.NET MVC 5](https://www.skillpipe.com/api/2.1/content/urn:uuid:4ed7876c-8864-4cf5-91ee-491726dfe322@2020-12-11T17:05:09Z/OPS/html/20486C11.html#P4) |
|  | [Examen du module et points à retenir](https://www.skillpipe.com/api/2.1/content/urn:uuid:4ed7876c-8864-4cf5-91ee-491726dfe322@2020-12-11T17:05:09Z/OPS/html/20486C11.html#P5) |

# **Aperçu du module**

L'authentification est une exigence vitale dans la plupart des applications Web. Les développeurs affichent généralement uniquement des informations restreintes à tous les utilisateurs. Les applications Web exigent que les utilisateurs s'authentifient pour afficher des informations exclusives. Les applications Web affichent également des informations spécifiques pertinentes pour des rôles d'utilisateur spécifiques. Microsoft ASP.NET comprend divers modèles d'authentification, notamment des fournisseurs d'authentification locaux, des systèmes d'authentification basés sur les revendications et des systèmes d'authentification fédérée. Vous devez savoir comment utiliser ces modèles d'authentification pour implémenter la fonctionnalité d'authentification dans votre application Web. Vous devez également savoir comment autoriser les utilisateurs et les rôles de votre application à restreindre l'accès aux informations en fonction de l'appartenance aux utilisateurs et aux rôles.

### **Objectifs**

Après avoir terminé ce module, vous serez en mesure de:

|  |  |
| --- | --- |
| • | Implémentez des systèmes d'authentification et d'autorisation dans votre application Web. |
| • | Gérez les utilisateurs et les rôles dans votre application Web. |
| • | Autorisez les utilisateurs et les rôles dans votre application Web. |

# Leçon 1: Implémentation de l'authentification et de l'autorisation

Les fournisseurs d'adhésion, qui ont été introduits dans ASP.NET 2.0, vous aident à créer des systèmes d'authentification et d'autorisation sécurisés pour les applications Web. Vous devez savoir comment utiliser le modèle de fournisseur d'appartenance dans ASP.NET pour modifier les méthodes d'authentification avec des modifications minimales du code. Vous pouvez utiliser l'authentification basée sur les revendications et l'authentification fédérée pour permettre aux utilisateurs externes de s'authentifier, d'accéder à l'application Web et de l'utiliser. Vous devez savoir comment configurer les restrictions dans l'application Web pour vous assurer que les utilisateurs accèdent uniquement aux informations spécifiques qui leur sont autorisées et pertinentes.

## **Objectifs de la leçon**

Après avoir terminé cette leçon, vous serez en mesure de:

|  |  |
| --- | --- |
| • | Répertoriez les options d'authentification locales disponibles pour une application Web ASP.NET. |
| • | Décrivez le fonctionnement de l'authentification basée sur le client. |
| • | Décrivez le fonctionnement de l'authentification fédérée. |
| • | Décrivez comment restreindre l'accès aux ressources. |
| • | Autorisez l'accès à des utilisateurs spécifiques aux actions du contrôleur d'application Web MVC 5. |

## **Fournisseurs d'authentification locaux**

Les fournisseurs d'authentification incluent du code qui s'exécute lorsque ASP.NET doit autoriser un utilisateur. Le code authentifie les utilisateurs à l'aide des informations stockées dans les bases de données principales, telles qu'Active Directory ou Microsoft SQL Server. Le système d'appartenance à ASP.NET comprend les fournisseurs d'authentification suivants: ActiveDirectoryMembershipProvider, SqlMembershipProvider, SimpleMembershipProvider et UniversalProviders. SimpleMembershipProvider et UniversalProviders sont des fournisseurs d'authentification qui prennent en charge le mécanisme d'authentification OAuth.

Le système d'appartenance à ASP.NET inclut également l'identité ASP.NET. L'identité ASP.NET inclut la prise en charge des profils et l'intégration OAuth et fonctionne avec OWIN.

Le système d'appartenance ASP.NET permet aux développeurs d'applications de basculer entre les fournisseurs d'authentification, sans modifier le code.

Voici une description des fournisseurs d'authentification: ActiveDirectoryMembershipProvider et SqlMembershipProvider:

|  |  |
| --- | --- |
| • | ActiveDirectoryMembershipProvider. Cette classe de fournisseur est définie dans l'espace de noms System.Web.Security et le fournisseur vous permet d'utiliser Active Directory comme référentiel d'appartenance et de rôle de votre application Web. |
| • | SqlMembershipProvider. Cette classe de fournisseur est définie dans l'espace de noms System.Web.Security. Le fournisseur fonctionne avec un schéma de table spécifique que vous pouvez générer à l'aide de la commande aspnet\_regdb.exe. |

**ActiveDirectoryMembershipProvider**et SqlMembershipProvider sont configurés pour fonctionner uniquement avec un schéma de table ou un répertoire spécifique. Pour surmonter cette restriction, Microsoft a développé SimpleMembershipProvider et UniversalProviders pour remplacer ActiveDirectoryMembershipProvider et SqlMembershipProvider.

|  |  |
| --- | --- |
| • | SimpleMembershipProvider. Il s'agit d'un fournisseur d'adhésion qui fonctionne avec SQL Server, SQL Server Compact Edition, Microsoft Azure SQL Database et d'autres versions de SQL Server. SimpleMembershipProvider ne nécessite que trois paramètres clés: le nom de la table, l'ID utilisateur et le nom d'utilisateur. Vous utilisez ce fournisseur pour implémenter l'authentification qui fonctionne avec n'importe quel schéma de table de base de données SQL Server. |
| • | Fournisseurs universels. Il s'agit d'un ensemble de fournisseurs d'appartenance qui fonctionne avec n'importe quelle base de données prise en charge par Entity Framework. Cependant, ces fournisseurs fonctionnent uniquement avec le schéma de base de données conçu par Microsoft. Lors de l'initialisation d'un fournisseur universel, si le schéma n'existe pas dans la base de données, le fournisseur génère le schéma. |

Vous pouvez utiliser ASP.NET Identity avec des frameworks ASP.NET, tels que ASP.NET MVC, Web Forms, Web Pages, Web API et SignalR. Vous pouvez utiliser ASP.NET Identity lors de la création d'applications Web, téléphoniques, de magasin ou hybrides.

**Question**: Quel est l'avantage d'utiliser SimpleMembershipProvider?

## **Authentification basée sur les revendications**

L'authentification basée sur les revendications est un modèle qui facilite la connexion unique. L'authentification unique est une fonctionnalité qui vous permet de recevoir une réclamation lorsque vous vous connectez à un système d'authentification centralisé. La revendication est un ticket que les systèmes d'authentification utilisent pour authentifier les ouvertures de session des utilisateurs. La revendication contient des informations sur l'identité de l'utilisateur qui aident les systèmes d'authentification à identifier les utilisateurs. Avec les systèmes d'authentification basés sur les revendications, vous pouvez concentrer vos efforts sur le développement de fonctions métier, plutôt que de vous soucier de l'authentification des utilisateurs.

L'authentification basée sur les revendications facilite:

|  |  |
| --- | --- |
| • | Authentification des utilisateurs pour accéder aux applications. |
| • | Stockage des informations de compte utilisateur et des mots de passe. |
| • | Vérification des répertoires d'entreprise pour les informations utilisateur. |
| • | Intégration de l'application avec les systèmes d'identité d'autres plateformes ou entreprises. |

Pour implémenter l'authentification basée sur les revendications dans votre application Web, vous pouvez utiliser Windows Identity Foundation (WIF). WIF est un ensemble de classes .NET Framework qui permet de lire les informations des revendications dans une application Web.

Les étapes suivantes décrivent le fonctionnement des systèmes d'authentification basés sur les revendications:

|  |  |
| --- | --- |
| 1. | Lorsqu'un utilisateur non authentifié demande une page Web, la demande est redirigée vers les pages du fournisseur d'identité (IP). |
| 2. | L'adresse IP vous oblige à entrer des informations d'identification telles que le nom d'utilisateur et le mot de passe. |
| 3. | Une fois que vous avez entré les informations d'identification, l'adresse IP émet un jeton. Ensuite, le jeton est renvoyé au navigateur Web. |
| 4. | Le navigateur Web est redirigé vers la page Web que vous avez initialement demandée. WIF détermine si le jeton satisfait aux exigences pour accéder à la page Web. Si le jeton satisfait à toutes les exigences, un cookie est émis pour établir une session. Ce cookie garantit que le processus d'authentification n'a lieu qu'une seule fois. Ensuite, le contrôle d'authentification est transmis à l'application. |

ASP.NET Identity prend en charge l'authentification basée sur les revendications. Dans ASP.NET Identity, un ensemble de revendications représente l'identité de l'utilisateur.

**Question**: Quels sont les avantages de l'utilisation de l'authentification basée sur les revendications?

## **Authentification fédérée**

Les fédérations s'appuient sur l'authentification basée sur les revendications pour permettre à des parties externes telles que des entreprises de confiance d'accéder à leurs applications. Les fédérations utilisent la norme de réclamation WS-Federations pour permettre l'échange de réclamations entre deux parties de manière standardisée.

WIF prend en charge les fédérations à l'aide du module HTTP WSFederationAuthenticationModule. WSFederationAuthenticationModule vous permet d'implémenter la prise en charge des fédérations dans votre application ASP.NET, sans implémenter de logique individuelle.

L'authentification fédérée permet au service de jetons de sécurité (STS) de:

|  |  |
| --- | --- |
| • | Traitez les réclamations des partenaires commerciaux. |
| • | Extraire les informations utilisateur des revendications. |

STS vous permet de vous concentrer davantage sur l'écriture de la logique métier. Il élimine le besoin de gérer les informations d'authentification des partenaires commerciaux, dans votre application Web.

La configuration de WSFederationAuthenticationModule permet de spécifier le STS vers lequel les demandes non authentifiées doivent être redirigées. WIF fournit deux méthodes d'authentification fédérée: FederatedPassiveSignIn et Passive Redirect.

**Contrôle FederatedPassiveSignIn**

Considérez les cas où des utilisateurs non authentifiés tentent d'accéder à des ressources protégées et que vous souhaitez rediriger ces utilisateurs vers une page de connexion. Dans de tels cas, vous pouvez incorporer le contrôle FederatedPassiveSignIn dans la page de connexion de votre application Web, pour rediriger les utilisateurs non authentifiés vers la page de connexion. Le contrôle FederatedPassiveSignIn est configuré avec des informations sur l'émetteur (STS).

Vous pouvez utiliser le contrôle FederatedPassiveSignIn pour:

|  |  |
| --- | --- |
| • | Excluez la protection à l'échelle de l'application. |
| • | Incluez une page de connexion avec des contrôles cliquables. |

**Redirection passive**

Considérez les cas où des utilisateurs non authentifiés tentent d'accéder à une ressource protégée et que vous souhaitez rediriger ces utilisateurs vers un STS sans utiliser de page d'ouverture de session d'application. Dans de tels cas, vous pouvez utiliser la redirection passive.

La redirection passive permet à STS de:

|  |  |
| --- | --- |
| • | Vérifiez l'identité des utilisateurs non authentifiés. |
| • | Émettez des jetons de sécurité contenant les revendications appropriées pour les utilisateurs. |

La redirection passive vous oblige à ajouter WSFederationAuthenticationModule dans le pipeline des modules HTTP (Hypertext Transfer Protocol), pour identifier les demandes d'utilisateurs non authentifiés et rediriger les utilisateurs vers le STS que vous spécifiez. L'ajout de WSFederationAuthenticationModule dans le pipeline HTTP traite les informations de revendication avant de transmettre la revendication au moteur ASP.NET.

Vous pouvez instancier WSFederationAuthenticationModule et l'utiliser pour déclencher le processus de connexion. Cette fonctionnalité est similaire à celle du contrôle FederatedPassiveSignIn; cependant, la redirection passive implémente cette fonctionnalité de connexion avec un effort de codage minimal.

**Question**: Quels sont les avantages de l'utilisation de l'authentification fédérée?

## **Restreindre l'accès aux ressources**

Vous pouvez restreindre l'accès des utilisateurs en implémentant l'attribut Authorize dans un contrôleur au lieu d'utiliser le fichier Web.config comme vous le feriez dans une application ASP.NET WebForms. Le fichier Web.config nécessite que des fichiers physiques existent pour que le contrôle d'accès fonctionne. Vous ne pouvez pas utiliser le fichier Web.config pour restreindre l'accès des utilisateurs car les applications MVC acheminent les demandes vers des actions et non vers des pages.

**Autorisation des méthodes d'action**

[Autoriser()]

public ActionResult GetEmployee ()

{

retour Vue ();

}

Observez l'attribut Authorize dans l'exemple de code. Si vous spécifiez cet attribut, ASP.NET exige que les utilisateurs soient autorisés à accéder à la vue renvoyée par l'exemple de code.

Si vous ajoutez l'attribut Authorize au niveau de la classe, l'attribut oblige les utilisateurs à se connecter avant de pouvoir accéder à quoi que ce soit dans la classe de contrôleur. Pour permettre aux utilisateurs anonymes d'accéder à une partie spécifique de votre classe, vous pouvez utiliser l'attribut AllowAnonymous.

**Utilisation de l'attribut AllowAnonymous**

[AllowAnonymous ()]

Registre public ActionResult ()

{

retour Vue ();

}

**Question**: Pourquoi devriez-vous utiliser l'attribut Authorize au lieu du fichier Web.config pour contrôler l'autorisation des pages dans votre application ASP.NET MVC?

## **Démonstration: comment autoriser l'accès aux actions du contrôleur**

Dans cette démonstration, vous verrez comment:

|  |  |
| --- | --- |
| • | Générez une authentification pour accéder à une action du contrôleur. |
| • | Gérez les demandes d'actions non authentifiées qui nécessitent une authentification à l'aide d'ASP.NET. |

### **Étapes de démonstration**

|  |  |
| --- | --- |
| • | Vous trouverez les étapes dans la section «Leçon 1: Implémentation de l'authentification et de l'autorisation» sur la page suivante:<https://github.com/MicrosoftLearning/20486-DevelopingASPNETMVCWebApplications/blob/master/Instructions/20486C/20486C_MOD11_DEMO.md>. |

# Leçon 2: Attribution de rôles et d'appartenance

Les rôles et les appartenances complètent les fonctionnalités d'authentification, pour vous aider à contrôler tous les modes d'accès dans les applications Web. Pour définir des niveaux d'accès pour différents types d'utilisateurs, vous devez savoir comment implémenter les rôles et les utilisateurs. ASP.NET fournit des fournisseurs de rôles et des fournisseurs d'appartenance pour vous aider à attribuer des rôles et des utilisateurs. ASP.NET vous permet également de créer des fournisseurs de rôles personnalisés et des fournisseurs d'appartenance personnalisés. Ces fournisseurs personnalisés vous permettent de stocker des informations sur les rôles et les utilisateurs dans des sources de données non prises en charge par ASP.NET, telles qu'une base de données Oracle.

## **Objectifs de la leçon**

Après avoir terminé cette leçon, vous serez en mesure de:

|  |  |
| --- | --- |
| • | Décrivez les fournisseurs de rôles dans ASP.NET. |
| • | Décrivez comment ajouter des comptes d'utilisateurs aux rôles. |
| • | Décrivez comment créer des fournisseurs de rôles personnalisés. |
| • | Décrivez les services aux membres. |
| • | Décrivez comment créer des fournisseurs d'appartenance personnalisés. |
| • | Ajoutez du code à une application Web MVC 5 pour aider les utilisateurs à réinitialiser leur mot de passe. |

## **Fournisseurs de rôles dans ASP.NET**

Les fournisseurs de rôles aident à développer des applications à l'aide d'un modèle de sécurité basé sur les rôles. Il existe plusieurs fournisseurs de rôles disponibles dans ASP.NET:

|  |  |
| --- | --- |
| • | ActiveDirectoryRoleProvider. Cette classe de fournisseur est définie dans l'espace de noms System.Web.Security et le fournisseur vous permet d'utiliser Active Directory comme modèle de gestion des rôles. |
| • | SqlRoleProvider. Cette classe de fournisseur est définie dans l'espace de noms System.Web.Security. Le fournisseur fonctionne uniquement avec un schéma de table spécifique dans Microsoft SQL Server. Vous pouvez générer ce schéma à l'aide de la commande aspnet\_regdb.exe. |
| • | WindowsTokenRoleProvider. Ce fournisseur utilise le jeton d'authentification Windows pour déterminer le rôle des utilisateurs. Ensuite, le fournisseur vérifie si les utilisateurs appartiennent à un groupe stocké dans le jeton d'authentification Windows qui a été généré lors de leur première connexion. Le format du nom de groupe est Domaine \ Groupe. |

**ActiveDirectoryRoleProvider**et SqlRoleProvider ont des restrictions telles que le manque de prise en charge des bases de données non SQL ou du schéma défini par le développeur. Par conséquent, Microsoft a développé SimpleRoleProvider et UniversalProviders pour remplacer ActiveDirectoryRoleProvider et SqlRoleProvider. La liste suivante décrit SimpleRoleProvider et UniversalProviders:

|  |  |
| --- | --- |
| • | SimpleRoleProvider. Il s'agit d'un fournisseur de rôles qui fonctionne avec SQL Server, les éditions SQL Compact et d'autres versions de SQL Server. SimpleRoleProvider vous permet d'implémenter une autorisation basée sur la table définie par les développeurs d'applications. |
| • | Fournisseurs universels. Il s'agit d'une version indépendante de la base de données qui fonctionne avec n'importe quelle base de données prise en charge par Entity Framework. Cependant, le schéma de base de données est déterminé par Microsoft. Le fournisseur génère généralement des tables lors de l'initialisation. |

ASP.NET Identity contient un fournisseur de rôles qui vous permet de restreindre l'accès à l'application à l'aide de rôles. Par exemple, vous pouvez créer un rôle nommé Admin. Vous pouvez facilement créer des utilisateurs dans ASP.NET Identity et les ajouter à des rôles.

Vous pouvez choisir un fournisseur de rôles en fonction du fournisseur d'appartenance que vous sélectionnez. Vous pouvez également mélanger les fournisseurs de rôles; par exemple, vous pouvez combiner SimpleMembershipProvider et UniversalRoleProvider. Cependant, vous devez éviter de mélanger les fournisseurs de rôles car la manière dont chaque fournisseur identifie les utilisateurs est différente.

**Question**: Quelle est la différence entre SimpleRoleProvider et SqlRoleProvider?

## **Ajout de comptes d'utilisateurs aux rôles**

Lorsque vous utilisez un fournisseur de rôle basé sur une base de données, tel que SimpleRoleProvider ou UniversalRoleProvider, vous pouvez charger les données initiales dans la base de données en modifiant directement la table. Vous pouvez également modifier la table à l'aide de la fonction AddUsersToRoles. Pour utiliser la fonction AddUsersToRoles, vous devez implémenter l'outil de gestion personnalisé dans votre application. L'outil de gestion personnalisé permet d'appeler la fonction AddUsersToRoles, pour ajouter des rôles à une base de données.

**Utilisation de la fonction AddUsersToRoles**

AddUserToRoles ("Peter", nouvelle chaîne [] {"Admin", "Staff"})

Vous pouvez également utiliser la fonction AddUsersToRoles dans les applications d'entreprise pour ajouter des utilisateurs à un rôle. Cependant, vous devez vous assurer que le rôle auquel vous souhaitez ajouter des utilisateurs existe avant d'utiliser la fonction AddUsersToRoles.

## **Création d'un fournisseur de rôles personnalisés**

ASP.NET vous permet de créer des fournisseurs de rôles personnalisés. Les fournisseurs de rôles personnalisés vous permettent d'implémenter la gestion des rôles qui utilise votre propre schéma et logique de base de données. Pour créer un fournisseur de rôles personnalisé, vous devez créer une classe qui hérite de la classe RoleProvider.

**Création d'un fournisseur de rôle personnalisé**

public class CustomRoleProvider: RoleProvider

{

}

Dans la classe RoleProvider, vous devez implémenter la fonction GetRolesForUser. La fonction GetRolesForUser prend le nom d'utilisateur comme entrée et renvoie une liste de rôles auxquels appartient l'utilisateur. Vous pouvez écrire votre propre code pour obtenir des informations de rôle à partir de la base de données ou d'autres magasins principaux.

**Implémentation de la fonction GetRolesForUser**

chaîne de remplacement public [] GetRolesForUser (chaîne de nom d'utilisateur)

{

// code pour renvoyer une liste de rôles pour les utilisateurs

}

Après avoir implémenté la fonction GetRolesForUser, vous devez appliquer le fournisseur de rôle personnalisé à l'application en modifiant le fichier Web.config.

**Configuration d'un site pour utiliser un fournisseur de rôle personnalisé**

<roleManager defaultProvider = "CustomRoleProvider" enabled = "true" cacheRolesInCookie = "false">

<fournisseurs>

<clear />

<add name = "CustomRoleProvider" type = "CustomRoleProvider" />

</providers>

</roleManager>

**Question**: Pourquoi devriez-vous créer un fournisseur de rôle personnalisé?

## **Fournir des services aux membres**

Vous devez ajouter le SimpleMembershipProvider à la section d'appartenance du fichier Web.config pour l'utiliser pour les services d'adhésion.

**Configuration du fournisseur d'adhésion simple**

<Membership defaultProvider = "SimpleMembershipProvider">

<fournisseurs>

<clear />

<add name = "SimpleMembershipProvider" type = "WebMatrix.WebData.SimpleMembershipProvider, WebMatrix.WebData" />

</providers>

</membership>

Ensuite, vous devez créer une table User. Si vous n'avez pas encore créé de table User, vous pouvez modifier le fichier AccountModel.cs pour générer la table. Pour afficher le fichier AccountModel.cs, vous devez sélectionner le modèle d'application Internet lorsque vous créez le projet.

**Créer une table d'utilisateurs**

[Table ("UserProfile")]

classe publique UserProfile

{

[Clé]

[DatabaseGeneratedAttribute (DatabaseGeneratedOption.Identity)]

public int UserId {get; ensemble; }

chaîne publique UserName {get; ensemble; }

chaîne publique Country {get; ensemble; }

}

Après avoir modifié le fichier AccountModel.cs, vous devez appeler la fonction WebSecurity.InitializeDatabaseConnection dans le fichier App.Start.cs. La fonction WebSecurity.InitializeDatabaseConnection garantit que SimpleMembershipProvider est configuré avec des chaînes de connexion à la base de données.

**Connexion à la table des utilisateurs**

WebSecurity.InitializeDatabaseConnection ("DBConn", "User", "id", "Username", false)

Si une table spécifique n'existe pas dans une base de données, le paramètre autoCreateTables permet d'indiquer au fournisseur d'appartenance qu'il peut créer la table. Pour initialiser SimpleMembershipProvider, vous devez ajouter l'attribut InitializeSimpleMembership à la classe AccountController.

**Ajout de l'attribut InitializeSimpleMembership**

[Autoriser]

[InitializeSimpleMembership]

Public class AccountController: Controller

{

}

**Question**: Pourquoi utiliser la méthode InitializeDatabaseConnection?

## **Créer un fournisseur d'abonnement personnalisé**

Si vous devez ajouter des fonctionnalités à un fournisseur d'appartenance, comme une logique personnalisée pour l'authentification, vous devez implémenter un fournisseur d'appartenance personnalisé.

**Un fournisseur d'abonnement personnalisé**

public class CustomMembershipProvider: SimpleMembershipProvider

{

}

**Remplacement de la méthode ValidateUser**

public override bool ValidateUser (string username, string password)

{

}

Dans l'exemple de code précédent, la méthode ValidateUser valide le nom d'utilisateur et le mot de passe par rapport au magasin d'appartenance.

**Remplacer le constructeur du fournisseur**

public CustomAdminMembershipProvider (SimpleSecurityContext simpleSecurityContext)

{

}

**Ajout d'un fournisseur personnalisé à Web.config**

<Membership defaultProvider = "CustomMemberProvider">

<fournisseurs>

<clear />

<add name = "CustomMemberProvider" type = "CustomAdminMembershipProvider" />

</providers>

</membership>

**Question**: Pourquoi devriez-vous implémenter un fournisseur d'adhésion personnalisé?

## **Démonstration: Comment réinitialiser un mot de passe**

Dans cette démonstration, vous verrez comment écrire du code qui utilise un fournisseur de services d'adhésion pour réinitialiser le mot de passe de l'utilisateur. À l'aide de ce code, vous pouvez permettre aux utilisateurs de contrôler leurs propres mots de passe et de réinitialiser le mot de passe sans impliquer un administrateur de site.

### **Étapes de démonstration**

|  |  |
| --- | --- |
| • | Vous trouverez les étapes dans la section «Leçon 2: Attribution de rôles et d'appartenance» sur la page suivante:<https://github.com/MicrosoftLearning/20486-DevelopingASPNETMVCWebApplications/blob/master/Instructions/20486C/20486C_MOD11_DEMO.md>. |

# **Atelier: Contrôle de l'accès aux applications Web ASP.NET MVC 5**

### **Scénario**

Une grande partie des fonctionnalités de votre application de partage de photos proposée est en place. Cependant, les parties prenantes sont préoccupées par la sécurité car il n'y a aucune restriction sur les tâches que les utilisateurs peuvent effectuer. Les restrictions suivantes sont requises:

|  |  |
| --- | --- |
| • | Seuls les membres du site devraient pouvoir ajouter ou supprimer des photos. |
| • | Seuls les membres du site doivent pouvoir ajouter ou supprimer des commentaires. |

Il vous a été demandé de résoudre ces problèmes en créant un système d'adhésion pour l'application de partage de photos. Les visiteurs doivent pouvoir s'inscrire en tant qu'utilisateurs de l'application Web et créer des comptes utilisateur pour eux-mêmes. Après l'inscription, lorsque les utilisateurs se connecteront à l'application, ils auront accès à des actions telles que l'ajout et la suppression de photos et de commentaires. Les utilisateurs anonymes n'auront pas accès pour effectuer ces actions. De plus, les utilisateurs enregistrés devraient également pouvoir réinitialiser leur propre mot de passe.

### **Objectifs**

Après avoir terminé cet atelier, vous serez en mesure de:

|  |  |
| --- | --- |
| • | Configurez une application Web pour utiliser l'authentification par formulaire ASP.NET avec des comptes stockés dans la base de données Microsoft Azure SQL. |
| • | Écrivez des modèles, des contrôleurs et des vues pour authentifier les utilisateurs dans une application Web. |
| • | Fournissez l'accès aux ressources dans une application Web. |
| • | Permettez aux utilisateurs de réinitialiser leur propre mot de passe. |

##### **Configuration du laboratoire**

Durée estimée: 90 minutes

Vous trouverez les étapes de haut niveau sur la page suivante:<https://github.com/MicrosoftLearning/20486-DevelopingASPNETMVCWebApplications/blob/master/Instructions/20486C/20486C_MOD11_LAB_MANUAL.md>.

Vous trouverez les étapes détaillées sur la page suivante:<https://github.com/MicrosoftLearning/20486-DevelopingASPNETMVCWebApplications/blob/master/Instructions/20486C/20486C_MOD11_LAK.md>.

### **Exercice 1: Configuration des fournisseurs d'authentification et d'adhésion**

##### **Scénario**

Vous souhaitez utiliser une base de données Microsoft Azure SQL pour stocker les comptes d'utilisateurs et les informations d'appartenance.

Dans cet exercice, vous allez:

|  |  |
| --- | --- |
| • | Créez une base de données Microsoft Azure SQL. |
| • | Configurez un fournisseur pour se connecter à la base de données. |

### **Exercice 2: Création des vues de connexion et de registre**

##### **Scénario**

Vous avez configuré l'application de partage de photos pour se connecter à la base de données Microsoft Azure SQL pour les services d'authentification et d'adhésion. Cependant, pour utiliser l'authentification par formulaire dans une application MVC, vous devez créer des classes de modèle, des contrôleurs et des vues qui permettent aux utilisateurs de se connecter, de se déconnecter et de s'inscrire à un compte.

Dans cet exercice, vous allez:

|  |  |
| --- | --- |
| • | Ajoutez des classes de modèle. |
| • | Ajoutez des contrôleurs. |
| • | Importez les vues de connexion et d'enregistrement. |
| • | Testez les composants développés. |

### **Exercice 3: Autoriser l'accès aux ressources**

##### **Scénario**

Maintenant que vous avez activé et testé l'authentification, vous pouvez autoriser l'accès aux ressources pour les utilisateurs anonymes et authentifiés.

Vous devez vous assurer que:

|  |  |
| --- | --- |
| • | Seuls les membres du site peuvent ajouter ou supprimer des photos. |
| • | Seuls les membres du site peuvent ajouter ou supprimer des commentaires. |
| • | Les actions du contrôleur de compte sont autorisées correctement. |
| • | Seuls les utilisateurs authentifiés voient la vue\_Créer pour les commentaires dans la vue Affichage. |

### **Exercice 4: Facultatif - Création d'une vue de réinitialisation de mot de passe**

##### **Scénario**

Les visiteurs du site peuvent désormais s'inscrire en tant qu'utilisateurs de l'application de partage de photos et se connecter au site pour pouvoir ajouter des photos et des commentaires. Cependant, ils n'ont pas la possibilité de changer leur mot de passe. Dans cet exercice, vous allez créer une page de réinitialisation de mot de passe en utilisant le fournisseur de services aux membres.

Effectuez cet exercice si le temps le permet.

### **Question (s) de révision**

**Vérifiez vos connaissances**

**Découverte**

**Dans l'exercice 3, lorsque vous avez essayé d'ajouter une photo avant de vous connecter à l'application, pourquoi ASP.NET a-t-il affiché leConnexionvue?**

Afficher la solution Réinitialiser

**Vérifiez vos connaissances**

**Découverte**

**Comment pouvez-vous vous assurer que seuls les employés d'Adventure Works ont accès auEffaceraction de laphotomanette?**

Afficher la solution Réinitialiser

# **Examen du module et points à retenir**

Dans ce module, vous avez abordé les différents fournisseurs d'appartenance et de rôle dans ASP.NET. Vous avez comparé les avantages de l'utilisation de SimpleProviders et UniversalProviders avec les avantages de l'utilisation d'autres fournisseurs, pour les applications centrées sur les bases de données. Cependant, pour les applications qui doivent utiliser Windows pour l'authentification et l'autorisation, vous pouvez utiliser les fournisseurs de rôle et d'appartenance existants. Vous avez également vu comment implémenter des fournisseurs personnalisés, pour ajouter des fonctionnalités telles que le cryptage des mots de passe à votre application Web.

### **Problèmes et scénarios du monde réel**

Lorsque vous créez des applications Web, vous devrez peut-être créer des fournisseurs personnalisés car vous ne souhaitez pas utiliser le schéma fourni par Microsoft. Cependant, vous pouvez utiliser SimpleProviders pour supprimer le besoin de développer des fournisseurs personnalisés et réduire l'effort requis pour la création d'applications.