



Dotnet France  
Technologies Sharepoint, SQL Server & .NET

Association Dotnet France

**www.Mcours.com**

Site N°1 des Cours et Exercices Email: [contact@mcours.com](mailto:contact@mcours.com)

# Gestion de l'authentification et des autorisations

Version 1.1

BEDE Nicolas  
GERAUD Cédric



James RAVAILLE

<http://blogs.dotnet-france.com/jamesr>

# Sommaire

---

1	Introduction.....	3
2	Authentification.....	4
2.1	Contrôles implémentés dans ASP.NET .....	4
2.1.1	Partie Login de la Boîte à Outils .....	4
2.1.2	La classe Membership .....	4
2.1.3	La classe Roles .....	5
2.1.4	Outil d'administration Web.....	6
2.2	Les différents types d'authentification .....	9
2.2.1	Windows.....	9
2.2.2	Formulaire .....	9
2.2.3	Passport Microsoft .....	12
2.2.4	Emprunt d'identité .....	12
3	Gestion des autorisations.....	14
3.1	Autoriser / refuser l'accès suivant l'utilisateur ou le groupe .....	14
3.2	Appliquer des autorisations à un fichier ou dossier spécifique .....	15
4	Application pratique.....	16
4.1	Présentation de l'exercice .....	16
4.1.1	Présentation de l'application .....	16
4.1.2	Politique de sécurité.....	16
4.2	Présentation des différentes étapes à réaliser .....	17
4.3	Préparation d'une base de données .....	17
4.4	Définition des providers .....	23
4.5	Création des rôles et utilisateurs, et définition de leurs droits d'accès sur les ressources de l'application .....	24
4.6	Création de l'application .....	34
4.7	Utilisation de l'application.....	35
5	Conclusion .....	37



## 1 Introduction

Il est important pour un site internet de gérer l'identification de ses utilisateurs ainsi que les autorisations. Nous allons voir dans ce chapitre, les différentes méthodes d'authentification ainsi que la gestion des autorisations suivant un utilisateur ou un rôle précis.

Authentifier des utilisateurs permet d'autoriser l'accès à aux ressources de votre application, qu'à certaines personnes ou proposer des services/fonctionnalités supplémentaires à des personnes privilégiées (comptes payants, comptes d'administrateurs ou modérateurs entres autres). Cela sert aussi à enregistrer ses informations personnelles et à suivre les actions d'un utilisateur, par exemple sur un site d'achat pour remplir un panier.

Nous parlions d'administrateurs et de modérateurs. Ce sont aussi des exemples de gestions d'autorisations puisque seuls eux doivent être autorisés à changer certaines données du site. Pour nous faciliter la tâche, ASP.NET implémente déjà ces gestions, il ne nous reste quasiment plus qu'à les paramétrer, comme nous allons l'étudier dans la suite de ce chapitre.

## 2 Authentification

### 2.1 Contrôles implémentés dans ASP.NET

ASP.NET implémente déjà des propriétés et méthodes, qui vont nous simplifier la tâche et sont fournies avec certains paramètres de sécurité, qu'il ne nous serait pas forcément évident de reproduire.

#### 2.1.1 Partie Login de la Boîte à Outils

Nous allons voir une description des différents contrôles de login qui se trouvent dans la boîte à outils et qu'ASP.NET implémente par défaut :

Contrôle	Description
Login	Permet d'ajouter un formulaire de connexion à la page. Il demande le Login et le mot de passe.
LoginView	Nous permet, grâce à un <i>template</i> que nous faisons nous même, d'afficher des informations suivant le statut (connecté ou anonyme) de la personne.
LoginStatus	Suivant le statut de la personne, ce contrôle affichera Connexion ou Déconnexion.
LoginName	Si l'utilisateur est connecté, ce contrôle affichera son login. Par défaut, en local sur notre machine, nous sommes logués avec notre compte Windows.
PasswordRecovery	Affiche une interface permettant la récupération de mot de passe.
CreateUserWizard	Affiche un formulaire qui permet la création d'un nouvel utilisateur.
ChangePassword	Ce contrôle va afficher un formulaire permettant le changement du mot de passe de l'utilisateur.

Comme vous pouvez le constater, grâce aux contrôles d'ASP.NET, on peut créer l'interface graphique sans même saisir une ligne de code. Vous pouvez utiliser un contrôle *ValidationSummary* pour afficher les informations de validations retournées par certains de ces contrôles (contenant des contrôles de validation), cela améliorera l'accessibilité de votre site.

#### 2.1.2 La classe Membership

Cette classe permet de gérer les utilisateurs. Elle vous permet entre autres d'ajouter, supprimer, sélectionner et modifier des utilisateurs. Cette classe se trouve dans *System.Web.Security.Membership*.

Voici les méthodes qu'elle contient :

Méthode	Définition
CreateUser	Vous permet d'ajouter un utilisateur à la base de données.
DeleteUser	Vous permet de supprimer un utilisateur de la

	base de données.
FindUsersByEmail	Récupère sous forme de collection tous les utilisateurs contenant l'adresse passée.
FindUsersByName	Récupère sous forme de collection tous les utilisateurs contenant le nom passé.
GeneratePassword	Génère un mot de passe de la taille spécifiée.
GetAllUsers	Récupère tous les utilisateurs sous forme de collection.
GetNumberOfUsersOnline	Récupère le nombre de personnes connectées.
GetUser	Récupère l'utilisateur courant. Retourne un objet MembershipUser.
GetUserNameByEmail	Récupère l'utilisateur qui correspond à l'email passé.
UpdateUser	Met à jour l'utilisateur (inscrit les informations dans la base de données).
ValidateUser	Vérifie la validité du login et du mot de passe.

### 2.1.3 La classe Roles

Les rôles permettent de gérer les autorisations comme l'accès à certaines parties du site. Cette classe se trouve dans System.Web.Security.Roles. Voyons les méthodes statiques associées à cette classe :

Méthode	Description
AddUserToRole	Ajoute un utilisateur à un rôle. Il possède plusieurs variantes comme AddUsersToRole qui ajoute des utilisateurs à un rôle. Pour plus d'informations allez sur msdn : <a href="http://msdn.microsoft.com/fr-fr/library/system.web.security.roles_methods(VS.80).aspx">http://msdn.microsoft.com/fr-fr/library/system.web.security.roles_methods(VS.80).aspx</a>
CreateRole	Créer un rôle.
DeleteRole	Supprimer un rôle.
FindUsersInRole	Renvoie une collection contenant tous les utilisateurs du rôle spécifié.
GetAllRoles	Renvoie une collection contenant tous les rôles existant.
GetRolesForUser	Renvoie une collection contenant les rôles auxquels l'utilisateur appartient.
IsUserInRole	Renvoie un booléen indiquant si l'utilisateur courant est dans le rôle spécifié.
RemoveUserFromRole	Enlève l'utilisateur du rôle spécifié. Il y a pour lui aussi des variantes possibles comme RemoveUsersFromRoles qui enlève tous les utilisateurs définis du rôle spécifié. Pour plus d'informations voir msdn : <a href="http://msdn.microsoft.com/fr-">http://msdn.microsoft.com/fr-</a>

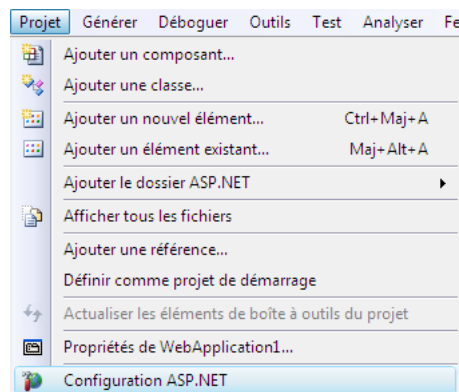
[fr/library/system.web.security.roles\\_methods\(VS.80\).aspx](http://fr/library/system.web.security.roles_methods(VS.80).aspx)

### 2.1.4 Outil d'administration Web

Avant de faire cette manipulation, il faut que vous prépariez votre site Web. Pour cela créez des sous dossiers sur lesquels seront appliquées les permissions.

Pour paramétrer les options d'authentification il existe un outil d'administration permettant, via une interface graphique, de gérer l'authentification sans toucher à aucun fichier.

Pour y accéder aller dans le menu Projet puis choisissez Configuration ASP.NET.



Une fois ceci fait vous verrez s'ouvrir une page Web ressemblant à celle ci-dessous (vous serez sur l'onglet Accueil). En haut de cette page se trouve des onglets, choisissez l'onglet Sécurité.



You can use the Web Site Administration Tool to manage all the security settings for your application. You can set up users and passwords (authentication), create roles (groups of users), and create permissions (rules for controlling access to parts of your application).

By default, user information is stored in a Microsoft SQL Server Express database in the Data folder of your Web site. If you want to store user information in a different database, use the Provider tab to select a different provider.

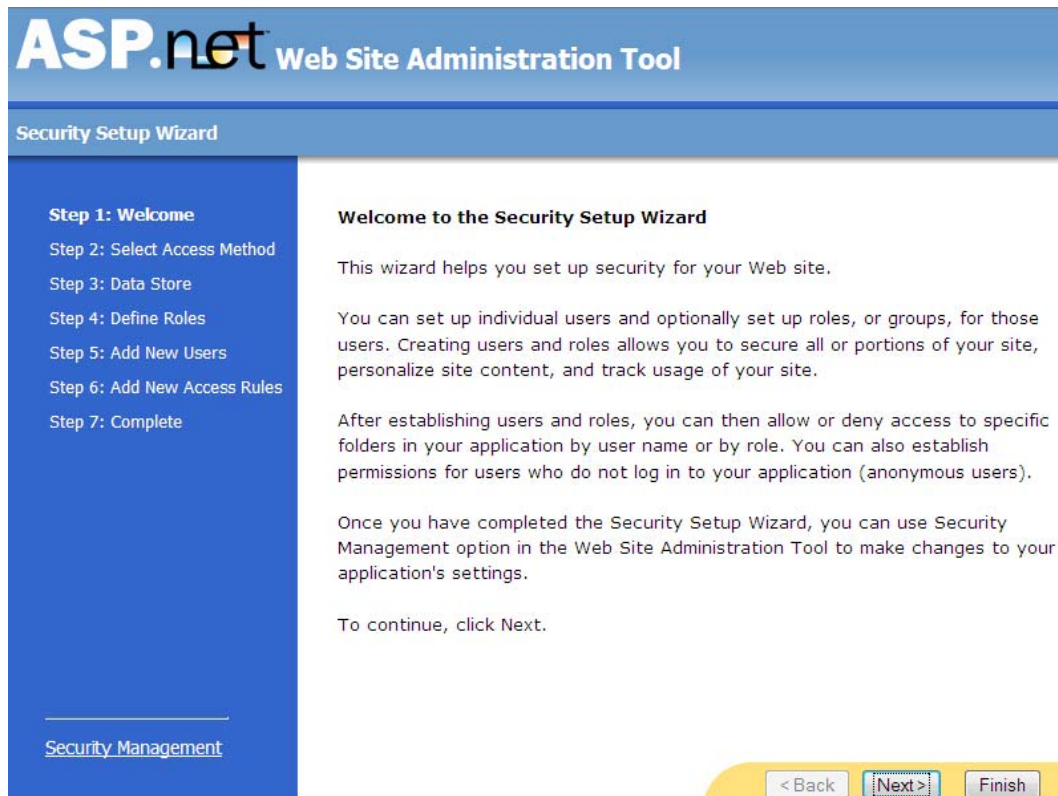
[Use the security Setup Wizard to configure security step by step.](#)

Click the links in the table to manage the settings for your application.

Users	Roles	Access Rules
Existing users: <b>0</b> <a href="#">Create user</a> <a href="#">Manage users</a>	Roles are not enabled <a href="#">Enable roles</a> Create or Manage roles	<a href="#">Create access rules</a> <a href="#">Manage access rules</a>
<a href="#">Select authentication type</a>		

Sélectionnez le lien au milieu de la page: *Use the security Setup Wizard to configure security step by step*. Vous allez arriver sur la page d'accueil de l'assistant de configuration de sécurité.





**ASP.net Web Site Administration Tool**

**Security Setup Wizard**

**Step 1: Welcome**

Step 2: Select Access Method  
Step 3: Data Store  
Step 4: Define Roles  
Step 5: Add New Users  
Step 6: Add New Access Rules  
Step 7: Complete

[Security Management](#)

**Welcome to the Security Setup Wizard**

This wizard helps you set up security for your Web site.

You can set up individual users and optionally set up roles, or groups, for those users. Creating users and roles allows you to secure all or portions of your site, personalize site content, and track usage of your site.

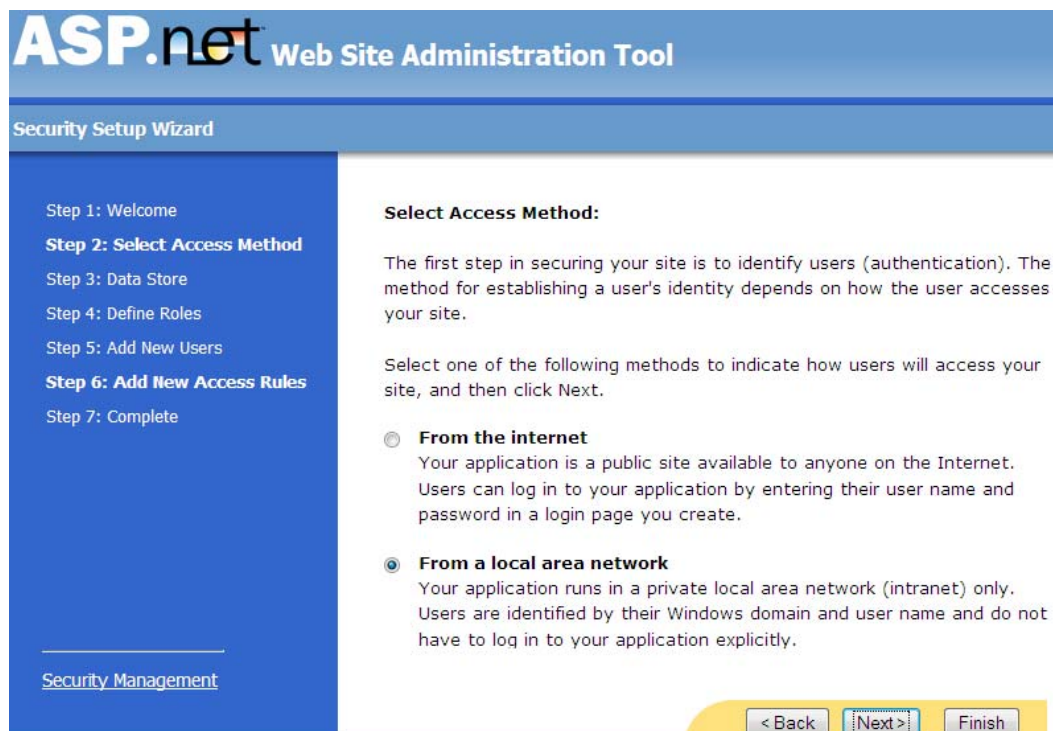
After establishing users and roles, you can then allow or deny access to specific folders in your application by user name or by role. You can also establish permissions for users who do not log in to your application (anonymous users).

Once you have completed the Security Setup Wizard, you can use Security Management option in the Web Site Administration Tool to make changes to your application's settings.

To continue, click Next.

< Back   **Next >**   Finish

Cliquez sur suivant. Vous arrivez maintenant sur le choix de l'authentification. Si c'est une authentification via internet, choisissez la première possibilité. En revanche si votre application Web est destinée à être sur un réseau local et qu'il va falloir utiliser l'authentification Windows ou Active directory, choisissez la seconde option. Choisir le réseau local vous fait passer directement à l'étape 6.



**ASP.net Web Site Administration Tool**

**Security Setup Wizard**

Step 1: Welcome  
**Step 2: Select Access Method**  
Step 3: Data Store  
Step 4: Define Roles  
Step 5: Add New Users  
**Step 6: Add New Access Rules**  
Step 7: Complete

[Security Management](#)

**Select Access Method:**

The first step in securing your site is to identify users (authentication). The method for establishing a user's identity depends on how the user accesses your site.

Select one of the following methods to indicate how users will access your site, and then click Next.

- From the internet**  
Your application is a public site available to anyone on the Internet. Users can log in to your application by entering their user name and password in a login page you create.
- From a local area network**  
Your application runs in a private local area network (intranet) only. Users are identified by their Windows domain and user name and do not have to log in to your application explicitly.

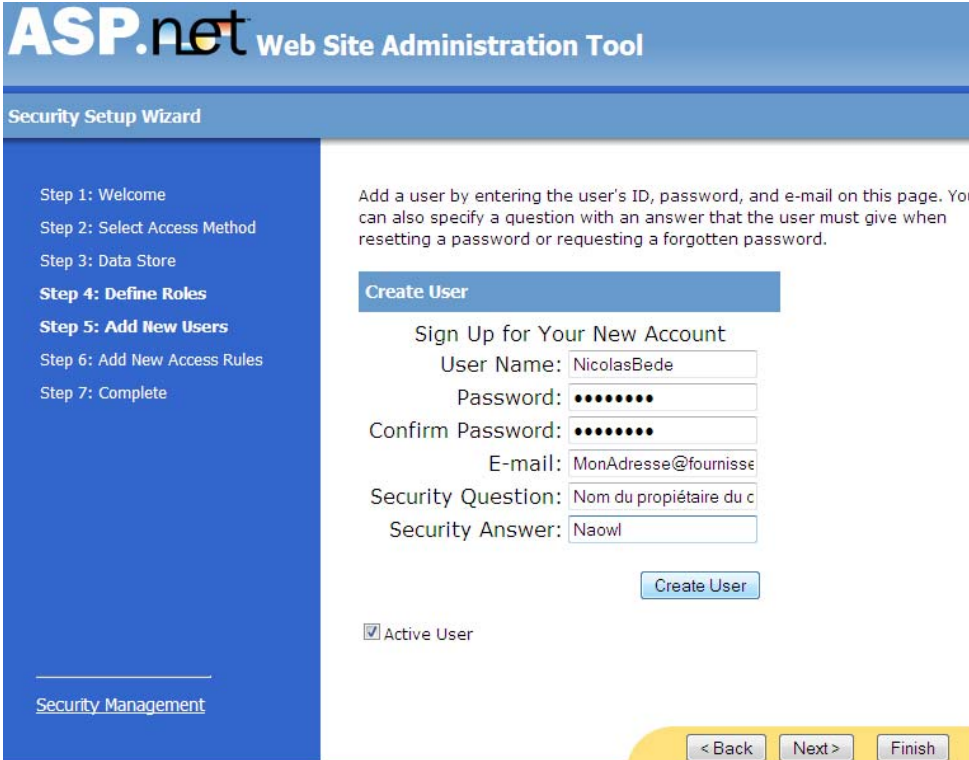
< Back   **Next >**   Finish

Nous allons suivre la progression étape par étape, c'est-à-dire qu'on va faire comme si on avait sélectionné la méthode par internet pour accéder à l'étape 3.

La prochaine étape vous amène sur une page vous expliquant comment changer de *provider*. Puisqu'il n'est pas utile de rester plus longtemps sur cette étape, continuons vers l'étape suivante.

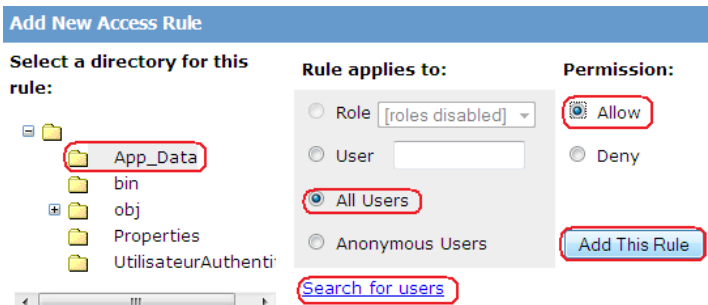
L'étape 4 va vous permettre d'activer/désactiver les rôles. Les rôles sont des groupes d'utilisateurs qu'il vous est permis de créer dans votre application Web. C'est utile pour pouvoir appliquer des droits sur tout un groupe de personne plutôt que de répéter les mêmes droits plusieurs fois. On peut en avoir besoin par exemple pour un groupe d'administrateur ou de modérateur. Si vous voulez l'activer, cocher la case, sinon assurez vous qu'elle est décoché et passez à l'étape suivante. Si vous avez activé les rôles, l'assistant de configuration va ouvrir une page depuis laquelle vous pourrez créer des rôles.

L'étape 5, elle, va vous permettre de créer des utilisateurs. Pour cela remplissez les champs et faites "Créer l'utilisateur". Quand vous aurez fini, passez à l'étape suivante.



The screenshot shows the ASP.NET Web Site Administration Tool interface. The title bar reads "ASP.NET Web Site Administration Tool". Below it is a "Security Setup Wizard" section with a sidebar listing steps 1 through 7. Step 5, "Add New Users", is highlighted. The main content area is titled "Create User" and contains the following fields: "User Name" (NicolasBede), "Password" (masked with dots), "Confirm Password" (masked with dots), "E-mail" (MonAdresse@fournisse), "Security Question" (Nom du propriétaire du c), and "Security Answer" (Naowl). There is a "Create User" button and a checked checkbox for "Active User". At the bottom, there are navigation buttons: "< Back", "Next >", and "Finish".



L'étape suivante (6) consiste en la création des règles de sécurité. Vous êtes arrivé directement ici si vous avez choisi l'authentification en local.



The screenshot shows the "Add New Access Rule" dialog box. It has three main sections: "Select a directory for this rule:", "Rule applies to:", and "Permission:". In the "Select a directory" section, a tree view shows folders like App\_Data, bin, obj, Properties, and UtilisateurAuthenti. In the "Rule applies to:" section, "All Users" is selected under the "Role" dropdown. In the "Permission:" section, "Allow" is selected. There is a "Search for users" button at the bottom.



Choisissez dans la partie tout à gauche le dossier sur lequel vous voulez ajouter une règle. Dans le panneau du milieu choisissez quel type d'utilisateur (Tous, Anonyme, un utilisateur spécifique, un rôle/groupe) la règle va s'appliquer. Remarquer le lien Chercher des utilisateurs qui vous permet de spécifier un utilisateur spécifique. Il ne vous reste plus qu'à spécifier si vous autorisez ou si vous refusez et à cliquer sur le bouton Créer la règle. Vous verrez alors vos règles dans le cadre juste en dessous comme sur l'image qui suit.

Permission	Users And Roles	Delete
Deny	 [anonymous]	Delete
Allow	 [all]	Delete

Quand vous aurez fini, faites suivant puis cliquez sur Terminer pour compléter la configuration.

Maintenant que l'assistant est terminé, si vous avez bien configuré votre application, les contrôles implémentés dans ASP.NET devraient fonctionner. Un contrôle Login devrait vous permettre de vous identifier ...

Remarque : Il est important de cibler l'authentification ou les rôles, c'est-à-dire que vous aller mettre dans un sous dossier les pages qui possèdent les mêmes besoins d'authentification. Par exemple si vous avez des utilisateurs et une gestion de profil. Pour modifier leurs profils les utilisateurs doivent être authentifié, vous allez donc mettre toute la partie gestion du profil dans un sous dossier qui ne permettra l'accès qu'aux utilisateurs authentifié et le refusera à tous les autres.

## 2.2 Les différents types d'authentification

Il y a 3 types d'authentification possible : avec Windows, par formulaire et avec un *Passport* qui est un service de Microsoft. Nous allons les voir plus en détails dans ce qui suit.

### 2.2.1 Windows

Si votre application Web se trouve en local avec la base de données des utilisateurs ou Active Directory, vous pouvez utiliser l'authentification Windows. Ce système d'authentification est surtout utilisé au sein des entreprises où les machines se connectant à l'application sont dans le même réseau.

Pour l'activer il faut ajouter ce code dans le web.config (plus exactement dans la balise system.web) :

```
<authentication mode="Windows" />
```

On peut configurer ce mode d'authentification avec l'assistant de configuration de sécurité comme on l'a déjà vu ou avec ce que l'on verra plus tard dans la partie 3 pour protéger l'accès à des fichiers et dossiers.

### 2.2.2 Formulaire

L'authentification par formulaire est le plus courant sur internet. Comme pour les autres types d'authentification, les utilisateurs possèdent un login et un mot de passe. La connexion

s'effectue par formulaire. Une fois qu'il a rentré ses données, on les récupère et on vérifie si ce couple login/ mot de passe existe dans la base de données. Voyons plutôt cela pas à pas. Commençons déjà par configurer ASP.NET pour utiliser les formulaires pour l'authentification.

### 2.2.2.1 Configurer le Web.config

Activer le mode formulaire dans le Web.config (dans system.web) :

```
Web.config
<system.web>
  <authentication mode="Forms" />
</system.web>
```

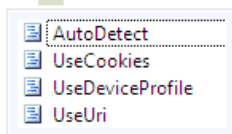
Bien sûr ce que nous avons vu dans la partie authentification avec Windows pour les autorisations est toujours valable. C'est-à-dire que dans le web.config on pourrait aussi avoir :

```
<authorization>
  <deny users="?" />
</authorization>
```

Il faut savoir que par défaut ASP.NET va rediriger l'utilisateur vers une page d'authentification si celui-ci n'est pas authentifié. Cette page est *LoginForm.aspx*. Pour changer cette page voici le code à mettre :

```
Web.config
<authentication mode="Forms">
  <forms loginUrl="AuthFormulaire.aspx" />
</authentication>
```

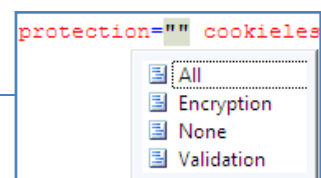
```
cookieless="" />
```



C'est sur cette balise *forms* que l'on va pouvoir spécifier comment sera stocké l'identifiant de session ainsi que d'autre chose comme le timeout de la connexion (avec la propriété *timeout* en minutes). La propriété à utiliser est *cookieless* comme vous le voyez sur l'image.

AutoDetect	Détecte automatiquement si le navigateur supporte les cookies. Si c'est le cas il envoie un cookie avec l'identifiant de session. Sinon ASP.NET utilisera un autre moyen : passer la variable par l'url.
UseCookies	Envoie un cookie même si le navigateur ne le supporte pas.
UseUri	Utilise l'url pour passer la variable.

Le problème du protocole HTTP utilisé pour la communication entre le serveur et l'utilisateur est qu'il n'est pas sécurisé. Un pirate pourrait donc récupérer l'identifiant



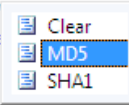
et le mot de passe d'un utilisateur (voir de l'administrateur). Pour palier à ce défaut il faut configurer le site Web pour utiliser le protocole HTTPS, qui lui est sécurisé, sur la page de login (au moins). On pourrait aussi utiliser un certificat SSL grâce à IIS. Le cookie qui va être généré après authentification est lui aussi sensible. Un pirate pourrait créer un faux cookie (celui-ci est écrit en clair). On peut se protéger contre cela en utilisant la propriété *protection* de la balise *authentication* (ou *forms*) pour crypter ce cookie. Par défaut il est positionné sur All. Si on met Encryption, le cookie sera crypté en 3DES mais le serveur n'effectuera pas de vérification sur le cookie pour vérifier son intégrité. Si elle est mise sur Validation, le cookie sera vérifié à chaque transaction pour vérifier qu'il n'a pas été modifié. Évidemment si cette propriété est mise sur None, ni la validation ni l'encrytage ne sera effectué.

Revenons à l'authentification. Pour authentifier un utilisateur, on va pouvoir se baser sur une base de données, une liste dans le Web.config, ou encore un fichier XML. On peut bien évidemment créer son propre système de fonctionnement (comme utiliser un objet).

### 2.2.2.2 Lister les utilisateurs dans le Web.config

Voici un exemple rapide de la création d'une liste depuis le Web.config :

```
<authentication mode="Forms">
  <forms loginUrl="AuthFormulaire.aspx"
    protection="Encryption"
    cookieless="AutoDetect"
    timeout="10">
    <credentials passwordFormat="">
      <user name="Identifiant"
        password="M0n_P0ss"
      />
    </credentials>
  </forms>
</authentication>
```



Comme vous pouvez le voir, la liste se trouve dans la balise *forms*. La liste des utilisateurs est contenue dans la balise *credentials* avec des balises orphelines *user*. La balise *user* prend comme propriété name (l'identifiant) et password (le mot de passe). En plus de cela, comme on le remarque sur l'image, on peut spécifier 3 types de format pour le mot de passe : clear (texte en clair), MD5 et SHA1. Les deux derniers sont des types de cryptage. Pour pouvoir générer ces mots de passe, reportez-vous au namespace `System.Security.Cryptography`.

### 2.2.2.3 Personnaliser l'authentification par formulaire

Pour ce faire nous utiliserons la classe `FormsAuthentication`. Avec cette classe vous pourrez récupérer certaines configurations ainsi qu'effectuer les actions que nous verrons un peu plus tard au niveau de l'authentification par formulaire.

Propriété	Définition
FormsCookieName	Récupère le nom du cookie utilisé par la configuration courante.
FormsCookiePath	Récupère le chemin du cookie pour l'application suivante.

LoginUrl	Accède ou modifie l'url de la page contenant le formulaire de login vers laquelle ASP.NET redirige.
RequireSSL	Permet de savoir si une connexion SSL est requise pour envoyer le cookie.

Méthodes	Définition
Authenticate	Essaye d'authentifier l'utilisateur à l'aide de la liste rentrée dans le Web.config.
GetAuthCookie	Créer le cookie d'authentification pour un utilisateur défini.
GetRedirectUrl	Récupère l'url de la page depuis laquelle a été effectuée la redirection.
HashPasswordForStoringInConfigFile	Hash le mot de passe. Il prend en argument le mot de passe et le type de hashage que l'on veut.
RedirectFromLoginPage	Redirige l'utilisateur vers la page de login spécifiée dans la configuration.
SetAuthCookie	Créer et envoie un ticket d'authentification.
SignOut	Supprimer le ticket d'authentification, ce qui a pour but de « déconnecter » l'utilisateur. S'il veut à nouveau avoir accès à la zone protégée, il doit à nouveau s'authentifier.

### 2.2.3 Passport Microsoft

Les passports sont issus d'un service de Microsoft qui permet d'authentifier des gens. Le principe est simple : votre compte Passport contient toutes les informations que vous voulez divulguer aux sites (adresse email, passe temps entres autres) et il vous suffit de passez votre compte Passport au site.

Comme ce système ne devrait pas être abordé lors de la certification Web 2.0 nous n'en parlerons pas.

### 2.2.4 Emprunt d'identité

Par défaut, ASP.NET effectue les requêtes pour les ressources (comme un fichier, accès à des données et autres) avec le compte d'ASP.NET. Et ce même si l'utilisateur courant est connecté. Cependant il peut arriver que cela génère des erreurs ou que vous vouliez utiliser un autre compte.

Quand on utilise le mode d'authentification Windows, ASP.NET peut utiliser le compte de l'utilisateur courant connecté, pour accéder aux ressources. On parle alors d'emprunt d'identité, en anglais "Impersonation". Pour ce faire on va utiliser la balise *identity* dans la zone *system.web* du *Web.config*. Il suffit de mettre l'attribut *impersonate* à *true*.



Si cet attribut est à *true* et que l'utilisateur est anonyme, alors un compte par défaut sera utilisé. En revanche si l'utilisateur est authentifié, son compte Windows sera utilisé. On peut aussi spécifier le compte qui sera utilisé en remplissant les attributs *userName* et *password* de la balise *identity*.

### 3 Gestion des autorisations

ASP.NET va vous permettre de gérer les autorisations d'accès aux différentes parties et ressources du site. On peut appliquer ces autorisations sur un utilisateur spécifique ou plus généralement sur un groupe pour restreindre l'accès à un fichier voir un dossier.

#### 3.1 Autoriser / refuser l'accès suivant l'utilisateur ou le groupe

Pour commencer nous allons voir comment autoriser ou refuser l'accès suivant l'utilisateur ou un groupe spécifique. On va utiliser le Web.config de base, c'est-à-dire celui qui se trouve à la racine de notre application Web. Les autorisations s'appliqueront à tout le site.

Ce qui suit se place dans la balise `<system.web>` :

*Web.config*

```
<authorization>
  <allow users="*" />
  <deny users="?" />
</authorization>
```

Comme le montre cet exemple, c'est dans la balise *authorization* que nous allons définir nos autorisations. Cette balise va en contenir d'autres. Il en existe deux importantes : *allow* et *deny*. La première va permettre de définir les autorisations et la seconde les refus. Comme vous le constaterez l'exemple présenté n'est pas fonctionnel puisqu'il autorise tout le monde (l'étoile \* représente tout le monde) et qu'il refuse ce qui ne sont pas authentifié ( ? représente les utilisateurs anonymes). Si au lieu de spécifier des utilisateurs (*users*) on veut spécifier des groupes, on utilisera plutôt l'attribut *roles*.

On pourrait ne vouloir passer que les utilisateurs Nicolas et Cedric. Dans ce cas voici le code que contiendrait *authorization* :

*Web.config*

```
<allow users="Nicolas, Cedric" />
<deny users="*" />
```

En ce qui concerne les autorisations, elles seront appliquées dans l'ordre dans lequel on les met. C'est-à-dire que Nicolas et Cedric vont être autorisés et qu'après il va interdire tout le monde.

Parlons maintenant du cas de l'authentification par Windows ou Active directory. Le nom à utiliser serait pareil que dans ce cas, c'est-à-dire le nom de l'utilisateur. En revanche il faudrait spécifier : le nom de la machine si le compte est en local ou le nom du domaine avant l'utilisateur (de la forme Nom-Machine\Utilisateur).





### 3.2 Appliquer des autorisations à un fichier ou dossier spécifique

Pour cela il peut y avoir deux manières de procéder, la première étant de créer un Web.config dans chaque dossier dans lesquelles on veut des autorisations différentes. Cependant on ne peut pas appliquer des autorisations spécifiques à un fichier avec cette méthode et surtout ce n'est pas pratique d'avoir nos autorisations séparées dans plusieurs fichiers. La seconde méthode est donc de tout mettre dans un seul Web.config : celui de la racine (c'est la bonne façon de procéder). Pour cela rien de bien compliqué : on va ajouter une balise importante qui se nomme *location*. Avant d'en parler voyons un exemple :

```
Web.config

<configuration>
  <system.web>
    <!-- ... -->
  </system.web>

  <location path="MaPage.aspx">
    <system.web>
      <authorization>
        <allow roles="Admin"/>
        <deny users="*/>
      </authorization>
    </system.web>
  </location>
</configuration>
```

La balise location permet de créer une configuration spécifique au chemin qu'on lui spécifie dans l'attribut *path* qui peut aussi bien être un fichier comme ici, ou un dossier. Ensuite location accepte la configuration comme si on était dans un Web.config différent avec dans notre exemple un *system.web* et la configuration des autorisations.



## 4 Application pratique

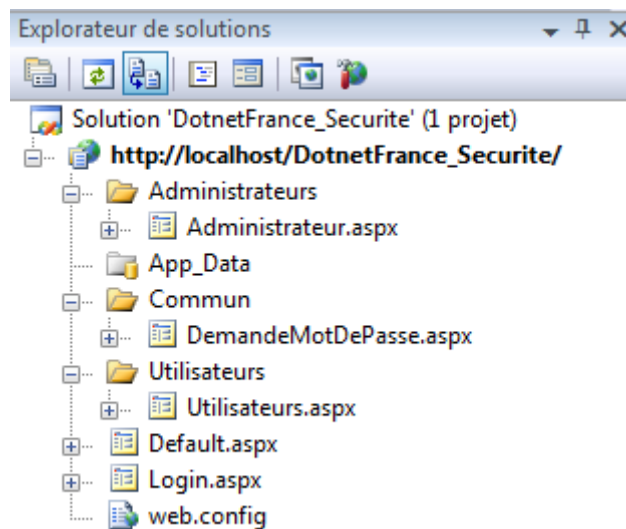
Dans cette partie, nous allons mettre en pratique, au sein d'un exercice, le service ASP .NET permettant de gérer la sécurité (à la fois l'authentification et les autorisations).

### 4.1 Présentation de l'exercice

Dans cet exercice, nous vous proposons de définir et d'utiliser des droits sur des utilisateurs, sur les ressources d'une application. Dans notre cas, les ressources sont uniquement constituées de pages ASP .NET.

#### 4.1.1 Présentation de l'application

Voici l'arborescence des répertoires de l'application, et des fichiers qu'ils contiennent :



Dans cette phase de présentation de l'exercice, nous pouvons uniquement définir le contenu des pages suivantes :

- La page Administrateur.aspx affiche le message « Bonjour les administrateurs ». Seuls les utilisateurs considérés comme administrateurs doivent pouvoir accéder à cette page.
- La page Utilisateurs.aspx affiche le message « Bonjour les utilisateurs ». Tous les utilisateurs authentifiés doivent pouvoir accéder à cette page.
- La page DemandeMotDePasse.aspx doit pouvoir être accéder par tous les utilisateurs (authentifiés et anonymes).
- La page Default.aspx affiche le message « Page par défaut ».

#### 4.1.2 Politique de sécurité

La politique de sécurité se base sur deux groupes d'utilisateurs (qu'on appellera plus tard des rôles) :

- Un groupe *RoleAdmin*, contenant des utilisateurs considérés comme administrateurs (utilisateurs avec pouvoir). Pour simplifier l'exercice, ce groupe contiendra un seul utilisateur dont les caractéristiques sont les suivantes :

## 4.2 Présentation des différentes étapes à réaliser

Les étapes de réalisation de cet exercice sont les suivantes :

- Préparation d'une base de données.
- Définition des providers.
- Création des rôles et utilisateurs, et définition de leurs droits d'accès sur les ressources de l'application.
- Création de l'application.

## 4.3 Préparation d'une base de données

Le service d'application ASP .NET de sécurité doit utiliser une base de données, pour stocker l'ensemble des informations sur les rôles (groupes d'utilisateurs), les utilisateurs, et leurs droits d'accès aux ressources de l'application. Pour accéder à une base de données, ce service nécessite de providers : un pour les rôles, et un autre pour les utilisateurs. De manière native, le Framework .NET 3.5 ne propose que des providers pour les bases de données SQL Server. Les classes représentant ces providers sont les suivantes :

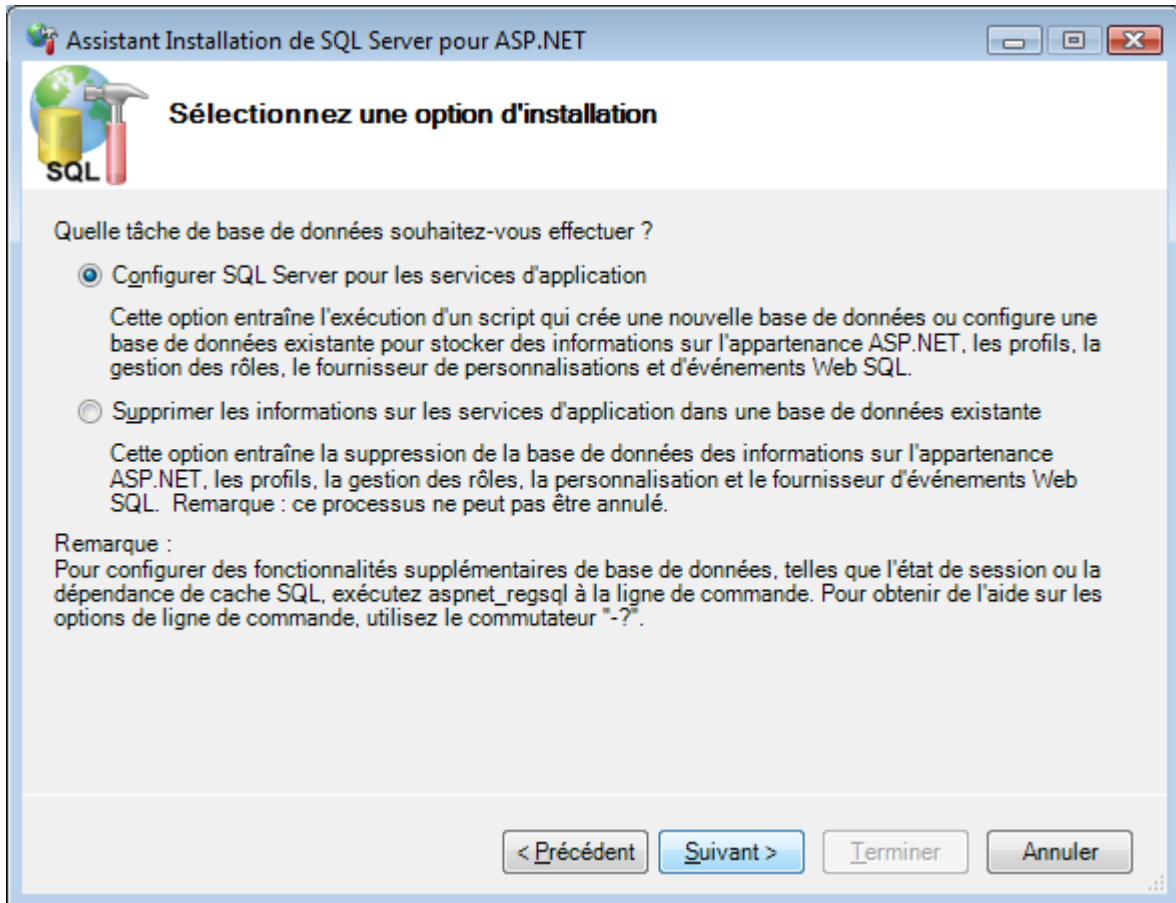
- *System.Web.Security.SqlMembershipProvider* : provider utilisé pour la gestion des utilisateurs.
- *System.Web.Security.SqlRoleProvider* : provider utilisé pour la gestion des rôles.

Ces providers pointent vers une base de données. Il convient alors de créer cette base de données, et de créer tous les objets SQL requis par le service ASP .NET de gestion de la sécurité. Pour ce faire, Microsoft propose au travers du Framework .NET, l'outil *aspnet\_regsql* (il est contenu dans le répertoire *C:\Windows\Microsoft.NET\Framework\v2.0.50727*). Cet outil permet de configurer les bases de données SQL Server, pour utiliser l'ensemble des services d'application ASP .NET. Il existe deux manières d'utiliser cet outil :

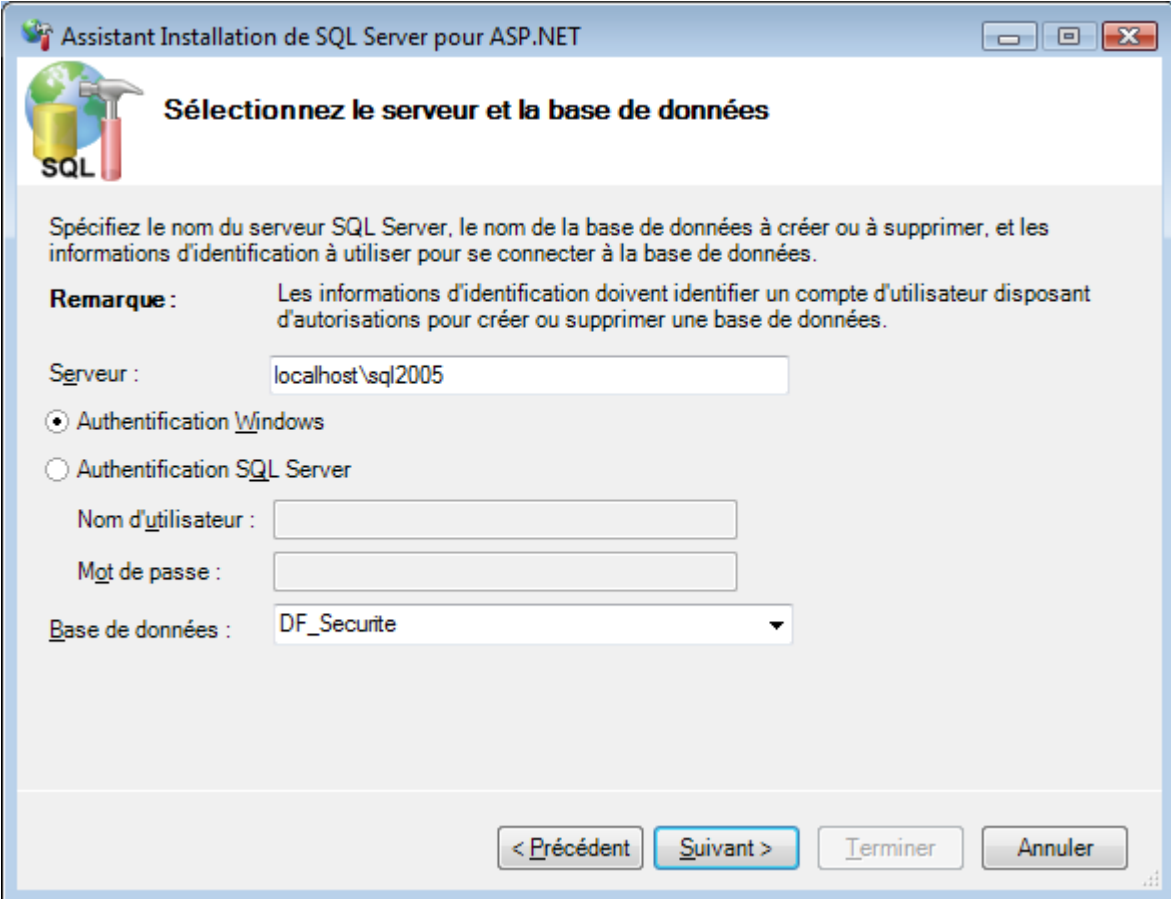
- Soit en lui demandant de configurer une base de données pour **l'ensemble des services d'application ASP .NET**. Cette opération est possible en lançant directement cet outil depuis l'explorateur de fichiers, ou en tapant son nom dans une fenêtre de commandes DOS. Un assistant est alors lancé pour vous guider dans la configuration de la base de données. Voici les écrans de cet outil :



Cliquez sur le bouton *Suivant*. La fenêtre suivante apparaît :



La question est simple : souhaitez-vous configurer une base de données SQL Server pour utiliser l'ensemble des services d'application ASP .NET, ou effectuer l'opération inverse ? Vous remarquerez que vous n'avez pas le choix sur les services d'application (c'est tout ou rien). Choisissez alors le premier choix et cliquez sur le bouton *Suivant*. La fenêtre suivante apparaît :



Assistant Installation de SQL Server pour ASP.NET

### Sélectionnez le serveur et la base de données

Spécifiez le nom du serveur SQL Server, le nom de la base de données à créer ou à supprimer, et les informations d'identification à utiliser pour se connecter à la base de données.

**Remarque :** Les informations d'identification doivent identifier un compte d'utilisateur disposant d'autorisations pour créer ou supprimer une base de données.

Serveur : localhost\\sql2005

Authentification Windows

Authentification SQL Server

Nom d'utilisateur :

Mot de passe :

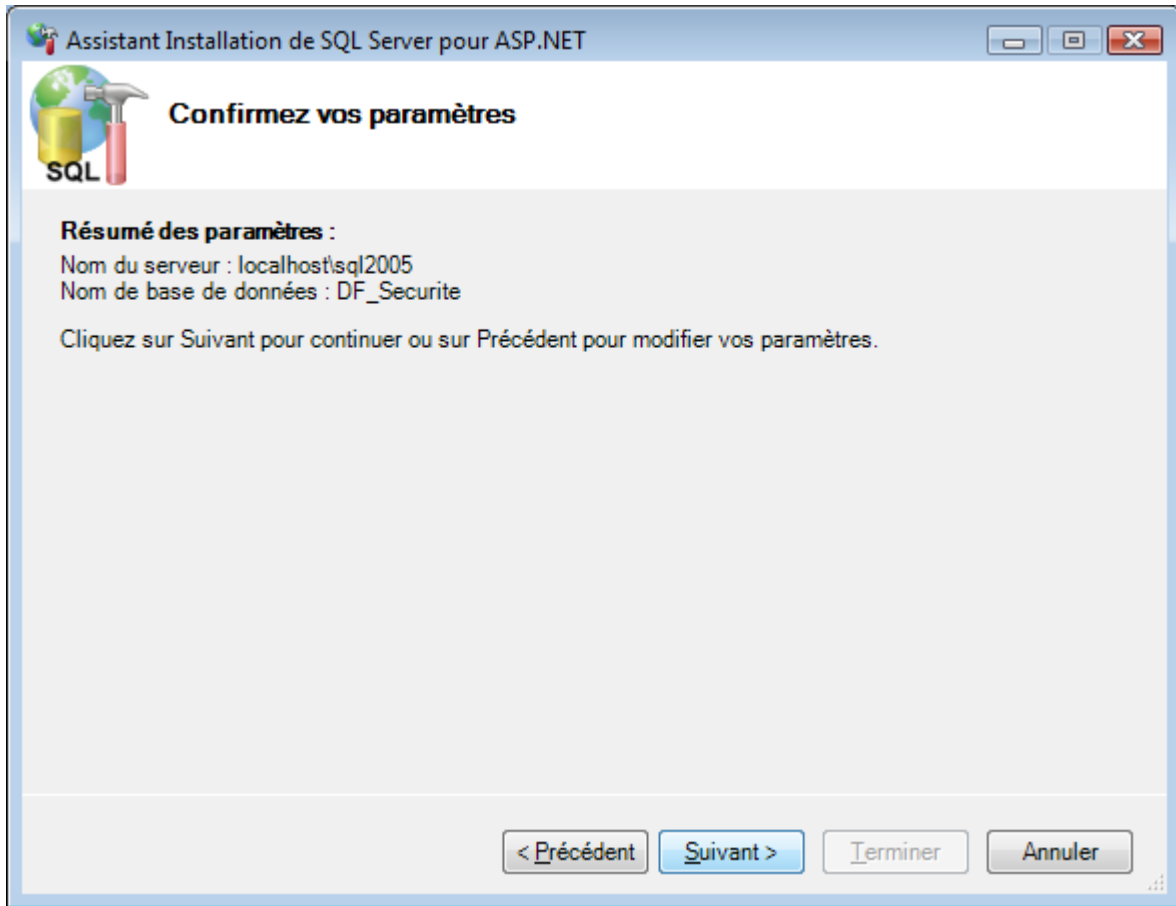
Base de données : DF\_Securite

< Précédent   Suivant >   Terminer   Annuler

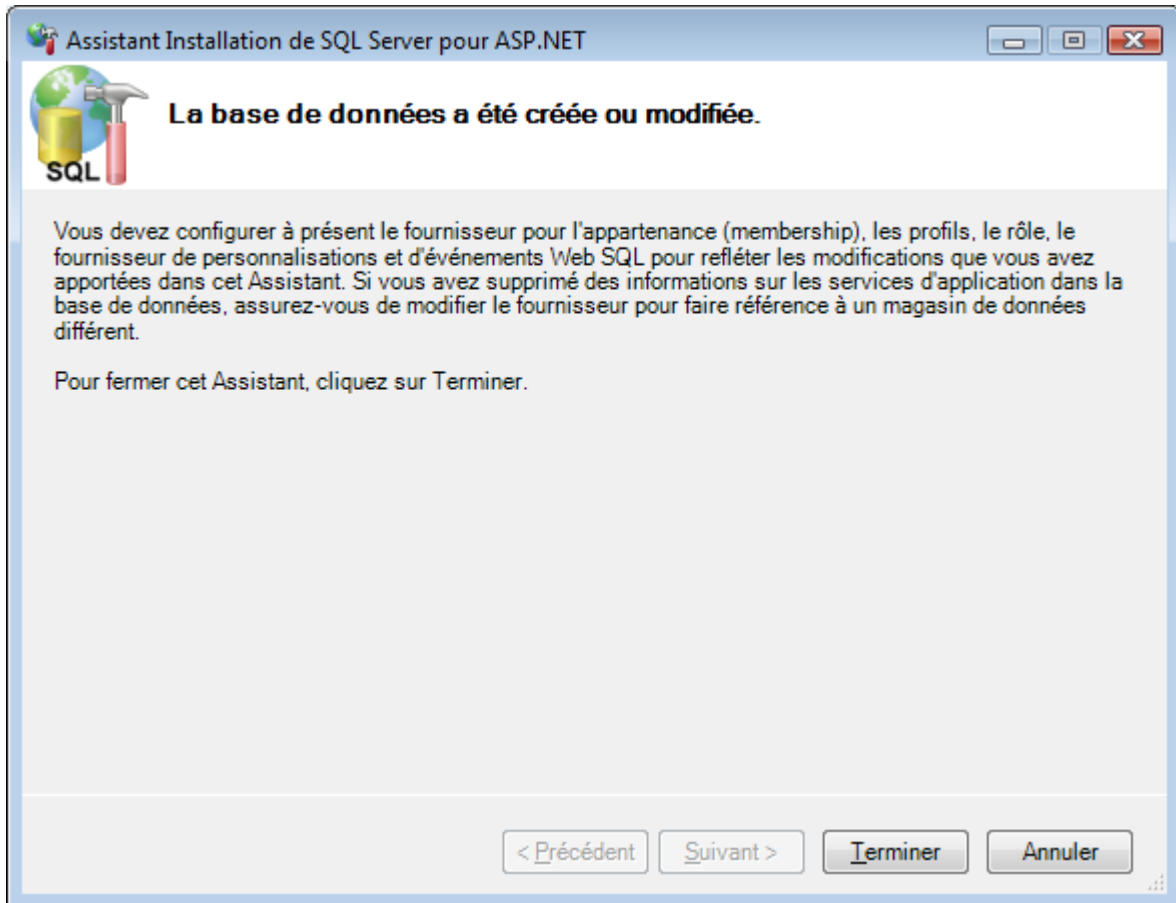
Préciser alors le nom de l'instance SQL Server et le nom de la base de données que vous souhaitez configurer. Vous pouvez choisir une base de données existante, ou alors saisir un nom de base de données. Si elle n'existe pas, elle sera automatiquement créée. Cliquez sur le bouton *Suivant*. La fenêtre suivante apparaît :

**www.Mcours.com**  
Site N°1 des Cours et Exercices Email: [contact@mcours.com](mailto:contact@mcours.com)





Cette fenêtre est une fenêtre récapitulative. Cliquez sur le bouton *Suivant* pour configurer votre base de données. Puis, la fenêtre suivante apparaît :



Cliquez sur le bouton Terminer pour fermer cette fenêtre. Vous pouvez alors ouvrir l'interface d'administration SQL Server Management Studio, et explorer les différents objets SQL qui ont été créés dans votre base de données (tables, vues et procédures stockées). Ces objets SQL seront utilisés par les services d'applications ASP .NET, au travers des providers.

- Soit en ouvrant une fenêtre de commande DOS pour Visual Studio 2008 (simple fenêtre DOS, avec le path bien configuré). Vous y aurez accès dans le répertoire *Visual Studio Tools* du répertoire Visual Studio 2008 du menu Démarrer. En choisissant ce mode de configuration de base de données, vous aurez la possibilité choisir les services d'application que vous souhaitez configurer. Voici une ligne de commande permettant de configurer une base de données, de manière à uniquement utiliser le service d'application de sécurité :

*Dans un fenêtre de commande DOS pour Visual Studio 2008 :*

```
C:\aspnet_regsql -S localhost\sql2005 -E -d Df_Seurite -A mr
```

Voici quelques explications sur cette ligne de commande :

-S : permet de spécifier l'instance SQL Server

-E : indique que l'outil *aspnet\_regsql* utilise l'authentification windows pour se connecter au serveur

-d : permet de spécifier la base de données

-A : permet de spécifier les services d'applications qu'on souhaite configurer :

- m : Membership (utilisateurs)

- r : Role (groupe d'utilisateurs)

Voici un exemple d'exécution de cette commande :

```
C:\>aspnet_regsql -S localhost\sql2005 -E -d Df_Seurite -A mr
Commencez l'ajout des fonctionnalités suivantes :
Membership
RoleManager
...
Terminé.
```

Une fois cette instruction exécutée, vous pouvez observer dans la base de données, SQL Server Management Studio, les différents objets SQL ayant été créés (tables, vues et procédures stockées).

#### 4.4 Définition des providers

Les providers sont des composants utilisés par les services d'application ASP .NET, qui permettent de configurer leur état et comportement. Par exemple ils permettent de spécifier les bases de données SQL Server, que les services d'applications doivent utiliser. Les providers possèdent un jeu de propriétés communes, et des propriétés qui leur sont propres.

Dans notre cas, nous allons définir deux providers :

- Un pour la gestion des utilisateurs (MembershipProvider).
- Un pour les rôles (RoleManager).

Le choix effectué par Microsoft, de proposer deux providers distincts pour les utilisateurs pour les rôles est judicieux. En effet, il permet de stocker les informations des utilisateurs dans une base de données, et les informations sur les rôles dans une autre base de données. Dans notre cas, les deux providers pointeront vers la même base de données. Ainsi, dans le fichier de configuration :

- Ajoutons une chaîne de connexion désignant la base de données (dans l'élément *connectionStrings*) :

*Fichier de configuration*

```
<connectionStrings>
  <add name="CS_DF_Seurite"
    connectionString="data source=localhost\sql2005; initial
      catalog=DF_Seurite; user id=userTest; password=passwd"/>
</connectionStrings>
```

Vous remarquerez que nous utilisons l'authentification SQL Server.

- Dans l'élément `system.web`, ajoutons les éléments suivants :

*Fichier de configuration*

```
<membership defaultProvider="ApplimmaMembership">
  <providers>
    <add connectionStringName="DF_Membership"
      name="DF_Membership"
      type="System.Web.Security.SqlMembershipProvider" />
  </providers>
</membership>

<roleManager enabled="true" defaultProvider="DF_RoleManager">
  <providers>
    <add connectionStringName="CS_DF_Seurite"
      name="DF_RoleManager"
      type="System.Web.Security.SqlRoleProvider" />
  </providers>
</roleManager>
```

## 4.5 Création des rôles et utilisateurs, et définition de leurs droits d'accès sur les ressources de l'application

Lancer l'outil de configuration ASP .NET pour votre application :



### Outil Administration de site Web

**Application** :/DotnetFrance\_Seurite

**Nom de l'utilisateur actuel** :ENIS\JRAVAILLE

<a href="#">Sécurité</a>	Vous permet de définir et de modifier les utilisateurs, les rôles et les autorisations d'accès de votre site. Le site utilise l'authentification Windows pour la gestion des utilisateurs.
<a href="#">Configuration de l'application</a>	Vous permet de gérer les paramètres de configuration de votre application.
<a href="#">Configuration de fournisseur</a>	Vous permet de spécifier où et comment stocker les données d'administration utilisées par votre site Web.

Cliquez sur l'onglet *Fournisseur*. L'écran suivant apparaît :

ASP.net Microsoft Outil Administration de site Web [Comment faire pour utiliser cet outil ?](#) ?

Accueil Sécurité Application **Fournisseur**



Utilisez cette page pour configurer le stockage des données de gestion de site Web, telles que l'appartenance. Vous pouvez utiliser un fournisseur unique pour toutes les données de gestion de votre site ou spécifier un fournisseur différent pour chaque fonctionnalité.

[Sélectionnez un fournisseur unique pour toutes les données de gestion de site](#)  
[Sélectionnez un fournisseur différent pour chaque fonctionnalité \(avancée\)](#)

Cliquez sur le lien hypertext « Sélectionnez un fournisseur différent pour chaque fonctionnalité (avancée) ». L'écran suivant apparaît :

ASP.net Microsoft Outil Administration de site Web [Comment faire pour utiliser cet outil ?](#) ?

Accueil Sécurité Application **Fournisseur**

Utilisez cette page pour sélectionner un fournisseur pour chaque fonctionnalité.

Fournisseur d'appartenances	
<input type="radio"/> AspNetSqlMembershipProvider	<a href="#">Test</a>
<input checked="" type="radio"/> DF_Membership	

Fournisseur de rôles	
<input type="radio"/> AspNetSqlRoleProvider	<a href="#">Test</a>
<input type="radio"/> AspNetWindowsTokenRoleProvider	
<input checked="" type="radio"/> DF_RoleManager	

Si l'écran précédent apparaît, alors les providers définis précédemment dans le fichier de configurations sont syntaxiquement corrects. Vérifier que les deux providers ajoutés sont sélectionnés. Si ce n'est pas le cas, veuillez les sélectionner. Puis, cliquer sur l'onglet *Sécurité*. L'écran suivant apparaît :





**ASP.net** Microsoft Outil Administration de site Web Comment faire pour utiliser cet outil ? ?

Accueil **Sécurité** Application Fournisseur

Vous pouvez utiliser l'outil Administration de site Web pour gérer tous les paramètres de sécurité de votre application. Vous pouvez définir des utilisateurs et des mots de passe (authentification), créer des rôles (groupes d'utilisateurs) et créer des autorisations (règles de contrôle d'accès à différentes parties de votre application).

Par défaut, les informations utilisateur sont stockées dans une base de données Microsoft SQL Server Express dans le dossier Data de votre site Web. Pour stocker les informations utilisateur dans une base de données différente, utilisez l'onglet Fournisseur pour sélectionner un autre fournisseur.

[Utilisez l'Assistant Installation de sécurité pour configurer la sécurité étape par étape.](#)

Cliquez sur les liens dans le tableau pour gérer les paramètres de votre application.

Utilisateurs	Rôles	Règles d'accès
Le type d'authentification actuel est <b>Windows</b> . La gestion des utilisateurs est donc désactivée pour cet outil. <a href="#">Sélectionnez le type d'authentification</a>	Rôles existants : <b>0</b> <a href="#">Désactiver des rôles</a> <a href="#">Créer ou gérer des rôles</a>	<a href="#">Créer des règles d'accès</a> <a href="#">Gérer les règles d'accès</a>

Cliquez sur le premier lien « Sélectionnez le type d'authentification », afin de faire un choix entre l'authentification Windows et l'authentification par formulaire. Dans notre cas, nous faisons le choix de l'authentification par formulaire (« à partir d'internet ») :

**ASP.net** Microsoft Outil Administration de site Web Comment faire pour utiliser cet outil ? ?

Accueil **Sécurité** Application Fournisseur

Comment les utilisateurs accèderont-ils à votre site ?

**À partir d'Internet**

Sélectionnez cette option si les utilisateurs accèdent à votre site Web à partir de l'Internet public. Les utilisateurs devront ouvrir une session à l'aide d'un formulaire Web. Le site utilisera une authentification par formulaire pour identifier les utilisateurs par rapport aux informations stockées dans une base de données.

**À partir d'un réseau local**

Sélectionnez cette option si les utilisateurs accèdent à votre site Web uniquement à partir d'un réseau local privé. Le site utilisera l'authentification intégrée de Microsoft Windows pour identifier les utilisateurs. Seuls les utilisateurs disposant d'un nom d'utilisateur et d'un mot de passe Windows valides seront en mesure d'accéder à votre site.

Terminé



En faisant ce choix, vous devrez définir les utilisateurs. Cliquer sur le bouton « Terminer » pour valider votre choix. La fenêtre suivante apparaît alors :



Vous pouvez utiliser l'outil Administration de site Web pour gérer tous les paramètres de sécurité de votre application. Vous pouvez définir des utilisateurs et des mots de passe (authentification), créer des rôles (groupes d'utilisateurs) et créer des autorisations (règles de contrôle d'accès à différentes parties de votre application).

Par défaut, les informations utilisateur sont stockées dans une base de données Microsoft SQL Server Express dans le dossier Data de votre site Web. Pour stocker les informations utilisateur dans une base de données différente, utilisez l'onglet Fournisseur pour sélectionner un autre fournisseur.

[Utilisez l'Assistant Installation de sécurité pour configurer la sécurité étape par étape.](#)

Cliquez sur les liens dans le tableau pour gérer les paramètres de votre application.

Utilisateurs	Rôles	Règles d'accès
Utilisateurs existants : <b>0</b> <a href="#">Créer un utilisateur</a> <a href="#">Gérer les utilisateurs</a>  <a href="#">Sélectionnez le type d'authentification</a>	Rôles existants : <b>0</b> <a href="#">Désactiver des rôles</a> <a href="#">Créer ou gérer des rôles</a>	<a href="#">Créer des règles d'accès</a> <a href="#">Gérer les règles d'accès</a>

Commençons à créer les rôles en cliquant sur le lien « Créer ou gérer des rôles ». L'écran suivant apparaît :



**ASP.net** Microsoft Outil Administration de site Web [Comment faire pour utiliser cet outil ?](#)

Accueil **Sécurité** Application Fournisseur

Vous pouvez ajouter des rôles ou des groupes qui vous permettent d'autoriser ou d'empêcher des groupes d'utilisateurs d'accéder à des dossiers spécifiques de votre site Web. Par exemple, vous pouvez créer des rôles, tels que "gestion", "ventes" ou "membres", chacun avec un accès à des dossiers spécifiques différents.

#### Créer un nouveau rôle

Nouveau nom de rôle :

Ajouter le rôle

Précédent

Nous allons créer deux rôles :

- Un premier qui se nomme *RoleAdmin*, qui contiendra des utilisateurs qui seront considérés comme administrateurs.
- Un second qui s'appelle *RoleUser*, qui contiendra des utilisateurs qui seront considérés comme simples utilisateurs.

Une fois ces rôles créés, cliquez sur le bouton *Précédent*, afin de revenir à la page principale de la sécurité :

**ASP.net** Microsoft Outil Administration de site Web [Comment faire pour utiliser cet outil ?](#) 

Accueil **Sécurité** Application Fournisseur

Vous pouvez utiliser l'outil Administration de site Web pour gérer tous les paramètres de sécurité de votre application. Vous pouvez définir des utilisateurs et des mots de passe (authentification), créer des rôles (groupes d'utilisateurs) et créer des autorisations (règles de contrôle d'accès à différentes parties de votre application).


Par défaut, les informations utilisateur sont stockées dans une base de données Microsoft SQL Server Express dans le dossier Data de votre site Web. Pour stocker les informations utilisateur dans une base de données différente, utilisez l'onglet Fournisseur pour sélectionner un autre fournisseur.

[Utilisez l'Assistant Installation de sécurité pour configurer la sécurité étape par étape.](#)

Cliquez sur les liens dans le tableau pour gérer les paramètres de votre application.

Utilisateurs	Rôles	Règles d'accès
Utilisateurs existants : <b>0</b> <a href="#">Créer un utilisateur</a> <a href="#">Gérer les utilisateurs</a>  <a href="#">Sélectionnez le type d'authentification</a>	Rôles existants : <b>2</b> <a href="#">Désactiver des rôles</a> <a href="#">Créer ou gérer des rôles</a>	<a href="#">Créer des règles d'accès</a> <a href="#">Gérer les règles d'accès</a>

Cliquer sur le lien hypertext « Gérer les utilisateurs ». L'écran suivant apparaît :

**ASP.net** Microsoft Outil Administration de site Web [Comment faire pour utiliser cet outil ?](#) 

Accueil **Sécurité** Application Fournisseur

Cliquez sur une ligne pour sélectionner un utilisateur, puis cliquez sur **Modifier l'utilisateur** pour afficher ou modifier son mot de passe ou d'autres propriétés. Pour assigner des rôles à l'utilisateur sélectionné, activez les cases à cocher appropriées situées sur la droite.

Pour empêcher un utilisateur de se connecter à votre application tout en conservant ses informations dans votre base de données, désactivez la case à cocher pour définir l'état sur inactif.

**Rechercher des utilisateurs**

Rechercher par : Nom d'utilisateur pour :

Les caractères génériques \* et ? sont autorisés.

[A](#) [B](#) [C](#) [D](#) [E](#) [F](#) [G](#) [H](#) [I](#) [J](#) [K](#) [L](#) [M](#) [N](#) [O](#) [P](#) [Q](#) [R](#) [S](#) [T](#) [U](#) [V](#) [W](#) [X](#) [Y](#) [Z](#) [Tous](#)

Aucun utilisateur n'a été créé.

[Créer un nouvel utilisateur](#)

Cliquez sur le lien hypertext « Créer un nouvel utilisateur ». L'écran suivant apparaît :

**ASP.net** Microsoft Outil Administration de site Web Comment faire pour utiliser cet outil ? ?

Accueil **Sécurité** Application Fournisseur

Ajoutez un utilisateur en entrant l'ID de l'utilisateur, son mot de passe et son adresse de messagerie sur cette page.

**Créer un utilisateur**

Inscrivez-vous pour obtenir votre nouveau compte

Nom d'utilisateur :

Mot de passe :

Confirmer le mot de passe :

Adresse de messagerie :

Question de sécurité :

Réponse de sécurité :

Utilisateur actif

**Rôles**

Sélectionnez des rôles pour cet utilisateur :

RoleAdmin

RoleUser

Nous allons créer deux utilisateurs. Voici les informations nécessaires à leur création :

	Premier utilisateur (simple utilisateur)	Second utilisateur (administrateur)
Nom d'utilisateur	User	Admin
Mot de passe	AZErtY1,	P@\$w0rd
Adresse de messagerie	user@dotnet-france.com	<a href="mailto:admin@dotnet-france.com">admin@dotnet-france.com</a>
Question de sécurité	Ville de naissance	Animal préféré
Réponse de sécurité	Nantes	Chat
Rôle	RoleUser	RoleAdmin

Quelques précisions sur la création de ces utilisateurs :

- Les mots de passe doivent être fortement sécurisés. Ils doivent respecter les règles suivantes :
  - o Ils doivent être constitués d'au moins sept caractères.
  - o Ils doivent contenir au moins un caractère alphabétique minuscule.
  - o Ils doivent contenir au moins un caractère alphabétique majuscule.
  - o Ils doivent contenir au moins un caractère « spécial », tel qu'un caractère de ponctuation.
- Deux utilisateurs ne peuvent avoir le même nom d'utilisateur.
- Deux utilisateurs ne peuvent avoir la même adresse email.
- La question de sécurité est la question qui sera posée à l'utilisateur lorsqu'il demandera son mot de passe. La réponse de sécurité est la réponse attendue.



Une fois ces deux comptes créés, cliquez sur le bouton *Précédent*, afin de revenir à la page principale de la sécurité :



Vous pouvez utiliser l'outil Administration de site Web pour gérer tous les paramètres de sécurité de votre application. Vous pouvez définir des utilisateurs et des mots de passe (authentification), créer des rôles (groupes d'utilisateurs) et créer des autorisations (règles de contrôle d'accès à différentes parties de votre application).

Par défaut, les informations utilisateur sont stockées dans une base de données Microsoft SQL Server Express dans le dossier Data de votre site Web. Pour stocker les informations utilisateur dans une base de données différente, utilisez l'onglet Fournisseur pour sélectionner un autre fournisseur.

[Utilisez l'Assistant Installation de sécurité pour configurer la sécurité étape par étape.](#)

Cliquez sur les liens dans le tableau pour gérer les paramètres de votre application.

Utilisateurs	Rôles	Règles d'accès
Utilisateurs existants : <b>2</b> <a href="#">Créer un utilisateur</a> <a href="#">Gérer les utilisateurs</a>  <a href="#">Sélectionnez le type d'authentification</a>	Rôles existants : <b>2</b> <a href="#">Désactiver des rôles</a> <a href="#">Créer ou gérer des rôles</a>	<a href="#">Créer des règles d'accès</a> <a href="#">Gérer les règles d'accès</a>

Maintenant, une fois les rôles et utilisateurs définis, nous allons définir les règles d'accès, c'est-à-dire leurs autorisations et interdictions sur les ressources de l'application. Pour ce faire, cliquez sur le lien hypertext « Gérer les règles d'accès ». L'écran suivant apparaît :



Utilisez cette page pour gérer les règles d'accès de votre site Web. Les règles s'appliquent dans l'ordre. La première règle qui correspond s'applique et l'autorisation de chaque règle a priorité sur les autorisations de toutes les règles suivantes. Utilisez les boutons **Monter** et **Descendre** pour modifier la position de la règle sélectionnée.

Les règles grisées sont héritées d'un parent et ne peuvent pas être modifiées à ce niveau.



Terminé

Les règles d'accès se définissent sur les répertoires de l'application. Un répertoire hérite des règles d'accès de son répertoire parent. Toutefois, il peut aussi :

- Renforcer la sécurité, en définissant des règles d'accès supplémentaires, plus restrictives.
- Au contraire l'alléger, en définissant des règles d'accès supplémentaires, plus permissives.

Dans notre cas, nous allons définir les règles de sécurité suivantes :

- A la racine de l'application, tous les utilisateurs doivent être authentifiés pour accéder aux ressources de l'application. Autrement dit, les utilisateurs anonymes sont interdits. Les répertoires Administrateurs, Utilisateurs et Commun héritent de cette règle d'accès. Pour définir cette règle d'accès, se positionner sur le répertoire racine de l'application (DotnetFrance\_Securite), et cliquer sur le lien hypertext « Ajouter une nouvelle règle d'accès ». L'écran suivant apparaît :

**Ajouter une nouvelle règle d'accès**

Sélectionnez un répertoire pour cette règle :	La règle s'applique à :	Autorisation :
<ul style="list-style-type: none"> <li><input checked="" type="checkbox"/> DotnetFrance_Securite               <ul style="list-style-type: none"> <li><input type="checkbox"/> Administrateurs</li> <li><input type="checkbox"/> App_Data</li> <li><input type="checkbox"/> Commun</li> <li><input type="checkbox"/> Utilisateurs</li> </ul> </li> </ul>	<input type="radio"/> Rôle <input type="text" value="RoleAdmin"/>  <input type="radio"/> utilisateur <input type="text"/> <a href="#">Rechercher des utilisateurs</a>  <input type="radio"/> Tous les utilisateurs <input checked="" type="radio"/> Utilisateurs anonymes	<input type="radio"/> Autoriser  <input checked="" type="radio"/> Refuser

Choisissez alors de refuser les utilisateurs anonymes, et validez votre choix.

- Sur le répertoire Administrateurs, nous allons renforcer la sécurité : seuls les utilisateurs appartenant au rôle RoleAdmin, peuvent accéder aux ressources de ce répertoire. Pour définir cette règle d'accès, se positionner sur le répertoire Administrateurs, et cliquer sur le lien hypertext « Ajouter une nouvelle règle d'accès ». L'écran suivant apparaît :

**Ajouter une nouvelle règle d'accès**

Sélectionnez un répertoire pour cette règle :	La règle s'applique à :	Autorisation :
<ul style="list-style-type: none"> <li><input checked="" type="checkbox"/> DotnetFrance_Securite               <ul style="list-style-type: none"> <li><input checked="" type="checkbox"/> Administrateurs</li> <li><input type="checkbox"/> App_Data</li> <li><input type="checkbox"/> Commun</li> <li><input type="checkbox"/> Utilisateurs</li> </ul> </li> </ul>	<input checked="" type="radio"/> Rôle <input type="text" value="RoleAdmin"/>  <input type="radio"/> utilisateur <input type="text"/> <a href="#">Rechercher des utilisateurs</a>  <input type="radio"/> Tous les utilisateurs <input type="radio"/> Utilisateurs anonymes	<input checked="" type="radio"/> Autoriser  <input type="radio"/> Refuser

Autorisez seuls les utilisateurs appartenant au rôle RoleAdmin, à accéder aux ressources de ce répertoire. Puis ajoutez une seconde règle d'accès, permettant de refuser tous les utilisateurs :



### Ajouter une nouvelle règle d'accès

#### Sélectionnez un répertoire pour cette règle :

- DotnetFrance\_Securit
  - Administrateurs
  - App\_Data
  - Commun
  - Utilisateurs

#### La règle s'applique à :

Rôle

utilisateur

[Rechercher des utilisateurs](#)

Tous les utilisateurs

Utilisateurs anonymes

#### Autorisation :

Autoriser

Refuser

A la suite de création de ces deux règles, on obtient l'écran suivant :

### Gérer les règles d'accès

- DotnetFrance\_Securit
  - Administrateurs
  - App\_Data
  - Commun
  - Utilisateurs

Autorisation	Utilisateurs et rôles	Supprimer
Autoriser	RoleAdmin	<a href="#">Supprimer</a>
Refuser	[tous]	<a href="#">Supprimer</a>
Autoriser	[tous]	<a href="#">Supprimer</a>

[Ajouter une nouvelle règle d'accès](#)

Monter

Descendre

L'ordre des règles est très important. En effet, lors de l'accès à une ressource, les règles d'accès sont analysées séquentiellement. Dès qu'une règle rencontrée est satisfaisante, elle est appliquée et les règles suivantes sont automatiquement ignorées. Ainsi, si les règles sont inversées, alors aucun utilisateur ne pourra accéder aux ressources du répertoire Administrateurs.

- Sur le répertoire Utilisateurs, pas règle d'accès particulière par rapport aux règles héritées.
- Sur le répertoire Commun, nous allons contredire la sécurité héritée. Tous les utilisateurs, qu'ils soient authentifiés ou non, peuvent accéder aux ressources de ce répertoire. Ajoutez alors la règle d'accès suivante : autoriser les utilisateurs anonymes.

### Ajouter une nouvelle règle d'accès

#### Sélectionnez un répertoire pour cette règle :

- DotnetFrance\_Securit
  - Administrateurs
  - App\_Data
  - Commun
  - Utilisateurs

#### La règle s'applique à :

Rôle

utilisateur

[Rechercher des utilisateurs](#)

Tous les utilisateurs

Utilisateurs anonymes

#### Autorisation :

Autoriser

Refuser

## 4.6 Création de l'application

Une fois notre politique de sécurité définie, nous allons utiliser les contrôles ASP .NET de sécurité.

Par défaut, la page Login.aspx située à la racine de l'application est le fichier de connexion par défaut. Dans cette page, ajoutons le contrôle Login :

*C# et VB .NET*

```
<form id="form1" runat="server">
  <div>
    <asp:Login ID="Login1" runat="server">
    </asp:Login>
  </div>
</form>
```

Le résultat d'affichage de ce code est le suivant :

Se connecter

Nom d'utilisateur :

Mot de passe :

Mémoriser le mot de passe.

Dans la page DemandeMotDePasse.aspx, ajoutons un contrôle de type PasswordRecovery :

*C# et VB .NET*

```
<form id="form1" runat="server">
  <div>
    <asp>PasswordRecovery ID="PasswordRecovery1" runat="server">
    </asp>PasswordRecovery>
  </div>
</form>
```

Le résultat d'affichage de ce code est le suivant :

Vous avez oublié votre mot de passe ?  
Entrez votre nom d'utilisateur pour recevoir votre mot de passe.

Nom d'utilisateur :

Ces contrôles de sécurité, comme tous ceux présents dans l'onglet Connexion de la boîte à outils, utilisent nativement le service d'application ASP .NET gérant la sécurité.

## 4.7 Utilisation de l'application

Lors de l'accès à une ressource de l'application, ASP .NET regarde si la requête HTTP contient un cookie d'authentification. S'il est présent et valide, alors l'accès est autorisé. Dans le cas échéant, l'utilisateur est automatiquement renvoyé vers la page de connexion (dans notre cas, Login.aspx). L'utilisateur doit alors s'authentifier. S'il réussit, alors il sera automatiquement redirigé vers la ressource initialement demandée. Puis une réponse sera renvoyée au client, avec un cookie d'identification, qui sera automatiquement envoyé par le client dans les requêtes http suivantes.

Le schéma ci-dessous illustre ces propos :

① Le client envoie une requête HTTP au serveur IIS, pour accéder à une ressource. Cette ressource est protégée.

② Le processus ASP.NET regarde si la requête regarde si cette requête contient un cookie d'identification valide.

S'il est valide : ③ Alors l'accès à la ressource est autorisé.

Le cas échéant :

④ L'utilisateur est automatiquement redirigé vers la page de connexion de l'application. L'utilisateur tente de se connecter en fournissant un nom d'utilisateur et un mot de passe :

⑤ Si l'authentification réussie, alors l'accès à la ressource est autorisé, ⑦ et l'utilisateur est automatiquement redirigé vers l'application.

⑥ Si l'authentification échoue, alors l'accès est refusé.



Vous pouvez maintenant utiliser l'application, pour vérifier ce scénario, et vérifier les ressources accessibles en fonction de l'utilisateur avec lequel on se connecte.



## 5 Conclusion

Après ce chapitre nous connaissons les différents types de connexion (Windows, formulaire, Passport) et savons les configurer (à l'exception de Passport). Nous savons aussi comment gérer les autorisations pour des fichiers ou des dossiers spécifiques avec la balise *authorization* dans le Web.config à l'aide d'une autre balise : *location*.

Comme nous l'avons vu authentifier des utilisateurs et gérer les autorisations est très important pour un site Web. On pourrait prendre l'exemple d'un site de vente de produit qui à besoin que l'utilisateur soit enregistré avant de pouvoir commencer à acheter des produits, ou un site gérant des services ayant besoin d'administrateurs secondaires pour pouvoir valider, modifier les services (ils ont des privilèges que les autres utilisateurs enregistrés et authentifiés n'auront pas).

