



« Authentication API »

Microsoft .NET

System.Web.Security/Mvc

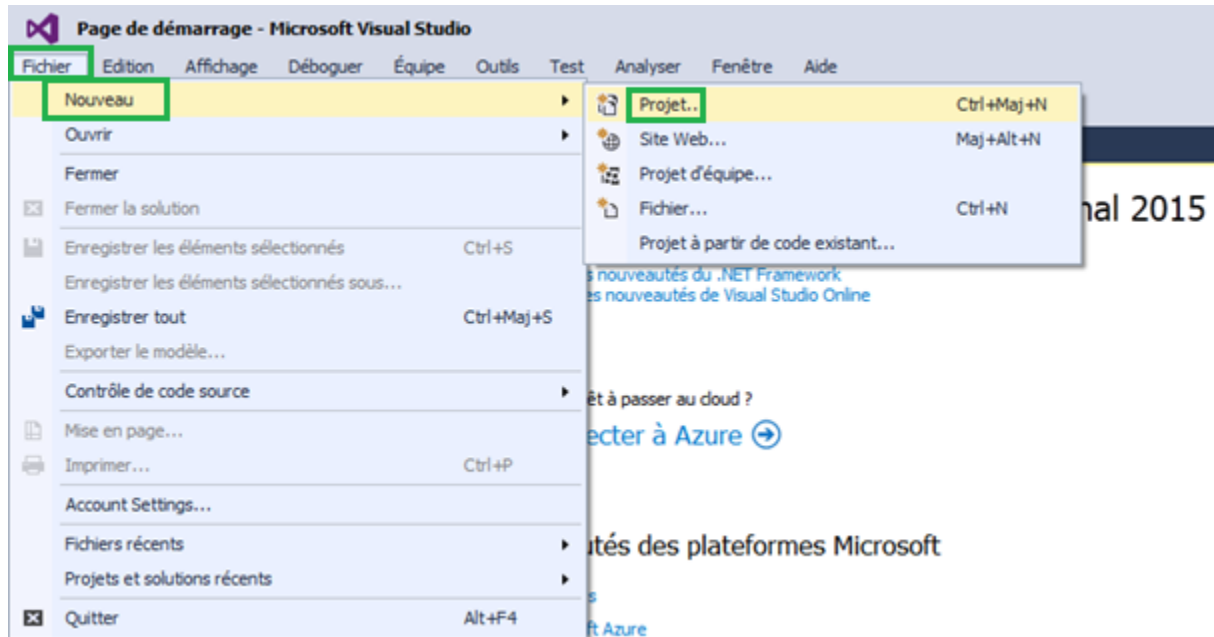
Table des matières

I-Préparation de l'application.....	3
1-Création du projet.....	3
2- Suppression des fichiers inutiles	6
II-Réalisation d'authentification.....	10
III-Démonstration	37

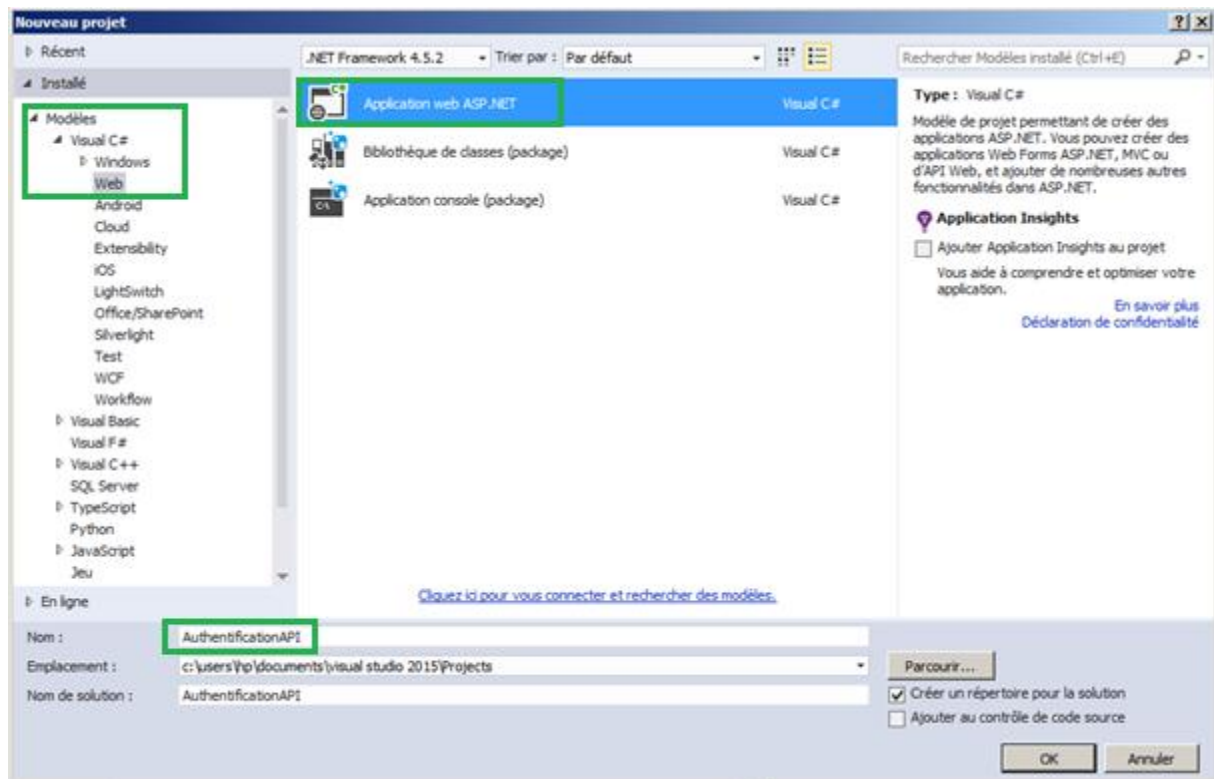
I-Préparation de l'application :

1- Création du projet : (Visual Studio2015)

Pour créer votre projet veuillez suivre les étapes indiquées en bas :

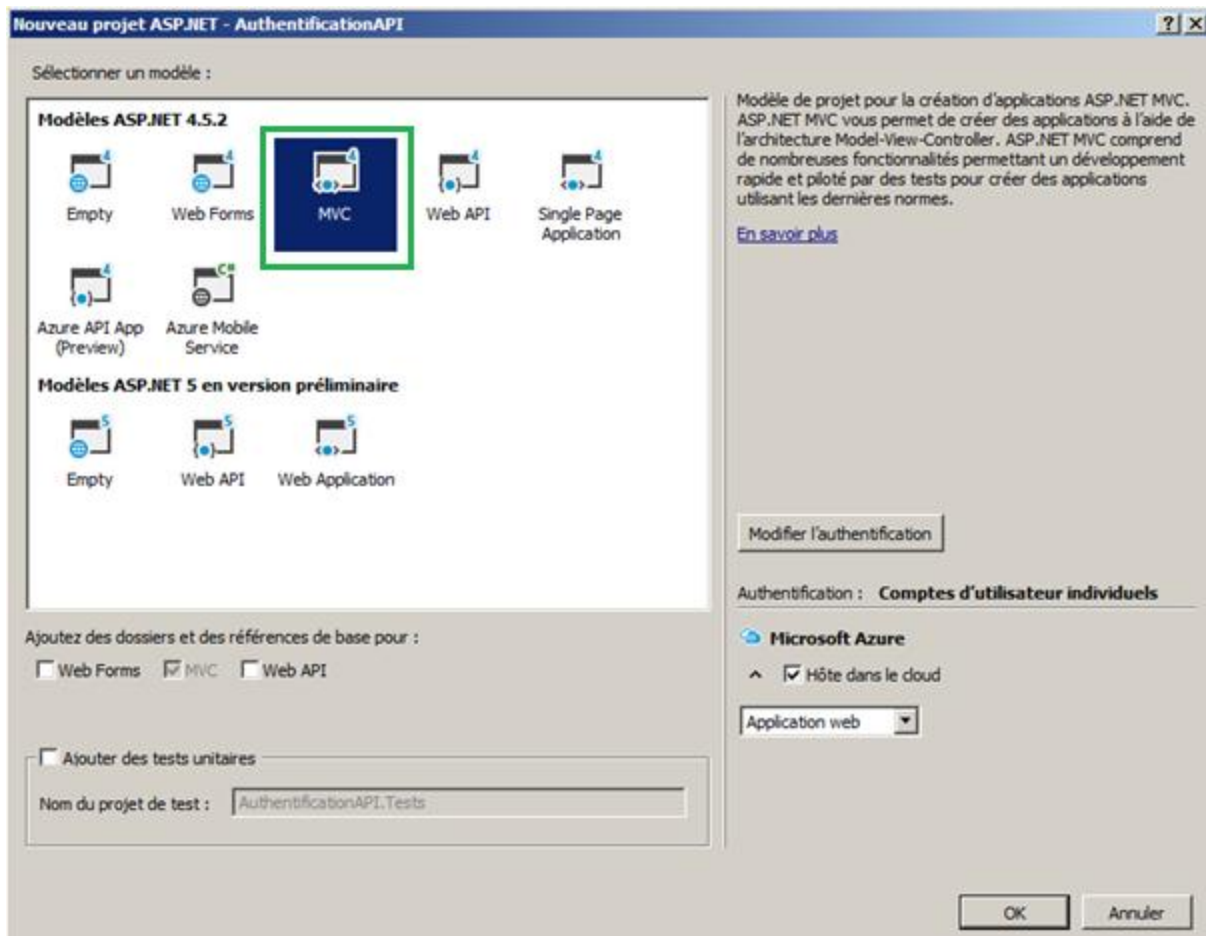


Puis :

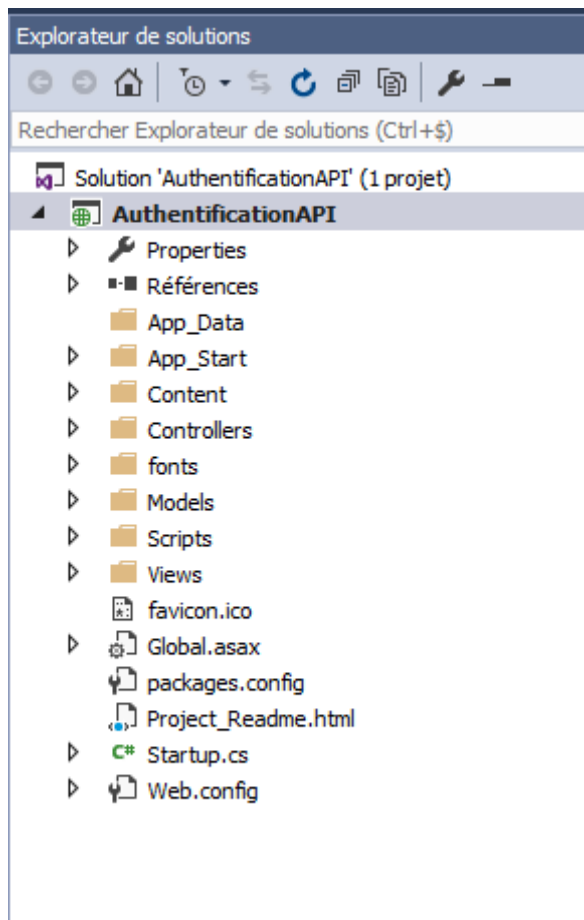


Entrez le nom du projet et appuyez sur OK.

Choisissez MVC et appuyez sur OK.



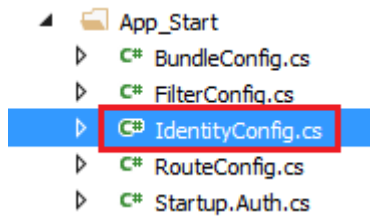
L'arborescence du projet se présente comme suit :



2- Suppression des fichiers inutiles :

Certaines modifications sont indispensables :

a- Supprimer le fichier cadré en rouge en bas :



b- Modifier le fichier suivant :

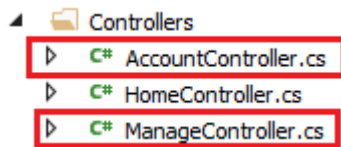
Le fichier Startup.Auth.cs doit être comme suit :

```
using System;
using Microsoft.AspNet.Identity;
using Microsoft.AspNet.Identity.Owin;
using Microsoft.Owin;
using Microsoft.Owin.Security.Cookies;
using Microsoft.Owin.Security.Google;
using Owin;
using AuthenticationAPI.Models;

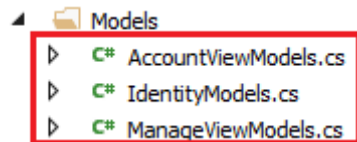
namespace AuthenticationAPI
{
    public partial class Startup
    {
        // Startup
    }
}
```

c- Ensuite supprimer les fichiers et les dossiers cadrés en rouge indiquées en bas :

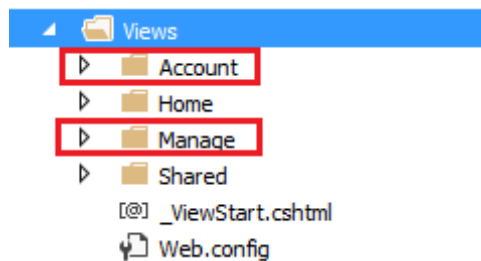
Dans le dossier Controllers :



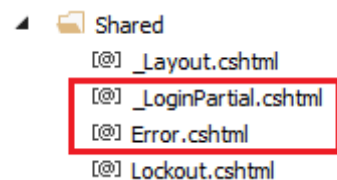
Dans le dossier Models :



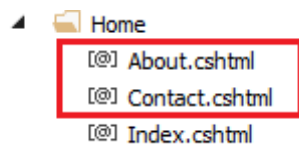
Dans le dossier Views :



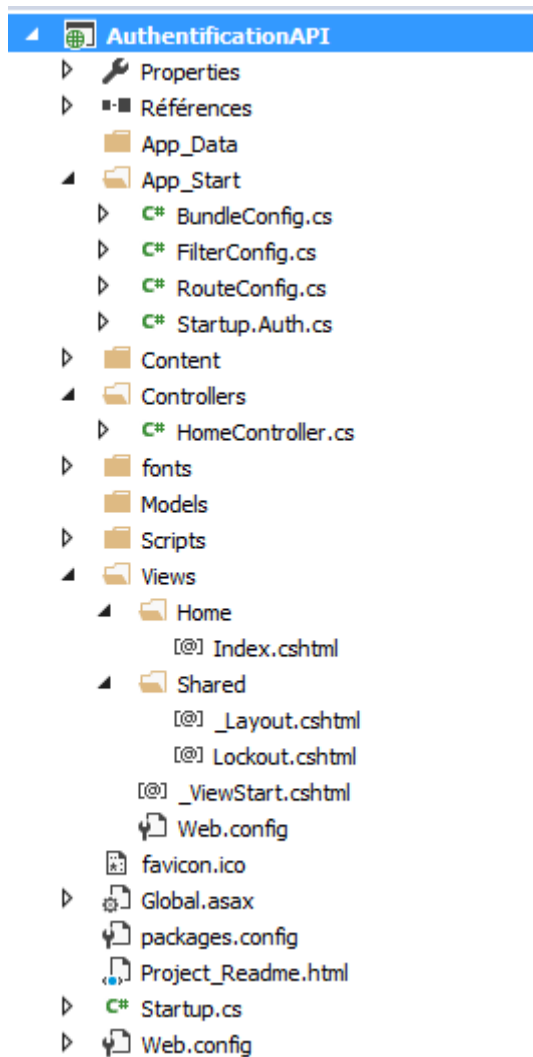
Dans le dossier Shared :



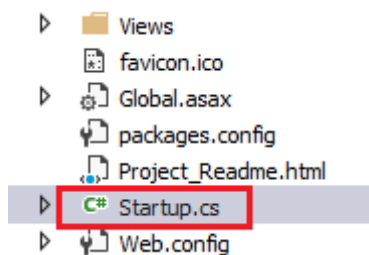
Dans le dossier Home :



d- L'arborescence se résume en ce qui suit :



e- Modifier le fichier suivant :



Le fichier Startup.cs doit être comme suit :

```
using Microsoft.Owin;
using Owin;

[assembly: OwinStartupAttribute(typeof(AuthenticationAPI.Startup))]
namespace AuthenticationAPI
{
    public partial class Startup
    {
    }
```



```
public void Configuration(IApplicationBuilder app)
{
    //ConfigureAuth(app);
}
}
```

Note :

« Les autres dossiers et fichiers (ainsi que leurs contenue) restent inchangeables, seul les modifications siglés dans ce document sont nécessaire pour aboutir à la fin de la réalisation de l'exemple. »

« L'installation du Visual Studio 2015 et SQLServer2008/2014, n'est pas prise en considération dans ce document. Le développeur doit être capable d'installer et de lancer ces outils »

« L'auteur de ce document considère que le lecteur est capable préalablement de programmer avec le langage C#, HTML5 et CSS3 »

« Aussi, la connaissance du modèle MVC est supposé acquise par les lecteurs »

« Pour toutes informations supplémentaires, le lecteur peut tout simplement se référé à Google pour avoir le détail sur les technologies utilisées »

II- Réalisation d'authentification :

Pour développer votre exemple veuillez suivre les étapes indiquées en bas :

- 1- Allez dans le dossier Views/Home et dans la page index.cshtml insérer le code suivant :

```
@{
    ViewBag.Title = "Home Page";
}<div id="Corps" class="container body-content">
    <h1>Je suis indexe !!!</h1>

</div>
```

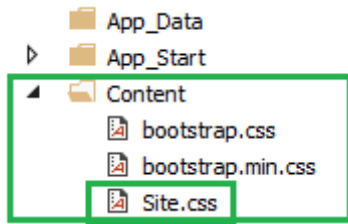
- 2- Aller dans le dossier Controllers et dans le fichier HomeController.cs insérer le code suivant :

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.Mvc;

namespace AuthentificationAPI.Controllers
{
    public class HomeController : Controller
    {
        [Authorize]
        public ActionResult Index()
        {
            ViewBag.Message = "welcome you are connected!! ::: ";

            return View();
        }
    }
}
```

3- Allez dans le dossier Content et dans le fichier Site.css insérer le code suivant :



Site.css :

```
*{
padding:0px;
margin-left: 0px;
margin-right: 0px;
}
#Bandeau
{
width:1500px;
height:30px;
background-color:#000000;
}
#Fonctionnalites
{
width:200px;
height:800px;
background-color:#bcb7ff;
position:relative;
float:left;
padding:5px;
}
#Corps
{background-color:#ffffff;
width:777px;
height:500px;
padding:5px;
position:center;
float:left;
top: 4px;
left: 3px;
}

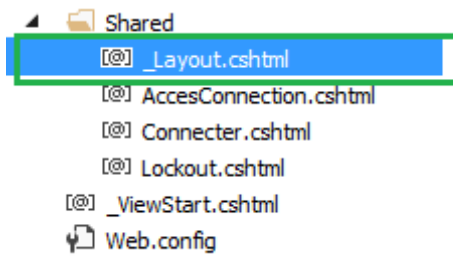
/*****
*** Liste de puces.
*****/
ul
{
padding-left:20px;
}
/*****
*** Sélecteurs d'éléments.
*****/
h1
{
padding-bottom:25px;
text-align:center;
}
h2
{
margin-bottom:16px;
```

```

text-align:center;
}
/***** Alignements.
*****/
.CentrerConteneur
{
margin-left: auto;
margin-right: auto;
}

```

4- Aller dans le dossier Shared :



Et copiez le code en bas dans le fichier _Layout.cshtml :

```

<!DOCTYPE html>
<html>
<head>
  <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
  <meta charset="utf-8" />
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>@ViewBag.Title - Mon application ASP.NET</title>
  @Styles.Render("~/Content/css")
</head>

<body>
  <div id="Bandeau" class="navbar navbar-inverse navbar-fixed-top">
    <div class="container">
      <div class="navbar-collapse collapse">
        <ul class="nav navbar-nav">
          <li>@Html.ActionLink("Accueil", "Index", "Home")</li>
          <li>@Html.ActionLink("À propos de", "About2", "Home")</li>
          <li>@Html.ActionLink("Contact", "Contact", "Home")</li>
          @Html.Partial("AccesConnection")
        </ul>
      </div>
    </div>
  </div>

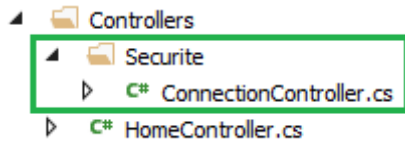
  <div id="Fonctionnalites" class="container body-content">
    Fonctionnalités :
    <hr />
    <footer>
      <p>&copy; @DateTime.Now.Year - My ASP.NET Application</p>
    </footer>
  </div>

```

```
<div id="Corps" class="container body-content">
  @RenderBody()
  <hr />
</div>

@Scripts.Render("~/bundles/jquery")
@Scripts.Render("~/bundles/bootstrap")
@RenderSection("scripts", required: false)
</body>
</html>
```

- 5- Allez dans le dossier Controllers et créer le dossier Securite et le fichier ConnectionController.cs comme suit :



Insérer dans le fichier ConnectionController. Le code suivant :

```
using AuthenticationAPI.Securite;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.Mvc;

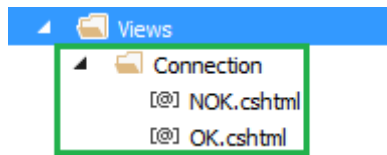
namespace AuthenticationAPI.Controllers.Securite
{
    public class ConnectionController : Controller
    {
        //
        // GET: /Connexion/
        private FormsAuthenticationService FormsAuth
        {
            get;
            set;
        }
        private ClientMembershipService MembershipService
        {
            get;
            set;
        }
        public ConnectionController()
        {
            this.FormsAuth = new FormsAuthenticationService();
            this.MembershipService = new ClientMembershipService();
        }

        public ActionResult Connecter()
        {
            return View();
        }
        public ActionResult OK()
        {
            ViewBag.Message = User.Identity.Name;
            return View();
        }
        public ActionResult NOK()
        {
            ViewBag.Message = "You are disconnected !!!";
            return View();
        }

        [AcceptVerbs(HttpVerbs.Post)]
        public ActionResult Connecter(string Reference, string MotDePasse)
        {
            bool bClientExiste;
            ActionResult oActionResult = this.View();
            bClientExiste = MembershipService.CompteExiste(Reference, MotDePasse);
        }
    }
}
```

```
        if (bClientExiste)
        {
            FormsAuth.Connector(Reference, false);
            oActionResult = this.RedirectToAction("OK", "Connection");
        }
        else
        {
            ModelState.AddModelError("_FORM", "Référence ou mot de passe
incorrect.");
        }
        return oActionResult;
    }
    public ActionResult Deconnecter()
    {
        FormsAuth.Deconnecter();
        return RedirectToAction("NOK", "Connection");
    }
}
```

- 6- Allez dans le dossier Views et créer le dossier Connection avec les deux pages NOK.cshtml et OK.cshtml comme suit :



Le code des eux page est comme suit :

4-1- OK.cshtml :

```
@{
    ViewBag.Title = "ok Page";
}

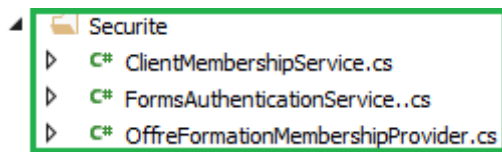
<div id="Corps" class="container body-content">
    <h1></h1>
    <h1>I am OK !!!</h1>
    <h2>You are connected by the login : @ViewBag.Message</h2>
</div>
```

4-2- NOK.cshtml :

```
@{
    ViewBag.Title = "nok Page";
}

<div id="Corps" class="container body-content">
    <h1></h1>
    <h1>Je suis nok !!!</h1>
    <h2> @ViewBag.Message</h2>
</div>
```


7- Créer le dossier Securite à la racine du projet et créer les trois fichiers suivant :



Avec le code en bas :

5-1- ClientMembershipService.cs:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.Security;

namespace AuthentificationAPI.Securite
{
    public class ClientMembershipService
    {
        private MembershipProvider Provider
        {
            get;
            set;
        }
        public ClientMembershipService()
        {
            Provider = Membership.Provider;
        }
        public bool CompteExiste(string aReference, string aMotDePasse)
        {
            return Provider.ValidateUser(aReference, aMotDePasse);
        }
    }
}
```

5-2- FormsAuthenticationService.cs:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.Security;

namespace AuthentificationAPI.Securite
{
    public class FormsAuthenticationService
    {
        public void Connector(string aReference, bool createPersistentCookie)
        {
            FormsAuthentication.SetAuthCookie(aReference, createPersistentCookie);
        }
        public void Deconnector()
        {
            FormsAuthentication.SignOut();
        }
    }
}
```

5-3- OffreFormationMembershipProvider.cs:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.Security;

namespace AuthentificationAPI.Securite
{
    public class OffreFormationMembershipProvider : MembershipProvider
    {
        public override string ApplicationName
        {
            get
            {
                throw new NotImplementedException();
            }
            set
            {
                throw new NotImplementedException();
            }
        }

        public override bool EnablePasswordReset
        {
            get
            {
                throw new NotImplementedException();
            }
        }

        public override bool EnablePasswordRetrieval
        {
            get
            {
                throw new NotImplementedException();
            }
        }

        public override int MaxInvalidPasswordAttempts
        {
            get
            {
                throw new NotImplementedException();
            }
        }

        public override int MinRequiredNonAlphanumericCharacters
        {
            get
            {
                throw new NotImplementedException();
            }
        }

        public override int MinRequiredPasswordLength
        {
            get
            {
                throw new NotImplementedException();
            }
        }
    }
}
```

```

    }
}

public override int PasswordAttemptWindow
{
    get
    {
        throw new NotImplementedException();
    }
}

public override MembershipPasswordFormat PasswordFormat
{
    get
    {
        throw new NotImplementedException();
    }
}

public override string PasswordStrengthRegularExpression
{
    get
    {
        throw new NotImplementedException();
    }
}

public override bool RequiresQuestionAndAnswer
{
    get
    {
        throw new NotImplementedException();
    }
}

public override bool RequiresUniqueEmail
{
    get
    {
        throw new NotImplementedException();
    }
}

public override bool ChangePassword(string username, string oldPassword,
string newPassword)
{
    throw new NotImplementedException();
}

public override bool ChangePasswordQuestionAndAnswer(string username, string
password, string newPasswordQuestion, string newPasswordAnswer)
{
    throw new NotImplementedException();
}

public override MembershipUser CreateUser(string username, string password,
string email, string passwordQuestion, string passwordAnswer, bool isApproved, object
providerUserKey, out MembershipCreateStatus status)
{
    throw new NotImplementedException();
}

```

```

    public override bool DeleteUser(string username, bool deleteAllRelatedData)
    {
        throw new NotImplementedException();
    }

    public override MembershipUserCollection FindUsersByEmail(string emailToMatch,
int pageIndex, int pageSize, out int totalRecords)
    {
        throw new NotImplementedException();
    }

    public override MembershipUserCollection FindUsersByName(string
usernameToMatch, int pageIndex, int pageSize, out int totalRecords)
    {
        throw new NotImplementedException();
    }

    public override MembershipUserCollection GetAllUsers(int pageIndex, int
pageSize, out int totalRecords)
    {
        throw new NotImplementedException();
    }

    public override int GetNumberOfUsersOnline()
    {
        throw new NotImplementedException();
    }

    public override string GetPassword(string username, string answer)
    {
        throw new NotImplementedException();
    }

    public override MembershipUser GetUser(string username, bool userIsOnline)
    {
        throw new NotImplementedException();
    }

    public override MembershipUser GetUser(object providerUserKey, bool
userIsOnline)
    {
        throw new NotImplementedException();
    }

    public override string GetUserNameByEmail(string email)
    {
        throw new NotImplementedException();
    }

    public override string ResetPassword(string username, string answer)
    {
        throw new NotImplementedException();
    }

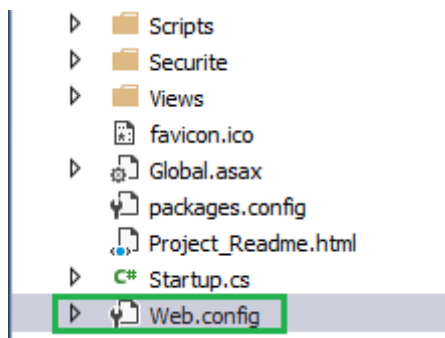
    public override bool UnlockUser(string userName)
    {
        throw new NotImplementedException();
    }

    public override void UpdateUser(MembershipUser user)
    {
        throw new NotImplementedException();
    }

```

```
    }  
    public override bool ValidateUser(string username, string password)  
    {  
        return AuthenticationAPI.Models.utilisateur.GetInstance(username,  
password);  
        throw new NotImplementedException();  
    }  
}
```

8- Aller dans Web.config :



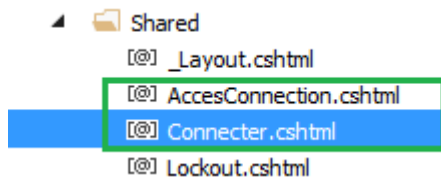
Et modifier comme suit la balise <system.web>

```
<system.web>
  <membership defaultProvider="CustomMembershipProvider">
    <providers>
      <clear />
      <add name="CustomMembershipProvider"
type="AuthenticationAPI.Securite.OffreFormationMembershipProvider" />
    </providers>
  </membership>
  <authentication mode="Forms">
    <forms loginUrl="~/Connection/Connecter" />
  </authentication>
  <compilation debug="true" targetFramework="4.5.2" />
  <httpRuntime targetFramework="4.5.2" />
</system.web>
```

Et la balise <system.webserver>

```
<system.webServer>
</system.webServer>
```

9- Allez dans le dossier Views/Shared et ajouter les deux pages suivant :



Avec les codes :

7-1- AccesConnection.cshtml :

```
@using Microsoft.AspNet.Identity
@if (Request.IsAuthenticated)
{

    <ul class="nav navbar-nav navbar-right">
        <li>@Html.ActionLink("Mon compte", "Compte", "Connexion")</li>
        <li>@Html.ActionLink("Déconnexion", "Deconnecter", "Connection")</li>
    </ul>
}

else
{
    <ul class="nav navbar-nav navbar-right">
        <li>@Html.ActionLink("Connexion", "Connecter", "Connection")</li>
    </ul>
}
```


7-2- Connector.cshtml :

```
@{
    ViewBag.Title = "Home Page";
}

<div id="Corps" class="jumbotron">
    <h1></h1>
    <h1>Je me connecte !!!</h1>
    <fieldset style="left:350px;border:6px ridge
gray;position:fixed;width:550px;height:200px;background-color:#ffffff;">
        @Html.ValidationSummary("", new { style = "width:100%;text-align:center;
margin-top:25px; " })
        @using (Html.BeginForm())
        {
            <table style="width:400px;text-align:left;"
                class="CentrerConteneur">
                <tr>
                    <td>
                        <label for="Reference">Login:</label>
                    </td>
                    <td>
                        @Html.TextBox("Reference")
                    </td>
                </tr>
                <tr>
                    <td>
                        <label for="MotDePasse">Passe:</label>
                    </td>
                    <td>
                        @Html.Password("MotDePasse")
                    </td>
                </tr>
                <tr>
                    <td colspan="2" style="text-align:right;">
                        <input type="submit" value="Valider" style="text-align:right;"
/>
                    </td>
                </tr>
            </table>
        }
    </fieldset>
</div>

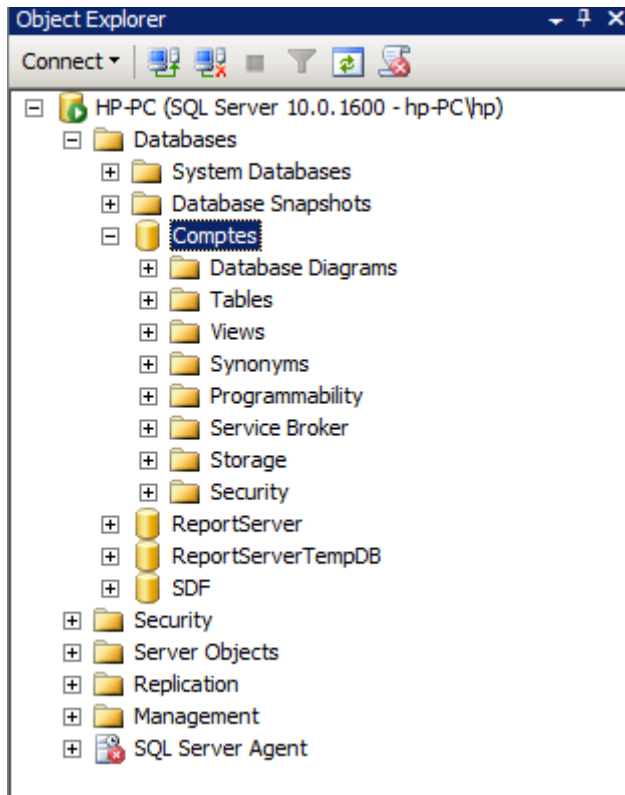
<div class="row">

</div>
```

10- Création du modèle des données:

8-1- Création de la base de données SQLServer:(2008 et 2014)

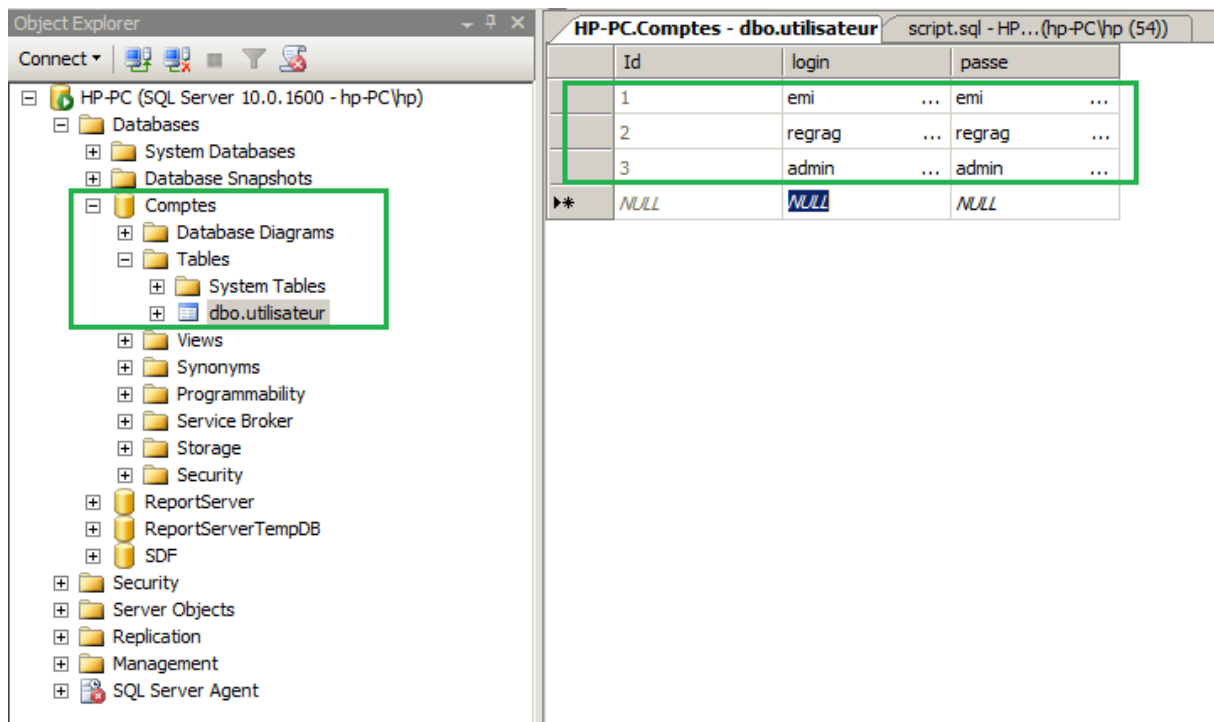
a- Lancez SQLServer et créer la base des données nommé Comptes comme en bas :



b- Le script de création de la base est comme suit :

```
USE [Comptes]
GO
/***** Object: Table [dbo].[utilisateur]      Script Date:
05/06/2018 11:01:54 *****/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[utilisateur](
    [Id] [int] IDENTITY(1,1) NOT NULL,
    [login] [nvarchar](100) NOT NULL,
    [passe] [nvarchar](100) NOT NULL,
    CONSTRAINT [PK_utilisateur] PRIMARY KEY CLUSTERED
(
    [Id] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF,
IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON)
ON [PRIMARY]
) ON [PRIMARY]
GO
```

c- Ajouter un ou plusieurs utilisateurs :

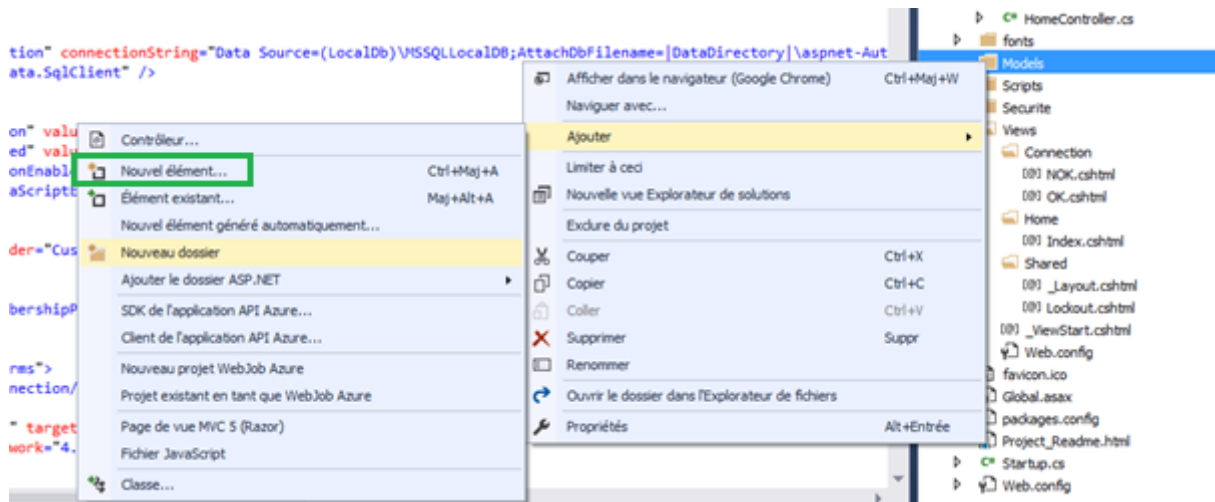


The screenshot shows the SQL Server Enterprise Manager interface. On the left, the 'Object Explorer' pane displays the hierarchy of the 'HP-PC' server. The 'Databases' folder is expanded, and the 'Comptes' database is selected. The 'dbo.utilisateur' table is highlighted with a green border. On the right, the 'script.sql' pane shows the table's data. The table has three columns: 'Id', 'login', and 'passe'. The data is as follows:

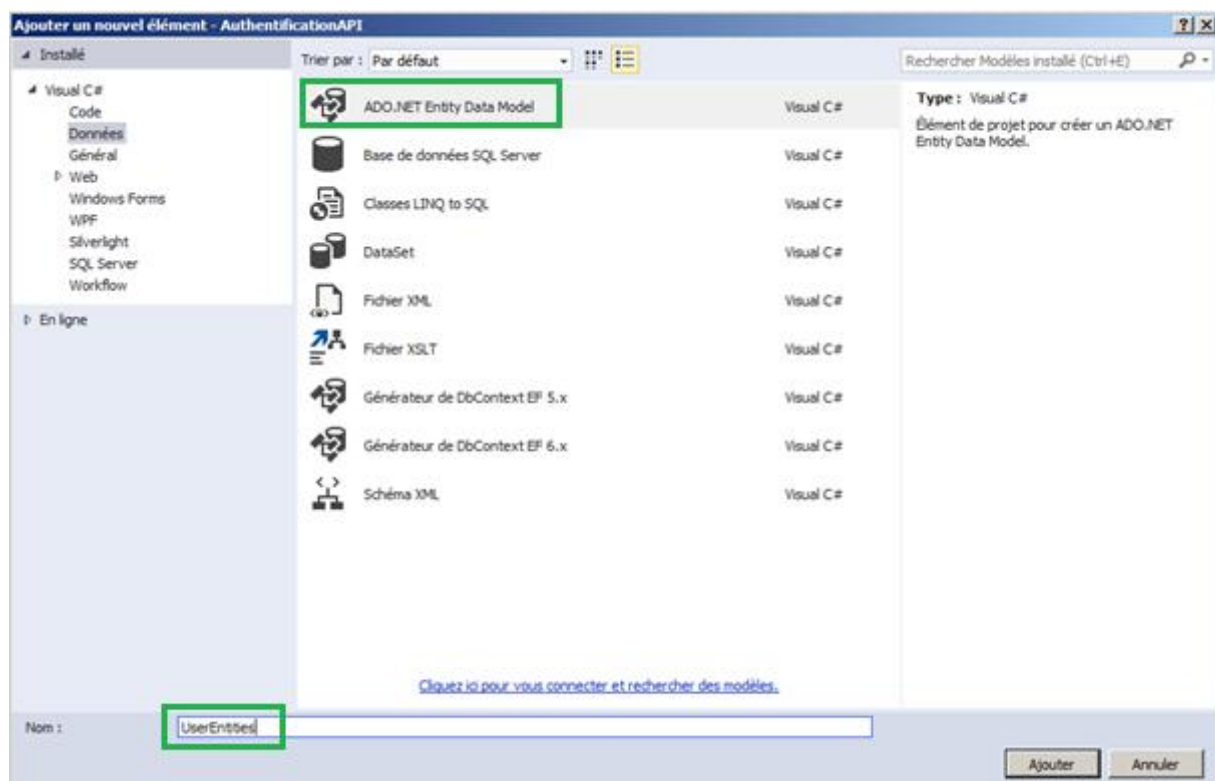
Id	login	passe
1	emi	emi
2	regrag	regrag
3	admin	admin
▶*	NULL	NULL

8-2- Utilisation de l'assistant EDM de Visual studio.NET :

Cliquez droit sur le dossier Models et suivre la procédure en bas :



Puis:



Cliquez sur suivant pour continuer:

Assistant EDM

Choisir le contenu du modèle

Que doit contenir le modèle ?

EF Designer à partir de la base de données

Modèle vide EF Designer

Modèle vide Code First

Code First à partir de la base de données

Crée un modèle dans EF Designer basé sur une base de données existante. Sélectionnez la connexion de base de données, les paramètres du modèle et les objets de base de données à inclure dans le modèle. Les classes avec lesquelles votre application va interagir sont générées à partir du modèle.

< Précédent Suivant > Terminer Annuler

Appuyez sur le bouton Nouvelle connexion :

Assistant EDM

Choisir votre connexion de données

Quelle connexion de données votre application doit-elle utiliser pour établir une connexion à la base de données ?

Nouvelle connexion...

Cette chaîne de connexion semble contenir des données sensibles (par exemple, un mot de passe), lesquelles sont indispensables pour établir une connexion à la base de données. Le stockage des données sensibles dans la chaîne de connexion peut entraîner un risque de sécurité. Voulez-vous indure les données sensibles dans la chaîne de connexion ?

☐ Non, exclure les données sensibles de la chaîne de connexion. Je définirai ces informations dans le code de mon application.

☐ Oui, indure les données sensibles dans la chaîne de connexion.

Chaîne de connexion :

☒ Enregistrer les paramètres de connexion dans Web.Config en tant que :

< Précédent Suivant > Terminer Annuler

Entrez le nom de votre serveur (poste de travail) et sélectionnez la base en question :

Propriétés de connexion

Entrez les informations pour vous connecter à la source de données sélectionnée ou cliquez sur "Modifier" pour sélectionner une autre source de données et/ou un autre fournisseur.

Source de données :
Microsoft SQL Server (SqlClient) [Modifier...]

Nom du serveur :
hp-pc [Actualiser]

Connexion au serveur

☒ Utiliser l'authentification Windows
☐ Utiliser l'authentification SQL Server

Nom d'utilisateur :
Mot de passe :
☐ Enregistrer mon mot de passe

Connexion à la base de données

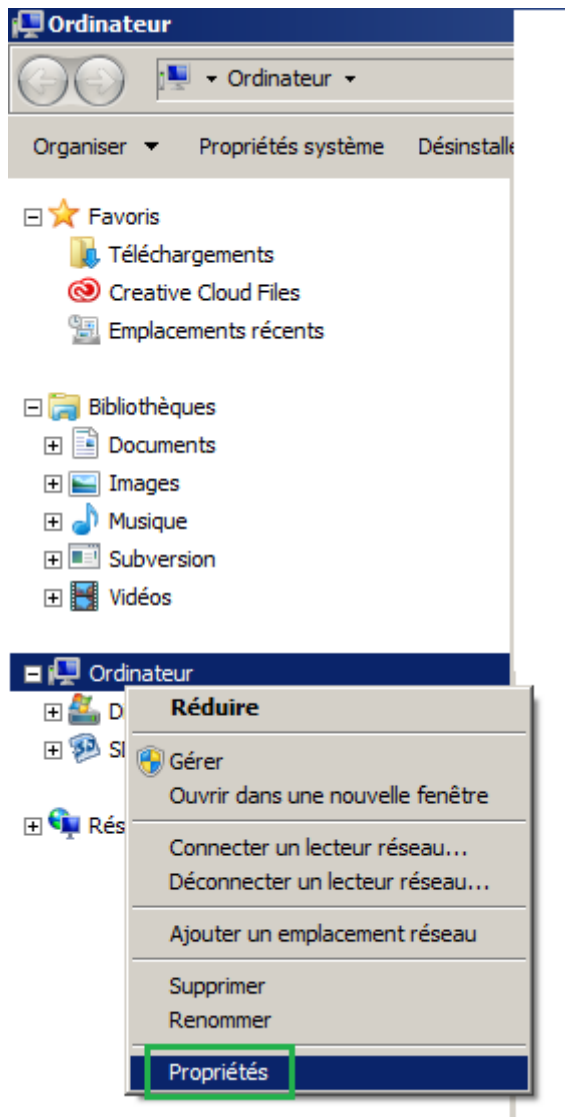
☒ Sélectionner ou entrer un nom de base de données :
Comptes
master
model
msdb
ReportServer
ReportServerTempDB
SDF
tempdb

Avancées...

Tester la connexion [OK] [Annuler]

Appuyez sur le bouton OK pour continuer.

Pour récupérer le nom du serveur cliquez droit sur ordinateur et puis propriété comme indiqué en bas :



Le nom du serveur est celui encadré en vert comme suit :

Informations système générales

Édition Windows

Windows 7 Professionnel

Copyright © 2009 Microsoft Corporation. Tous droits réservés.

Service Pack 1

Obtenir plus de fonctionnalités avec une nouvelle édition de Windows 7



Système

Évaluation :

4,8 Indice de performance Windows

Processeur :

Intel(R) Core(TM) i5-3320M CPU @ 2.60GHz 2.60 GHz

Mémoire installée (RAM) :

4,00 Go (3,87 Go utilisable)

Type du système :

Système d'exploitation 64 bits

Stylet et fonction tactile :

La fonctionnalité de saisie tactile ou avec un stylet n'est pas disponible sur cet écran

Paramètres de nom d'ordinateur, de domaine et de groupe de travail

Nom de l'ordinateur : hp-PC

Nom complet : hp-PC

 [Modifier les paramètres](#)

Description de l'ordinateur :

Groupe de travail : WORKGRC

Appuyez sur suivant :

The screenshot shows the 'Assistant EDM' window with the title 'Choisir votre connexion de données'. It contains a dropdown menu with 'hp-pc.Comptes.dbo' selected, a 'Nouvelle connexion...' button, a warning message about sensitive data, two radio buttons for handling sensitive data, a text box for the connection string, a checked checkbox for saving parameters, and a text box with 'ComptesEntites'. The 'Suivant >' button is highlighted with a green box.

Assistant EDM

Choisir votre connexion de données

Quelle connexion de données votre application doit-elle utiliser pour établir une connexion à la base de données ?

hp-pc.Comptes.dbo Nouvelle connexion...

Cette chaîne de connexion semble contenir des données sensibles (par exemple, un mot de passe), lesquelles sont indispensables pour établir une connexion à la base de données. Le stockage des données sensibles dans la chaîne de connexion peut entraîner un risque de sécurité. Voulez-vous indure les données sensibles dans la chaîne de connexion ?

☐ Non, exdure les données sensibles de la chaîne de connexion. Je définirai ces informations dans le code de mon application.

☐ Oui, indure les données sensibles dans la chaîne de connexion.

Chaîne de connexion :

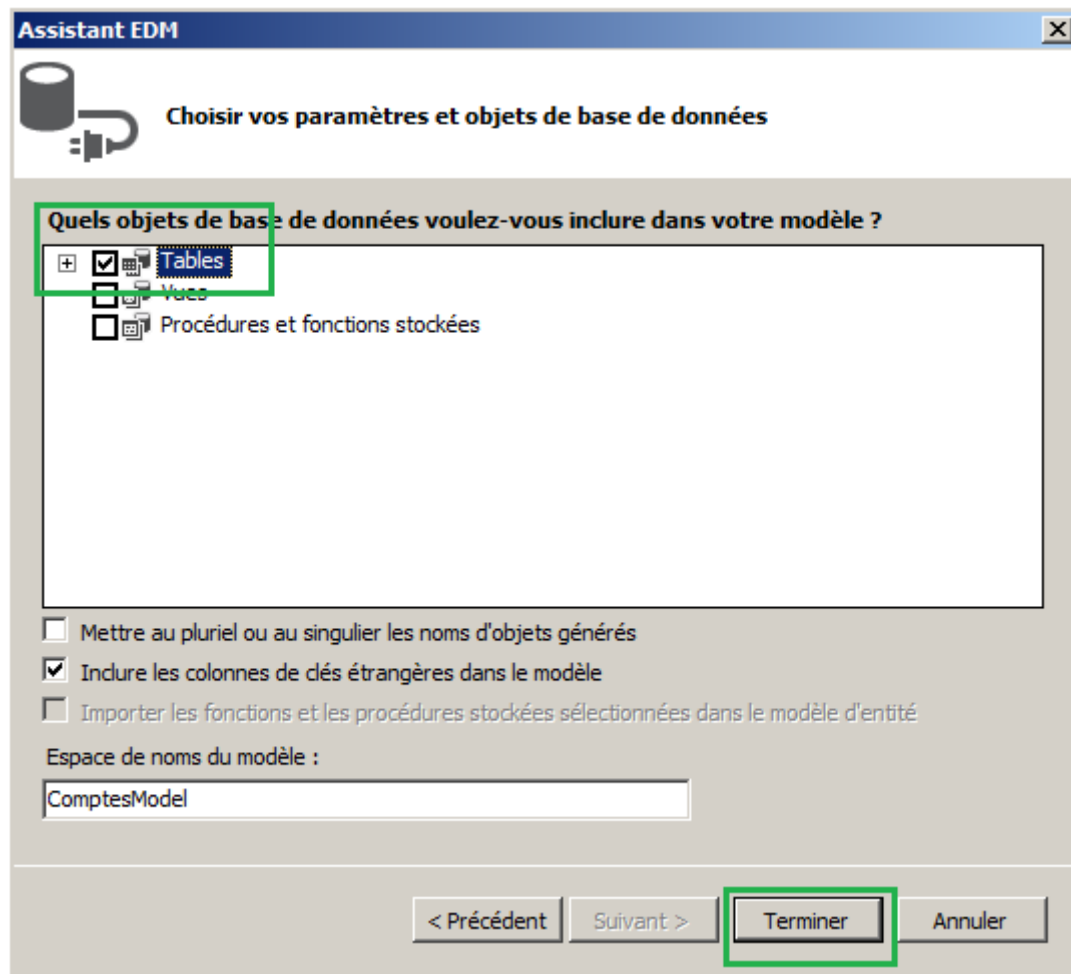
metadata=res://*/Models.UserEntities.csdl|res://*/Models.UserEntities.ssdl|
res://*/Models.UserEntities.msl;provider=System.Data.SqlClient;provider connection
string="data source=hp-pc;initial catalog=Comptes;integrated

☒ Enregistrer les paramètres de connexion dans Web.Config en tant que :

ComptesEntites

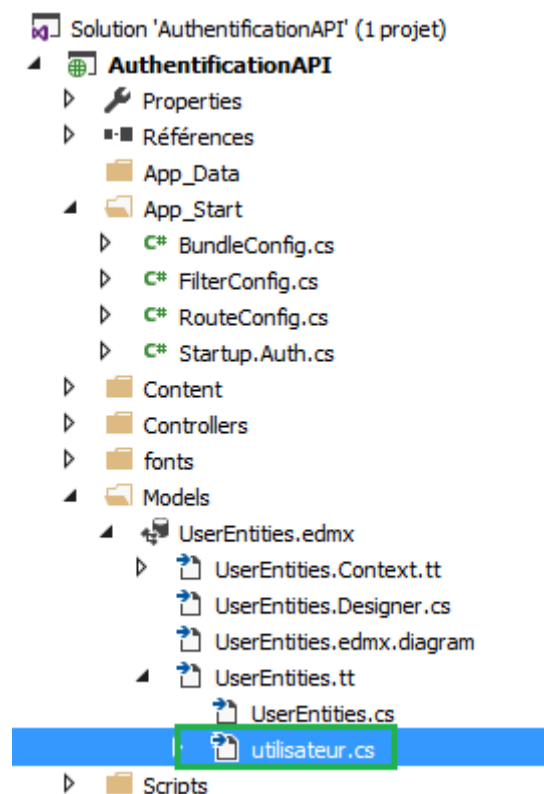
< Précédent **Suivant >** Terminer Annuler

Cochez Tables comme en bas :



Et enfin appuyez sur le bouton Terminer pour conclure.

Enfin vous devez compléter le fichier utilisateur.cs indiqué en bas :



En ajoutant la fonction GetInstance comme suit :

```

internal static bool GetInstance(string aReference, string aMotDePasse)
{
    using (AuthenticationAPI.Models.ComptesEntities context = new
AuthenticationAPI.Models.ComptesEntities())
    {
        List<Models.utilisateur> listeServiceser = new
List<Models.utilisateur>();
        //listeServiceser = requete.ToList();
        //string LastName = "emi";
        string LastName = aReference;
        //Boolean trv = true;
        object o = null;
        var contactQuery = from contact in context.utilisateur
                           where contact.login == LastName && contact.passe ==
aMotDePasse
                           select contact;
        listeServiceser = contactQuery.ToList();

        return (listeServiceser.Count != 0);
        //return false;
    }
    throw new NotImplementedException();
}

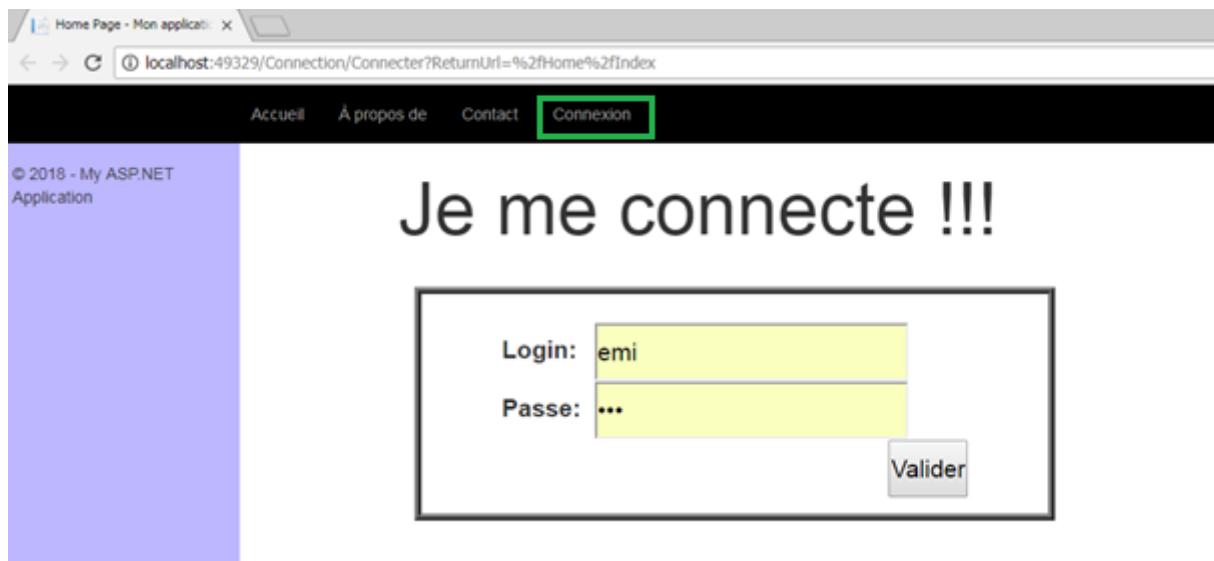
```

N'oubliez la directive:

```
using System.Linq;
```

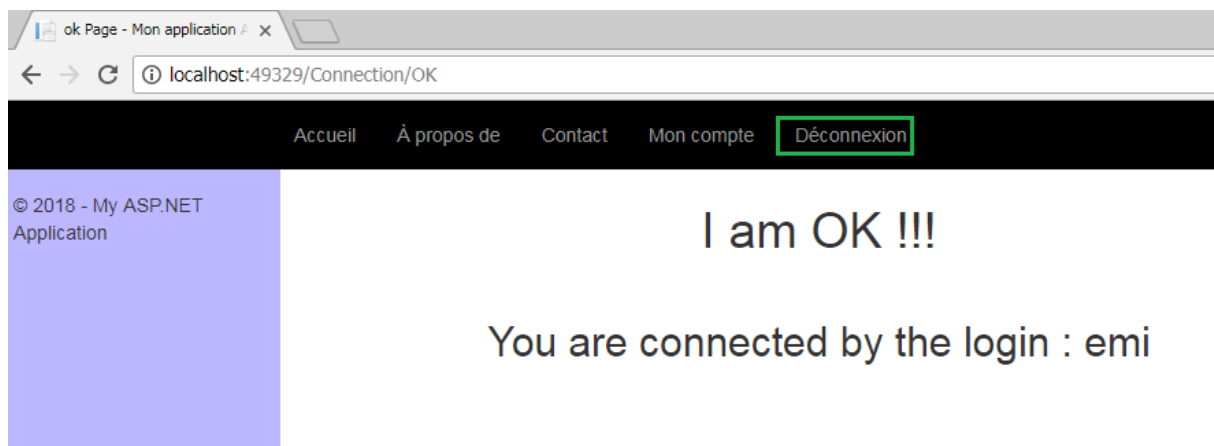
III-Démonstration :

Lancez l'application on obtient ce qui suit :



Pour se connecter appuyez sur le lien connexion.

Saisissez votre login et mot de passe et appuyez sur le bouton Valider.



Pour se déconnecter appuyer sur le lien Déconnexion.

Pour obtenir l'écran en bas :

