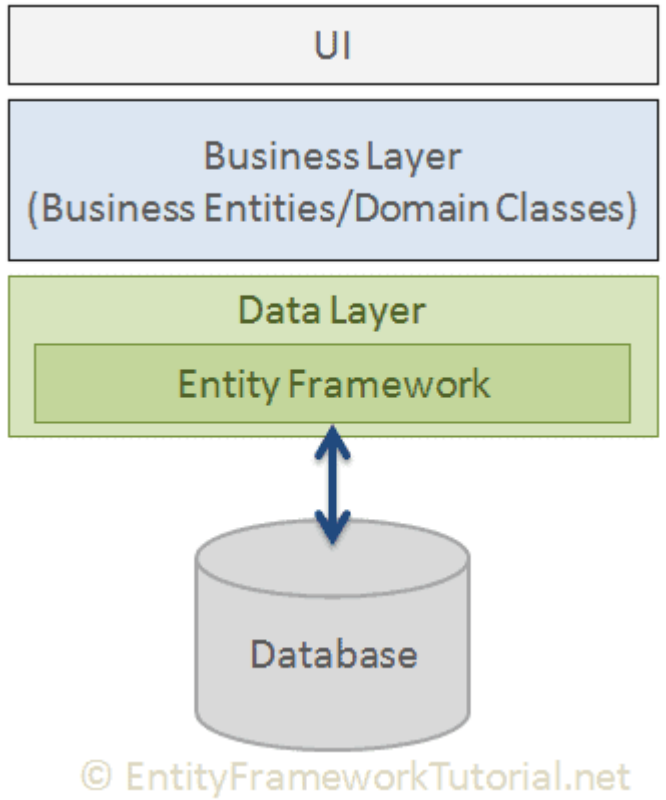
L’approche Code First d’Entity Framework pour l’accès au base de données

MOHAMED AIROUCHE

L'ORM Entity Framework

Rappels

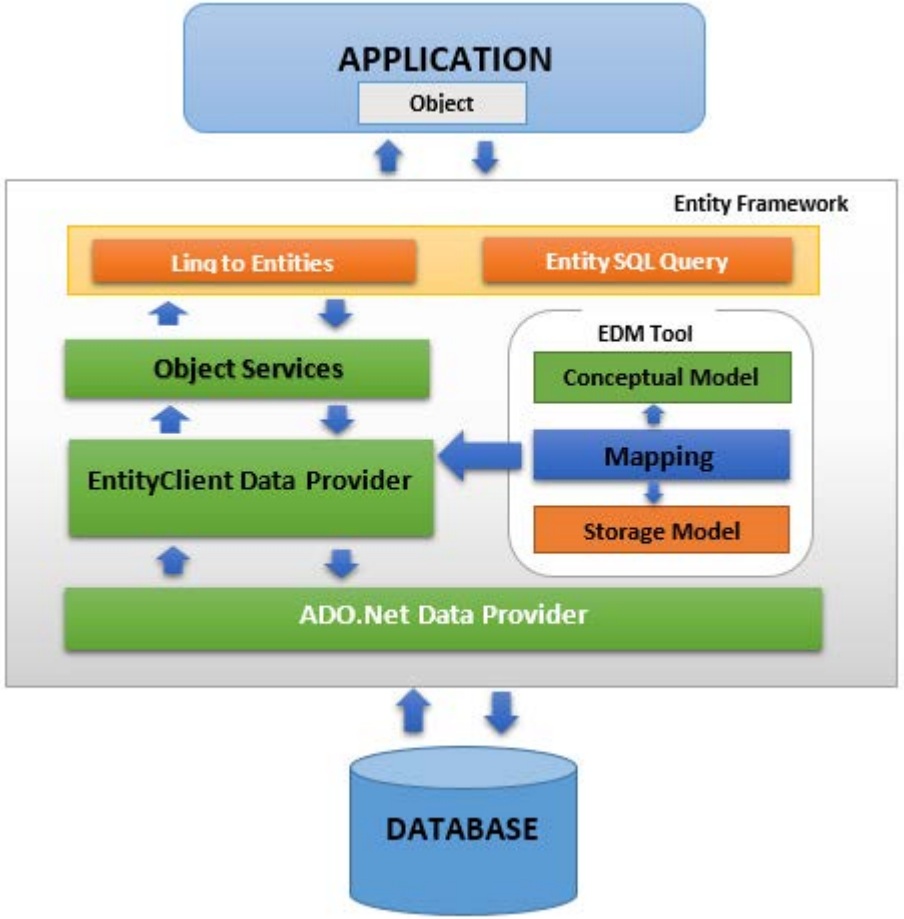
- ORM veut dire en anglais object-relational mapping.
- Il s'agit d'une technique de programmation informatique qui permet d'établir des correspondances entre une base de données relationnelle et un modèle objet de classes.
- Il existe plusieurs Frameworks qui permettent de faire le mapping objet relationnel. On peut citer :
  - NHibernate : Une version de Hibernate pour .NET
  - Entity Framework : Intégré à .NET.



https://www.entityframeworktutorial.net/what-is-entityframework.aspx

L'ORM Entity Framework

- ✓ Entity Framework: est une brique logicielle permettant de faciliter l'accès à une base de données.
- ✓ Ce Framework est un ORM(Object-Rational Mapping), c'est-à-dire un outil qui permet de travailler des objets C# plutôt qu'avec des requêtes en dur.
- ✓ Le développeur va ainsi gérer, ajouter ou supprimer des objets via l'ORM.
- ✓ L'ORM s'occuper de traduire automatiquement les opérations via des requêtes SQL.



Architecture d'Entity Framework

Entity Framework est un outil open source, ORM (Object-Relational Mapper) pour accéder à la base de données dans une approche orientée objet utilisant la technologie .NET.

https://www.c-sharpcorner.com/article/get-your-hands-on-entity-framework-code-first-workflow/

L'ORM Entity Framework

Entity Framework utilise 3 types d'approches : https://www.entityframeworktutorial.net/choosing-development-approach-with-entity-framework.aspx

Database first (La base de données d'abord)

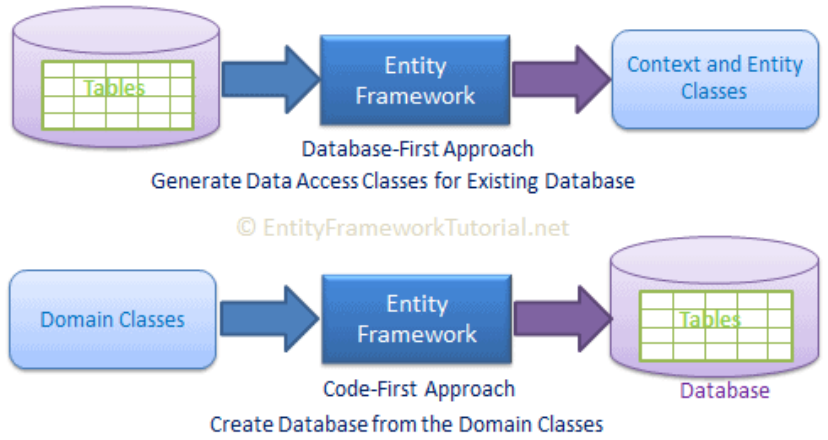
Dans l'approche Database first, nous concevons nos tables à l'aide d'un concepteur visuel ou à l'aide d'une requête SQL. Ensuite, Entity Framework génère une classe de domaine. C'est l'approche traditionnelle que de nombreux développeurs suivent depuis la version initiale d'EF 1 et de Visual Studio 2008.

Code first (Code d'abord)

Dans l'approche Code first, nous créons notre classe de domaine, puis l'EF génère une base de données pour nous. Le Code First a été publié avec EF 4.1.1 et Visual Studio 2010.

Model first (Modèle d'abord)

Dans l'approche Model first, nous utilisons le concepteur visuel de Visual Studio pour modéliser notre classe et ses associés. C'est comme un diagramme UML et sur cette base, EF génère une classe de domaine et une base de données. Il a été publié avec EF 4.0 et Visual Studio 2010.



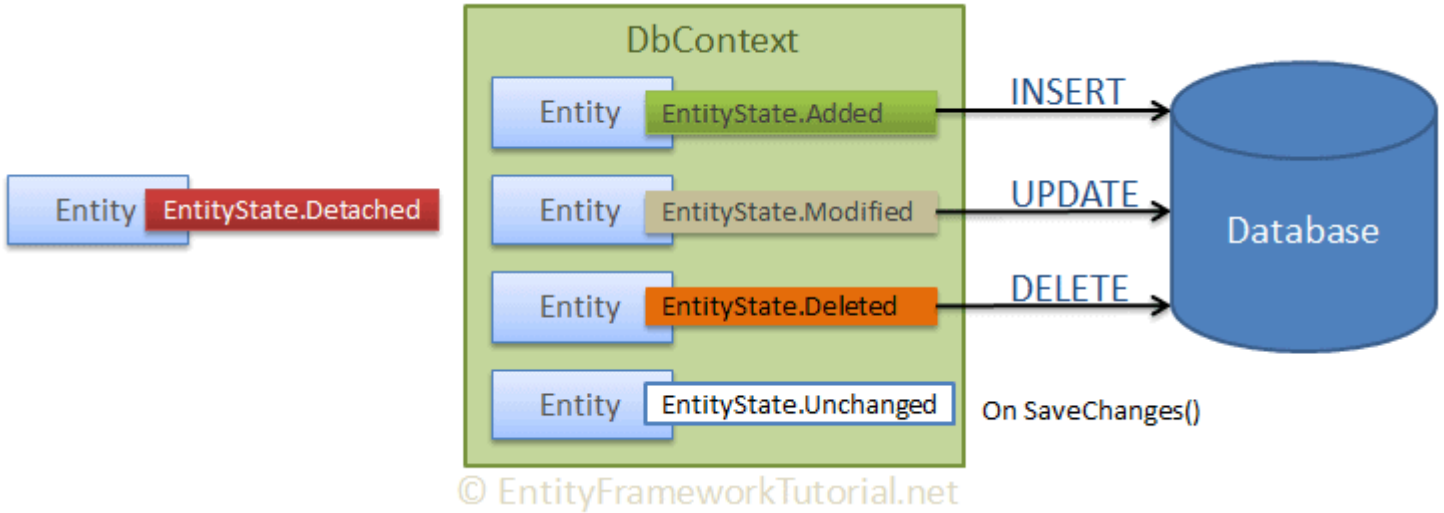
https://www.c-sharpcorner.com/article/get-your-hands-on-entity-framework-code-first-workflow/

https://stackoverflow.com/questions/37975075/entity-framework-clarification-about-model-objects-and-their-methods

L'ORM Entity Framework

Entity Framework se base sur les éléments suivants :

- ✓ La classe de contexte de la base de données. Ce contexte est une classe dérivée de la classe système [System.Data.Entity.DbContext]. Elle sert à définir les images objets des tables de la base de données. Elle est utilisée pour interroger ou enregistrer des données dans la base de données en effectuant les opérations CRUD (créer, lire, mettre à jour, supprimer).
- ✓ Les classes des entités Entity Framework, qui sont des classes dans lesquelles on encapsule les lignes des différentes tables de la base de données.



https://www.entityframeworktutorial.net/basics/entity-states.aspx

Exemple

Développement d'une application qui permet de gérer des employés et des départements :

On souhaite créer une application qui permet de gérer des employés et les départements de ces employés. Un département est défini par son identifiant et son nom. Chaque employé est défini par son identifiant, son nom, son prénom, son sexe, son identifiant de département et sa ville.

L'application doit permettre de :

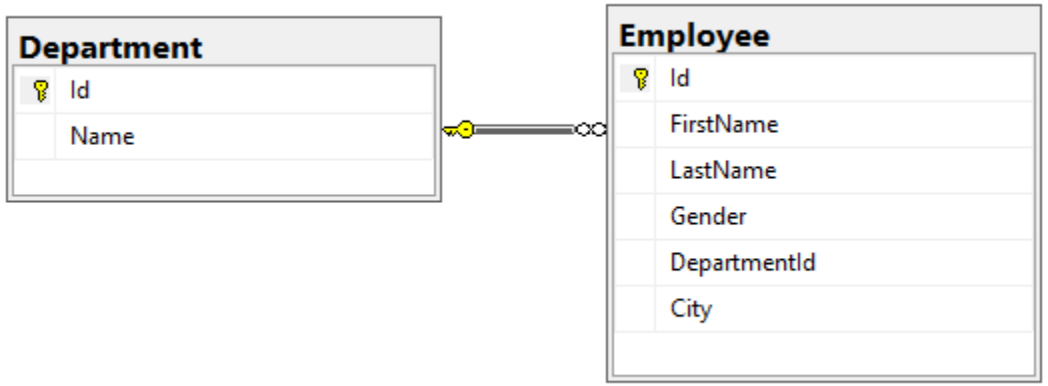
- Pour les employés :
  - Afficher tous les employés.
  - Afficher les détails d'un employé.
  - Saisir et ajouter un nouvel employé.
  - Éditer et modifier un employé.
  - Supprimer un employé.

- Pour les départements :
  - Afficher tous les départements.
  - Afficher les détails d'un département.
  - Saisir et ajouter un nouveau département.
  - Éditer et modifier un département.
  - Supprimer un département.

Exemple

Développement d'une application qui permet de gérer des employés et des départements :

Les employés et les départements seront stockés dans une base de données locale SQL Server Express. Cette base de données doit contenir deux tables (table Employee et table Department).



Dans cet exemple, l'application utilisera l'approche Code First de l'ORM Entity Framework pour l'accès aux données.

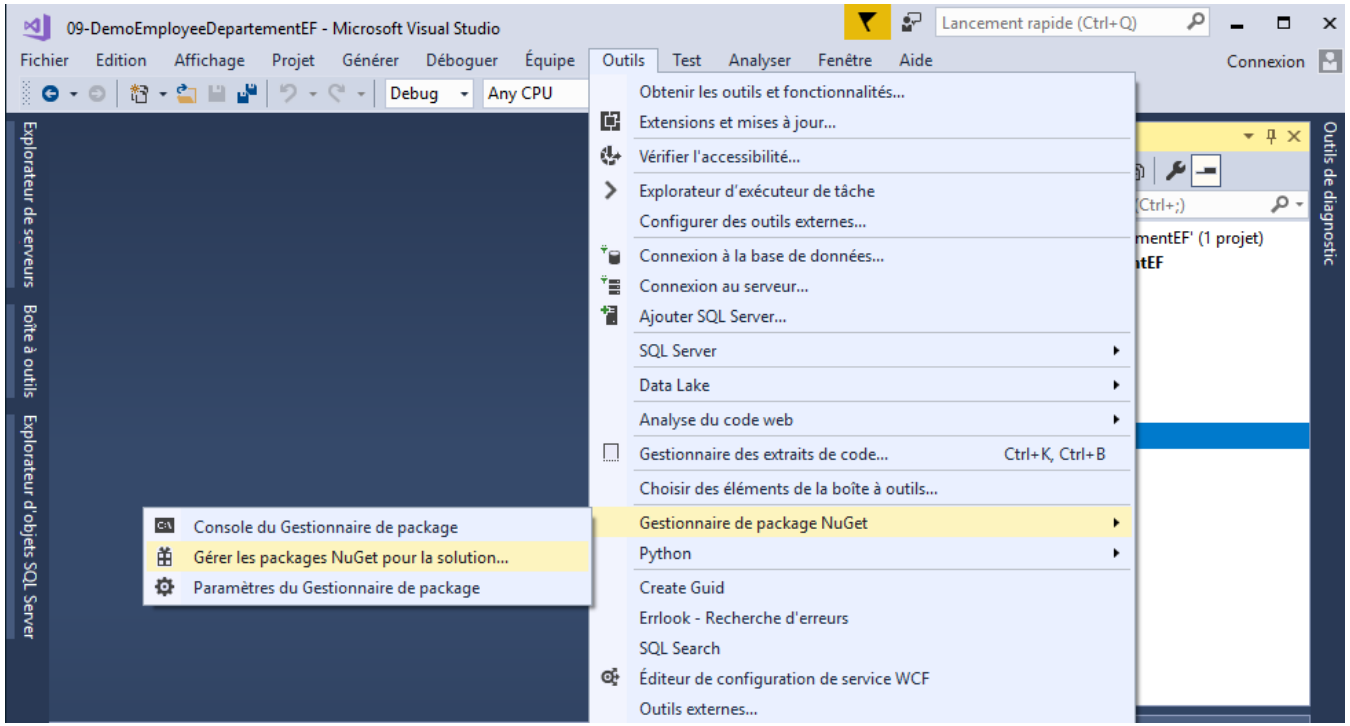
Exemple

Création d'une application ASP.NET MVC :

- Créez une application ASP.NET MVC : **09-DemoEmployeeDepartementEF**.
- Dans cet exemple nous utiliserons l'approche Code First de l'ORM Entity Framework.

Ajout d'Entity Framework au projet :

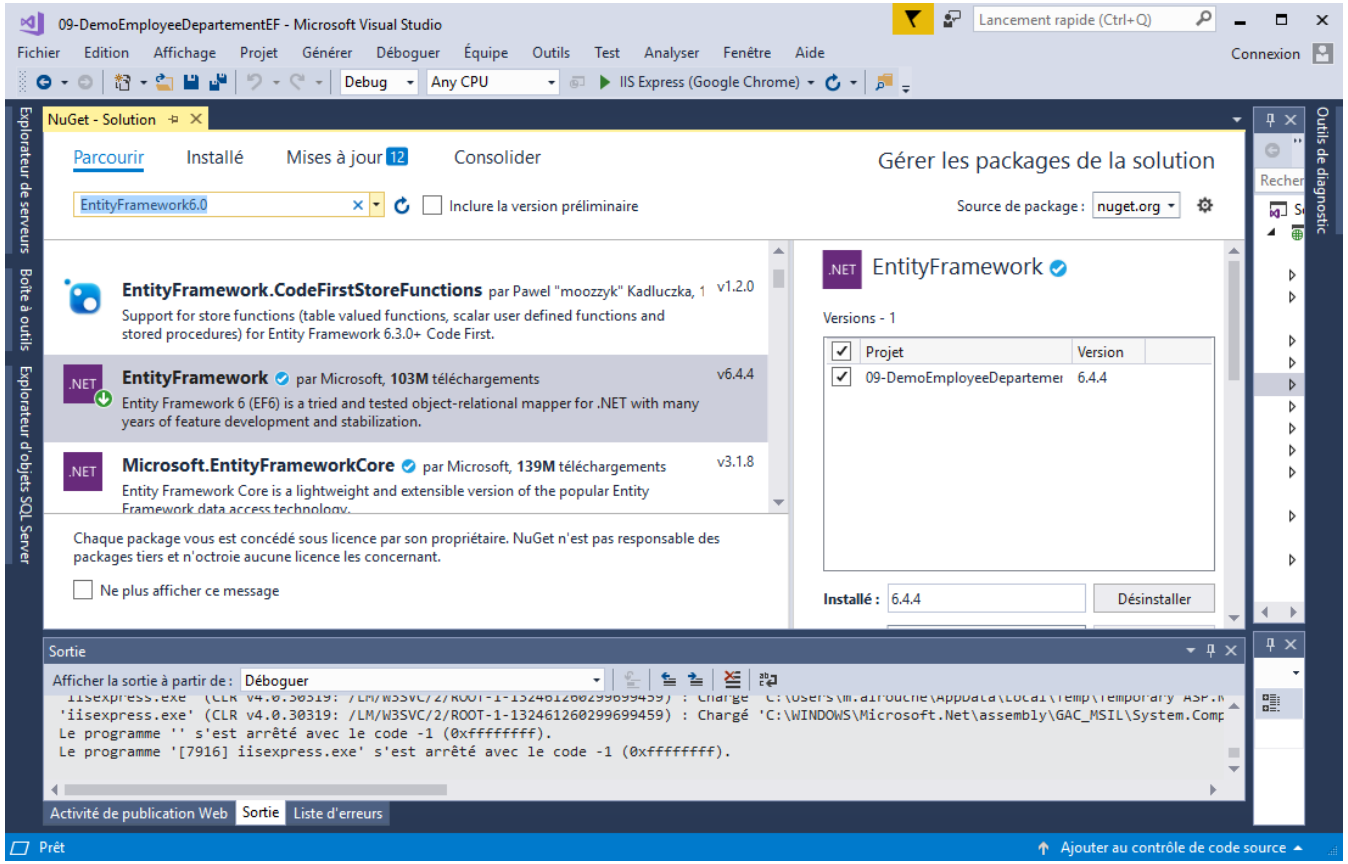
✓ Pour ajouter une dépendance à un projet, utilisez l'outil NuGet.



Exemple

Ajout d'Entity Framework au projet :

- ✓ Pour ajouter Entity Framework au projet, utilisez l'outil NuGet.
- ✓ Chercher Entity Framework, puis cocher la case à coucher pour l'ajouter au projet.



Exemple

Ajout de modèle à utiliser par Entity Framework :

- ✓ Pour utiliser l'approche Code First d'ORM Entity Framework, il faut ajouter les classes des entités et la classe de contexte qui hérite de la classe DbContext.

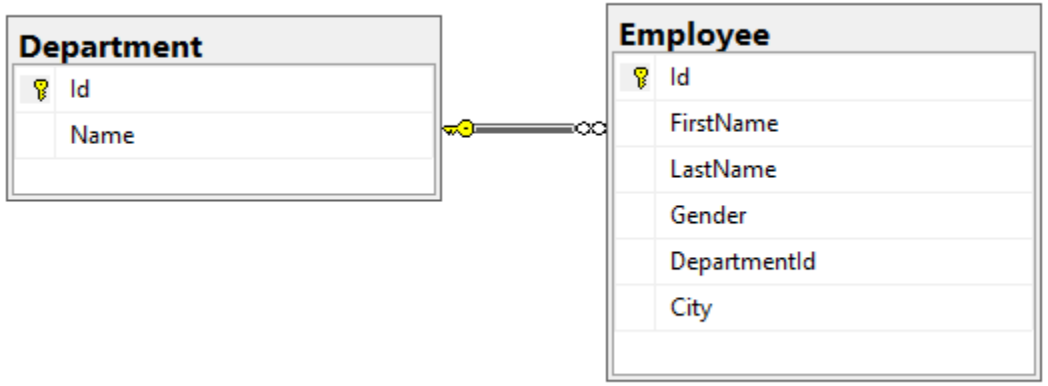
- ✓ La classe de contexte de la base de données : EmployeeEntitiesDBContext

```
using System.Data.Entity;

namespace _09_DemoEmployeeDepartementEF.Models
{
   public class EmployeeEntitiesDBContext: DbContext
   {
      public EmployeeEntitiesDBContext()
          : base("name=EmployeeEntitiesContextDB")
      {
      }

      public DbSet<Employee> Employees { get; set; }

      public DbSet<Department> Departments { get; set; }
   }
}
```

Exemple

L'association : One To Many ----- ManyToOne :

La base de données doit contenir deux tables reliées (table Employee et table Department).



- ✓ Prenons le cas où chaque employé appartient à un département et un département contient plusieurs employés.
- ✓ Dans ce cas, nous avons deux entités : Employee et Department.
- ✓ La relation qui lie ces deux entités est de type OneToMany ←→ ManyToOne

Exemple

✓ La classe d'entité : Employee

```
using System.ComponentModel.DataAnnotations;

namespace _09_DemoEmployeeDepartementEF.Models
{
    public class Employee
    {
        public int Id { get; set; }
        [Required]
        public string FirstName { get; set; }
        [Required]
        public string LastName { get; set; }
        [Required]
        public string Gender { get; set; }
        [Required]
        public int DepartmentId { get; set; }
        [Required]
        public string City { get; set; }

        public virtual Department Department { get; set; }
    }
}
```

✓ La classe d'entité : Department

```
using System.Collections.Generic;
using System.ComponentModel.DataAnnotations;

namespace _09_DemoEmployeeDepartementEF.Models
{
    public class Department
    {
        public int Id { get; set; }
        [Required]
        public string Name { get; set; }

        public virtual ICollection<Employee> Employees { get; set; }
    }
}
```

Propriété de clé

Propriétés de navigation

Exemple

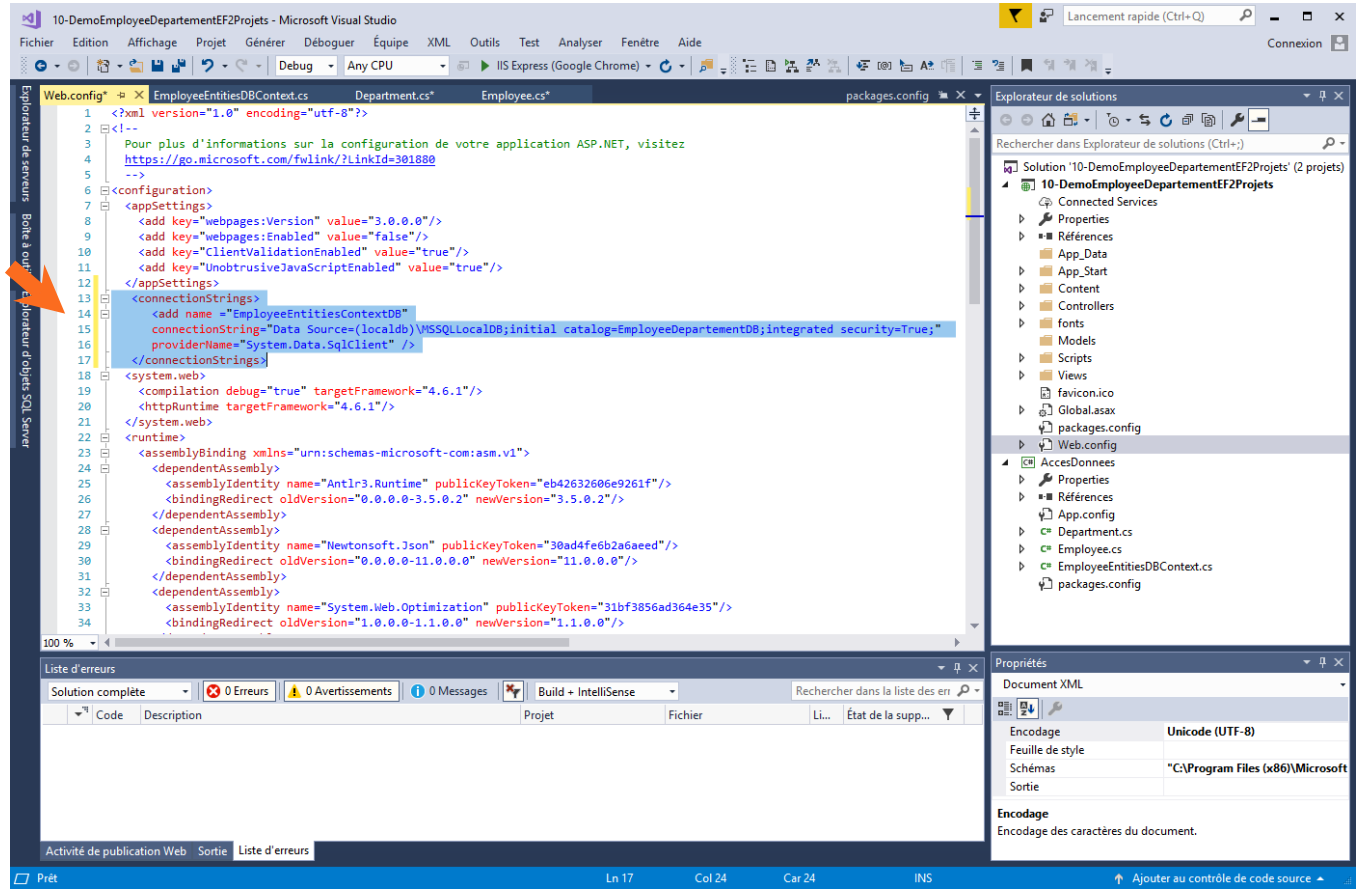
Lazy Loading

- Dans le monde des ORMs, une technique appelée "le lazy loading" est utilisée pour éviter de charger les entités liées en mémoire sans en avoir besoin. Cela peut être intéressant de faire si vous devez précharger en une seule requête un nombre important de données.
- Dans notre exemple, quand on demande à Entity Framework de charger un Objet employé, il ne va pas charger le Département lié à cet employé.
- Pour activer le lazy loading, il faut ajouter le mot clé **Virtual** devant les propriétés de navigation.
- Pour charger les entités liées, on peut utiliser la méthode **Include** dans les requêtes LINQ.
- L'exemple suivant permet de retourner la liste des employés et pour chaque employé, il charge avec son département: **return employees = db.Employees.Include(e => e.Department)**
- Si la méthode Include n'est pas utilisée, il ne va charger que la liste des employés.

Exemple

Ajout de la chaine de connexion nommée «EmployeeEntitiesContextDB» dans le fichier Web.config.

```
<add name ="EmployeeEntitiesContextDB"
      connectionString="Data
Source=(localdb)\MSSQLLocalDB;initial
catalog=EmployeeDepartementDB;integrated
security=True;"

providerName="System.Data.SqlClient" />
```



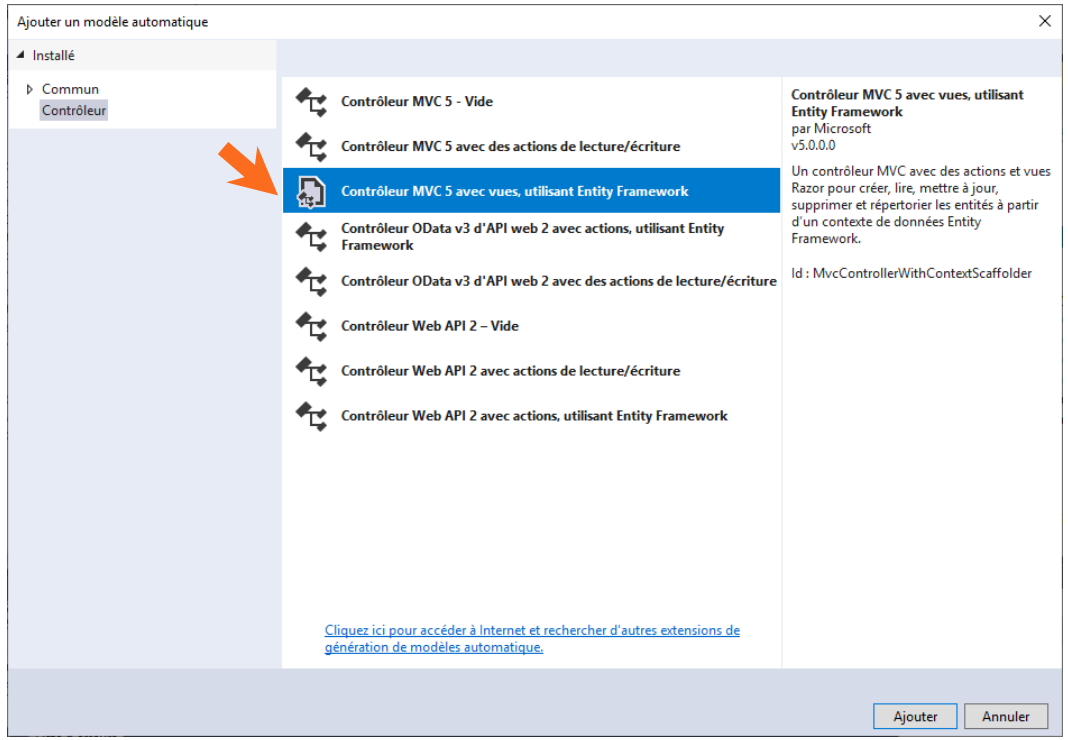
Générer la solution.

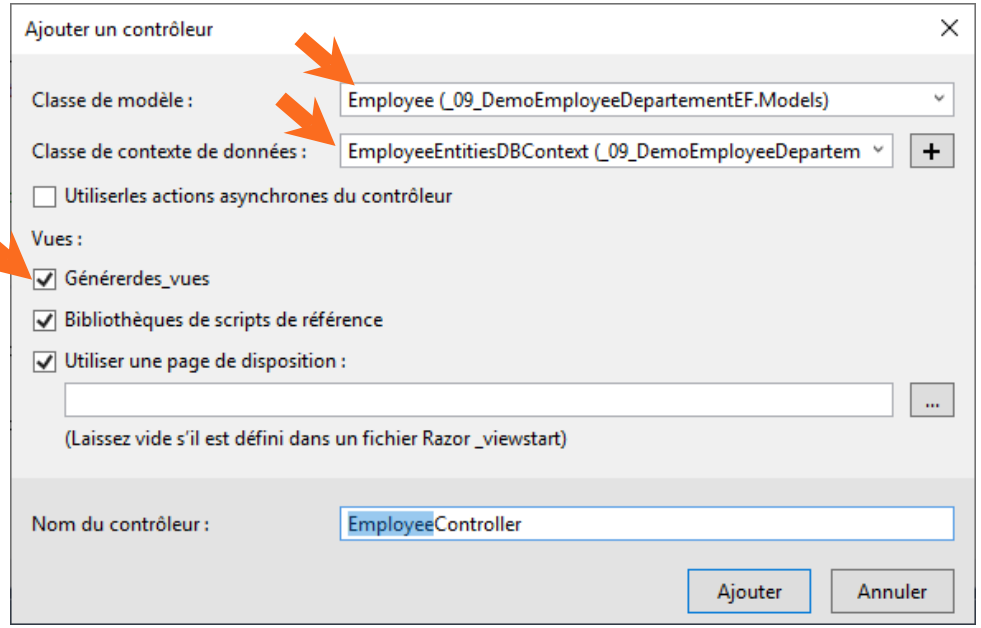
Exemple

Ajout d'un contrôleur Employee à l'application :

Faire un clic droit sur le dossier «Controllers» du projet >> Ajouter >> Contrôleur.

Sélectionner le modèle Contrôleur MVC 5 avec vues, utilisant Entity Framework.





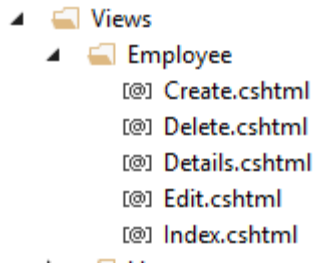
Exemple

Visual studio vas créer automatiquement dans le contrôleur **Employee** les méthodes d'actions suivantes :

- Index
- Create (GET, POST)
- Edit (GET, POST)
- Delete (GET, POST)
- Details

De plus, pour chaque méthode d'action de contrôleur Visual studio ajoute une vue. Au final, on doit trouver dans le dossier **Views/Employee** les vues suivantes :

- Index.cshtml
- Create.cshtml
- Edit.cshtml
- Delete.cshtml
- Details.cshtml

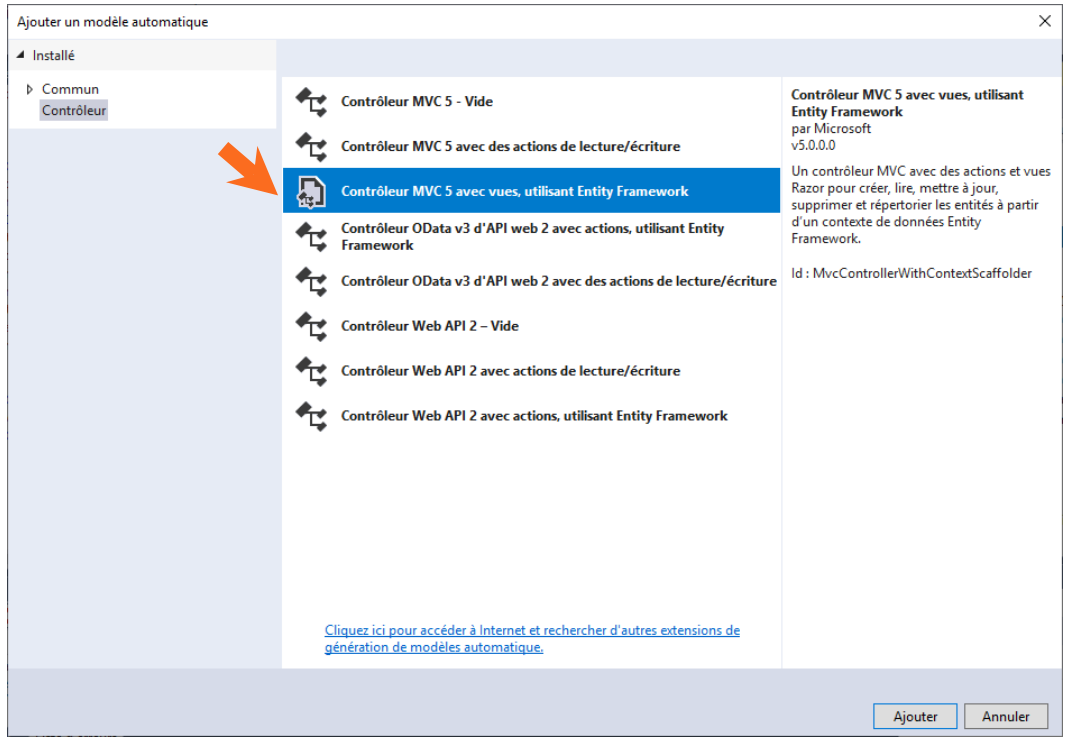


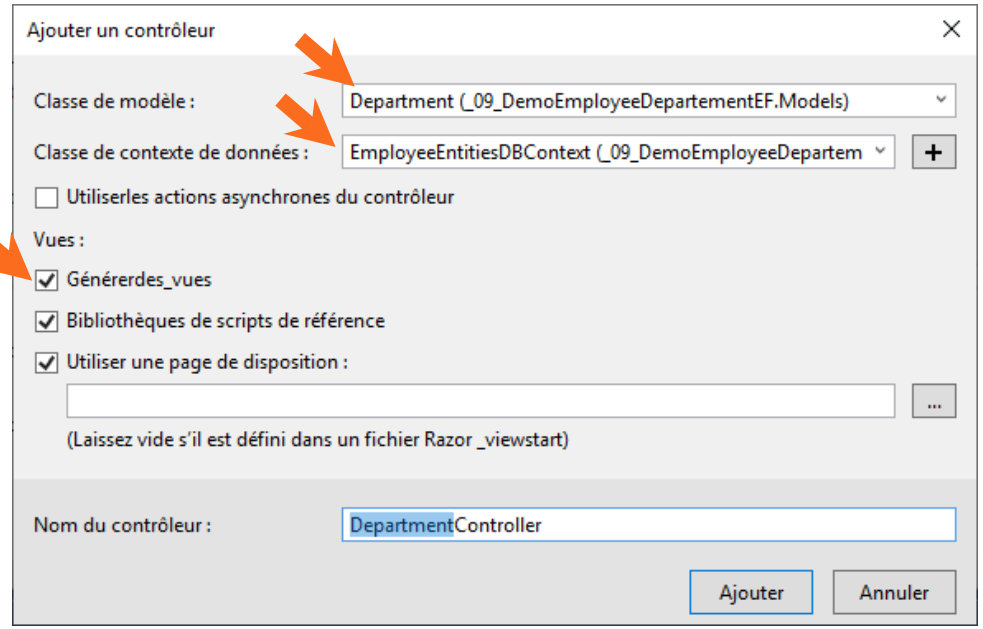
Exemple

Ajout d'un contrôleur Department à l'application :

Faire un clic droit sur le dossier «Controllers» du projet >> Ajouter >> Contrôleur.

Sélectionner le modèle Contrôleur MVC 5 avec vues, utilisant Entity Framework.





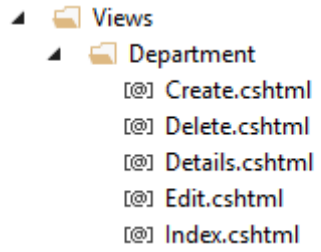
Exemple

Visual studio vas créer automatiquement dans le contrôleur **Department** les méthodes d'actions suivantes :

- Index
- Create (GET, POST)
- Edit (GET, POST)
- Delete (GET, POST)
- Details

De plus, pour chaque méthode d'action de contrôleur Visual studio ajoute une vue. Au final, on doit trouver dans le dossier **Views/Department** les vues suivantes :

- Index.cshtml
- Create.cshtml
- Edit.cshtml
- Delete.cshtml
- Details.cshtml



Exemple – Solution avec deux projets

Création d'une application ASP.NET MVC :

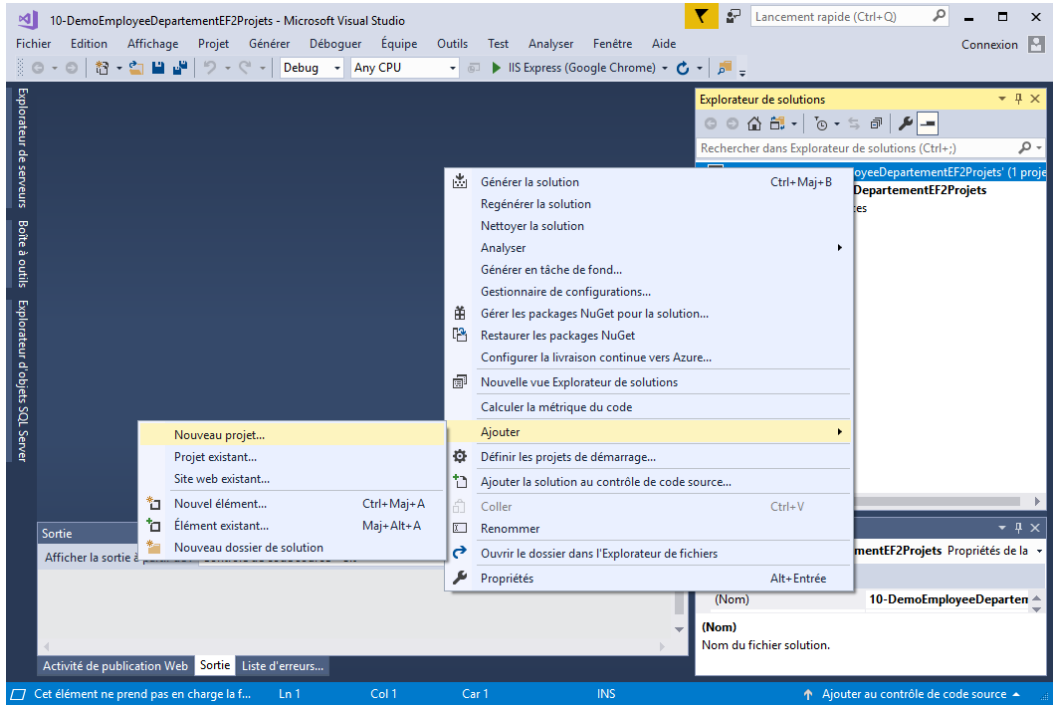
- Créez une application ASP.NET MVC : **10-DemoEmployeeDepartementEF2Projets**.
- Dans cet exemple nous utiliserons l'approche Code First de l'ORM Entity Framework avec la logique d'accès aux données dans un projet distinct.

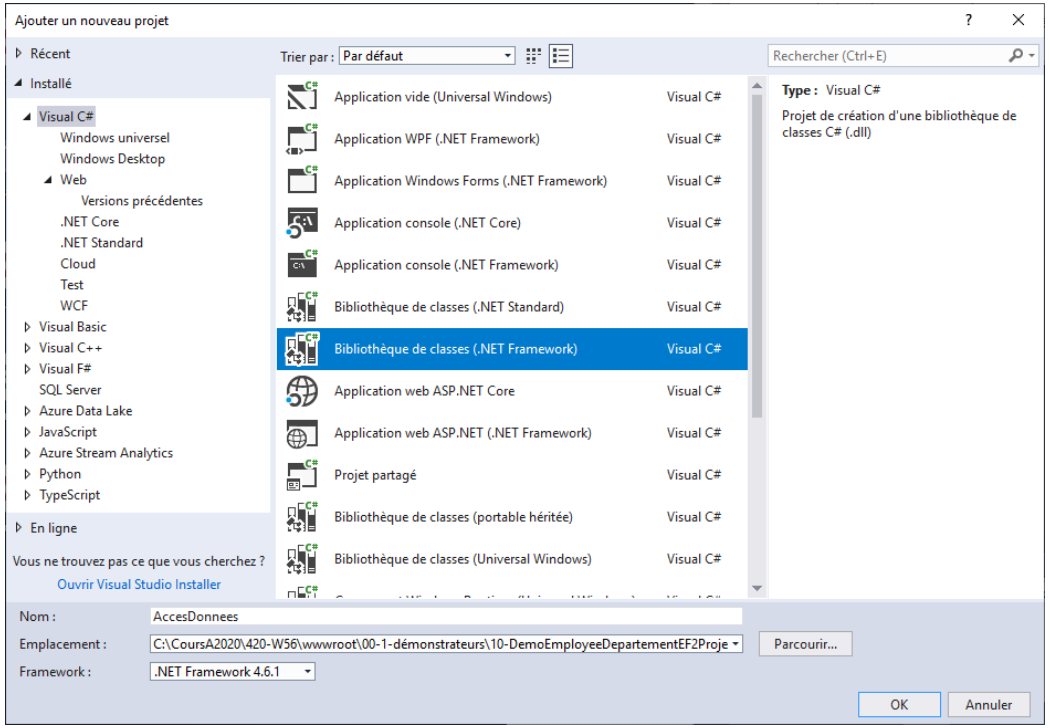
Ajout d'un projet Bibliothèque de classes à la solution :

1. Nous aurons le modèle d'entité dans un projet distinct.
2. Cliquez avec le bouton droit sur la solution **10-DemoEmployeeDepartementEF2Projets** dans l'Explorateur de solutions et sélectionnez Ajouter - Nouveau projet
3. Sélectionnez Projet de bibliothèque de classes (.NET Framework). Nommez le projet **AccesDonnees** et cliquez sur OK.

Exemple – Solution avec deux projets

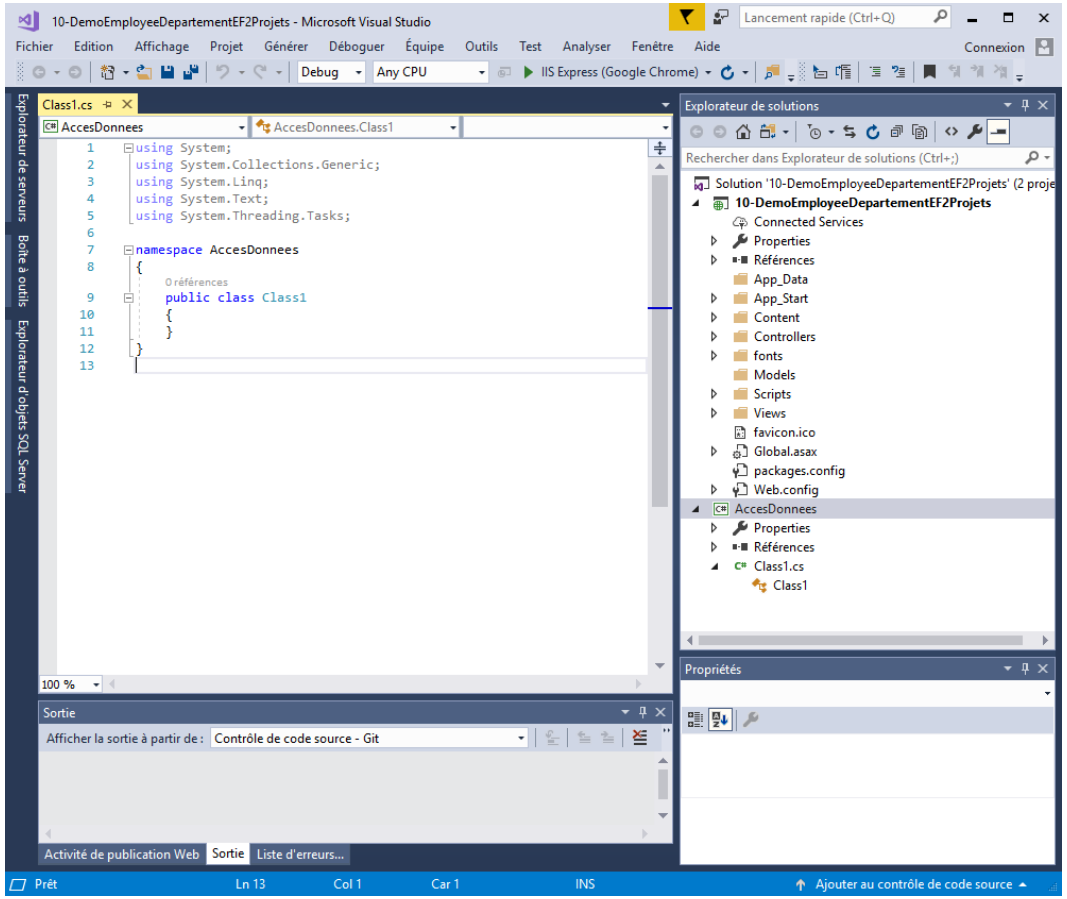
Ajout d'un projet Bibliothèque de classes à la solution :





Exemple – Solution avec deux projets

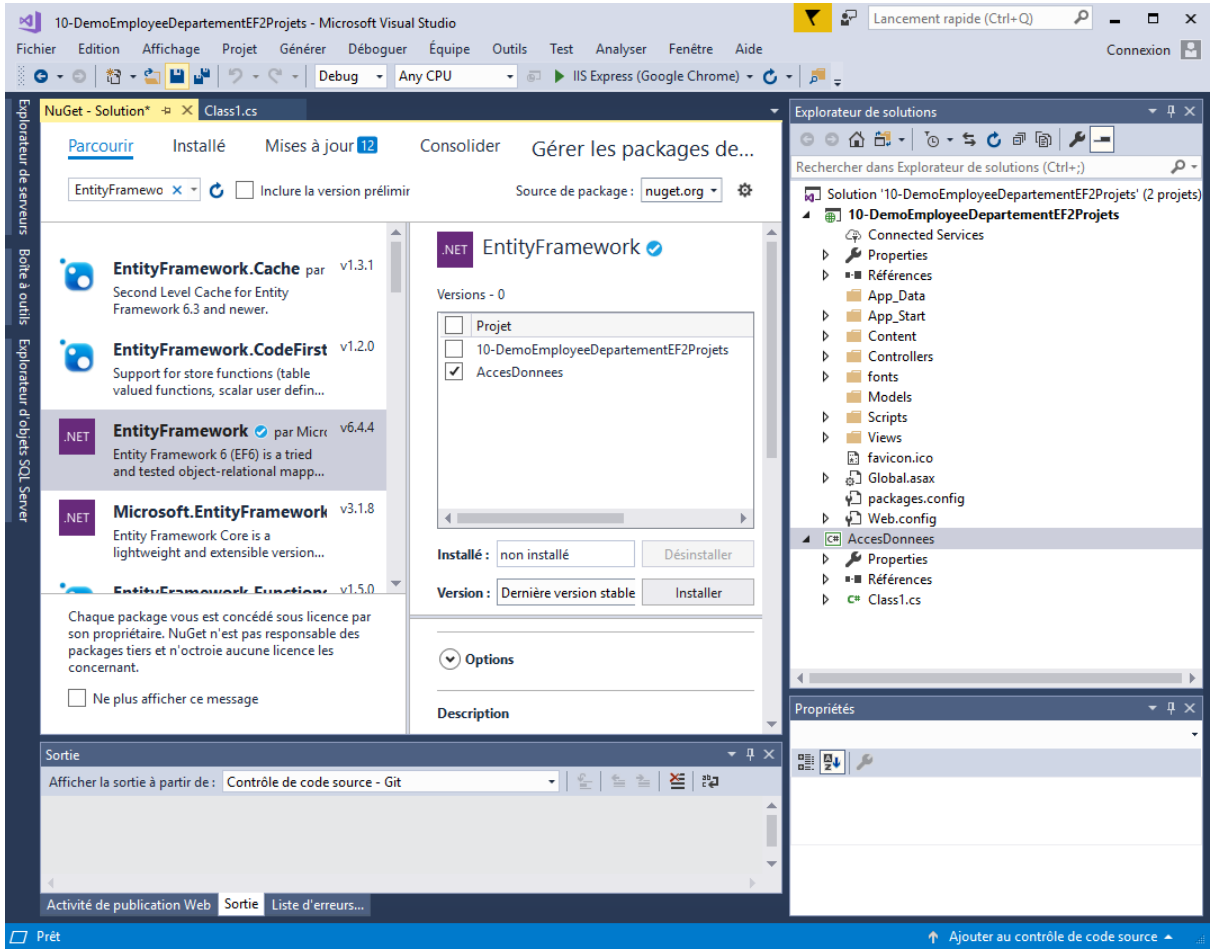
Solution contenant 2 Projets :



Exemple – Solution avec deux projets

Ajouter Entity Framework au projet AccesDonnees :

- ✓ Pour ajouter Entity Framework au projet **AccesDonnees,** utilisez l'outil NuGet
- ✓ Chercher Entity Framework, puis cocher la case à coucher pour l'ajouter au projet.



Exemple – Solution avec deux projets

Ajout de modèle à utiliser par Entity Framework au projet AccesDonnees :

- ✓ Pour utiliser l'approche Code First d'ORM Entity Framework, il faut ajouter au projet **AccesDonnees** les classes des entités et la classe de contexte qui hérite de la classe DbContext.

- ✓ La classe de contexte de la base de données générée : EmployeeEntitiesDBContext

```
using System.Data.Entity;

namespace AccesDonnees
{
   public class EmployeeEntitiesDBContext: DbContext
   {
      public EmployeeEntitiesDBContext()
          : base("name=EmployeeEntitiesContextDB")
      {
      }

      public DbSet<Employee> Employees { get; set; }

      public DbSet<Department> Departments { get; set; }
   }
}
```

Exemple – Solution avec deux projets

✓ La classe d'entité : Employee

```
using System.ComponentModel.DataAnnotations;

namespace AccesDonnees
{
    public class Employee
    {
        public int Id { get; set; }
        [Required]
        public string FirstName { get; set; }
        [Required]
        public string LastName { get; set; }
        [Required]
        public string Gender { get; set; }
        [Required]
        public int DepartmentId { get; set; }
        [Required]
        public string City { get; set; }

        public virtual Department Department { get; set; }
    }
}
```

✓ La classe d'entité : Department

```
using System.Collections.Generic;
using System.ComponentModel.DataAnnotations;

namespace AccesDonnees
{
    public class Department
    {
        public int Id { get; set; }
        [Required]
        public string Name { get; set; }

        public virtual ICollection<Employee> Employees { get; set; }
    }
}
```

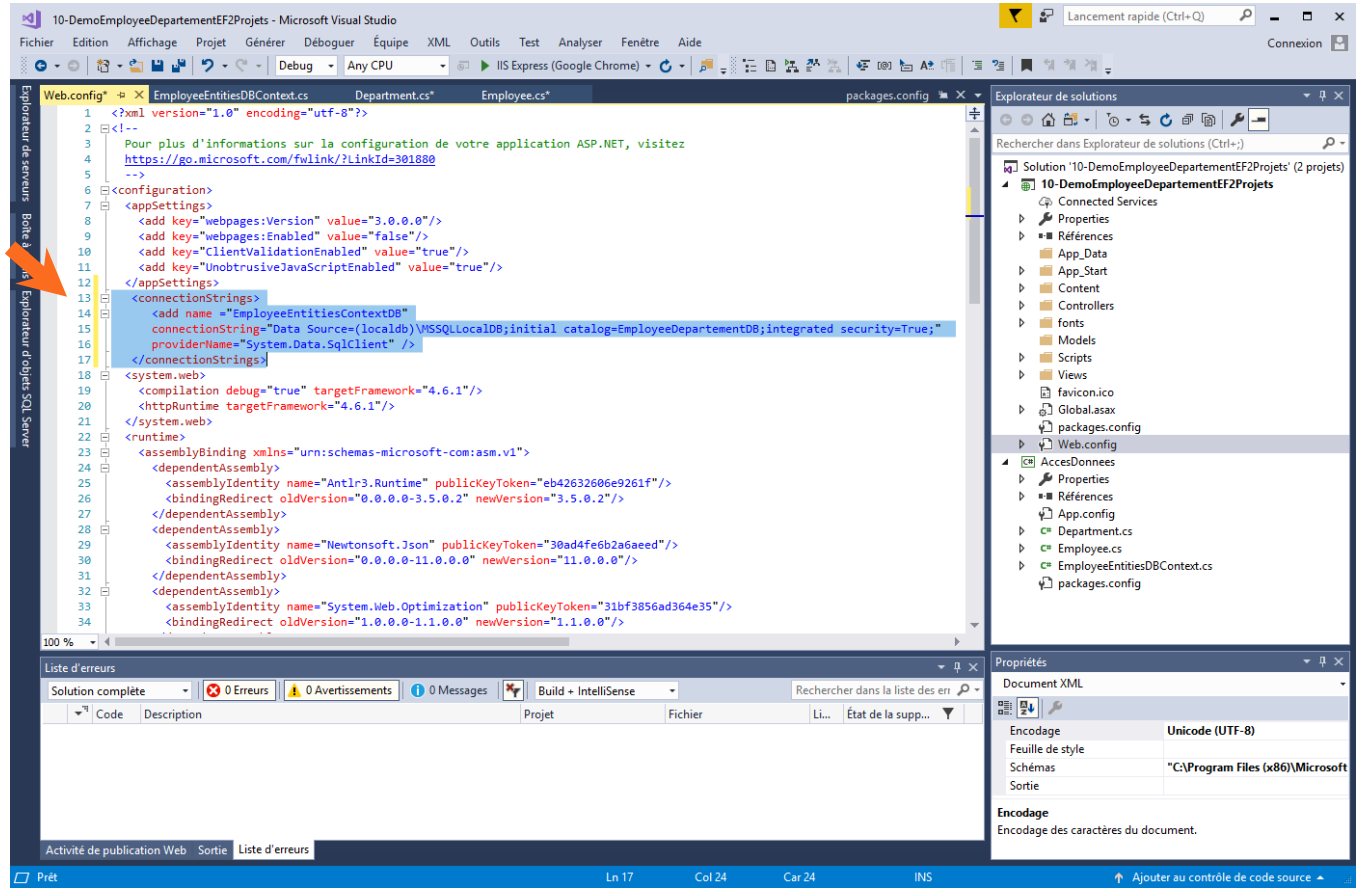
Propriété de clé

Propriétés de navigation

Exemple – Solution avec deux projets

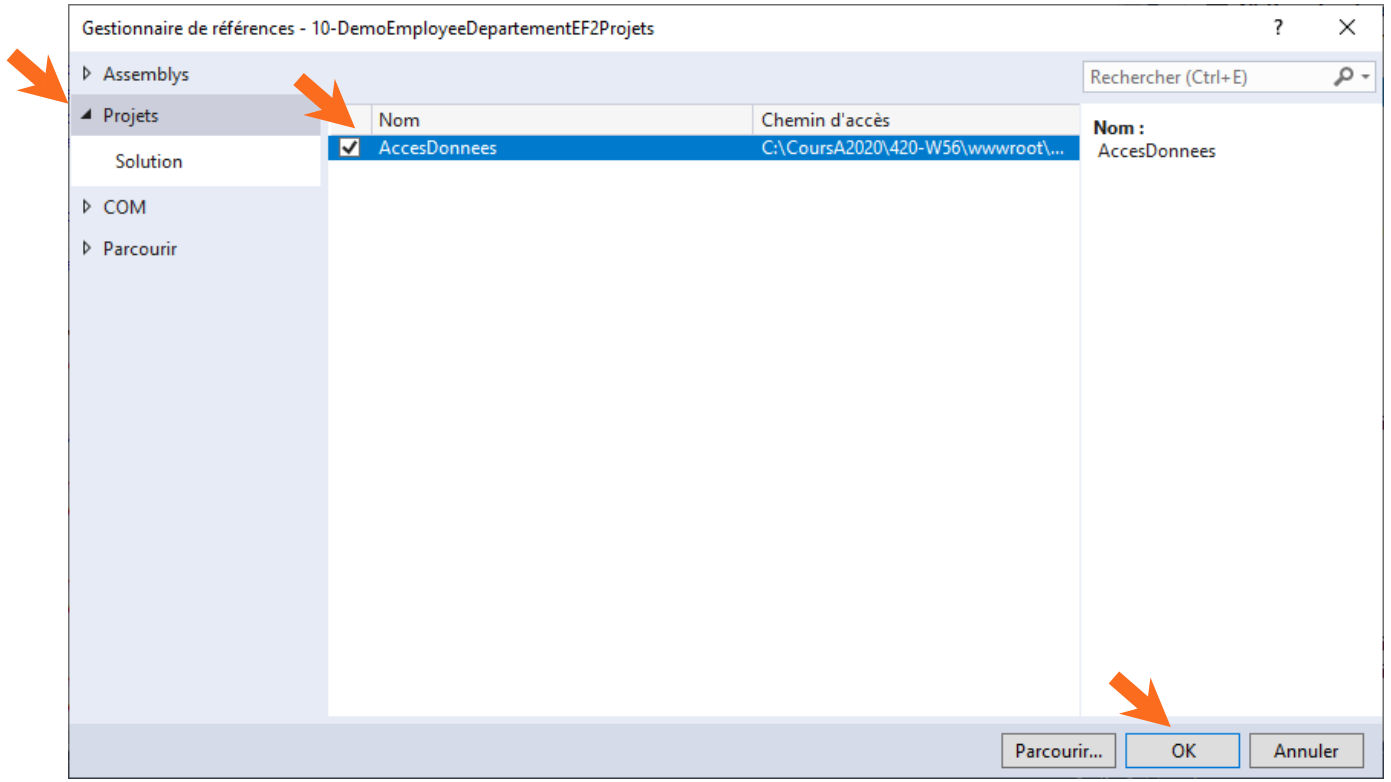
Ajout de la chaine de connexion nommée «EmployeeEntitiesContextDB» dans le fichier Web.config du projet Web : **10-DemoEmployeeDepartementEF2Projets**.

```
<add name ="EmployeeEntitiesContextDB"
      connectionString="Data
Source=(localdb)\MSSQLLocalDB;initial
catalog=EmployeeDepartementDB;integrated
security=True;"

providerName="System.Data.SqlClient" />
```



Exemple – Solution avec deux projets

Utilisation du projet «AccesDonnees» dans le projet MVC «**10-DemoEmployeeDepartementEF2Projets**». Pour cela cliquer sur le dossier «Références» du projet «**10-DemoEmployeeDepartementEF2Projets**» → «Ajouter une référence …». On veut ajouter une référence à un projet.



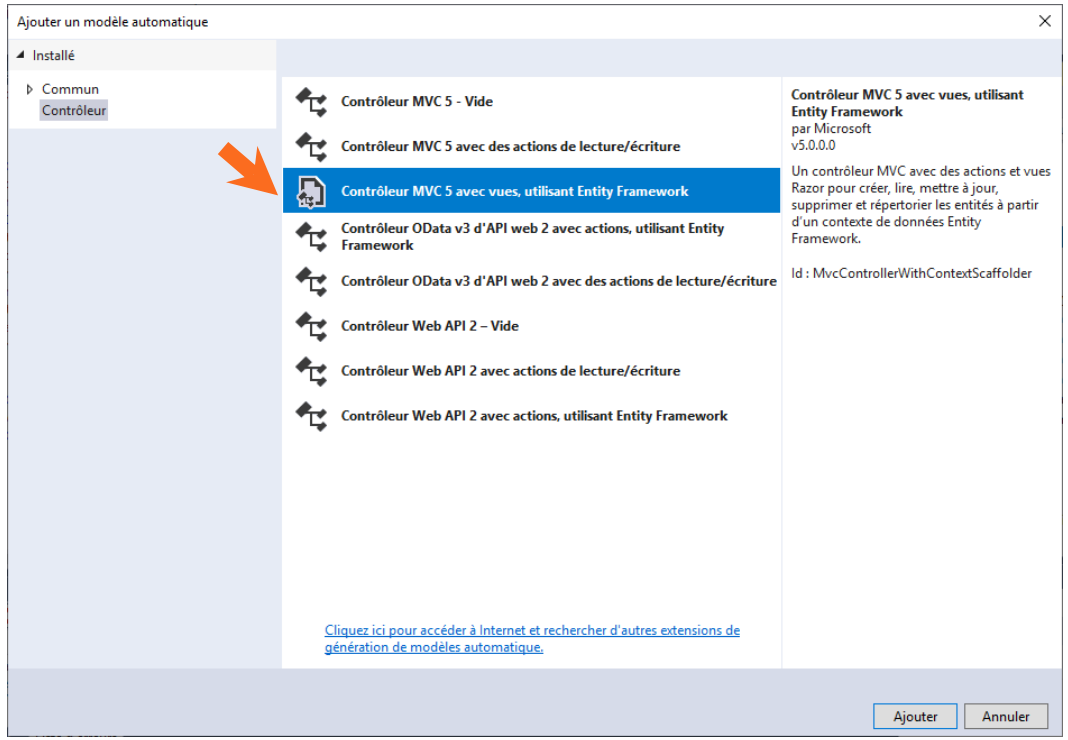
Générer la solution.

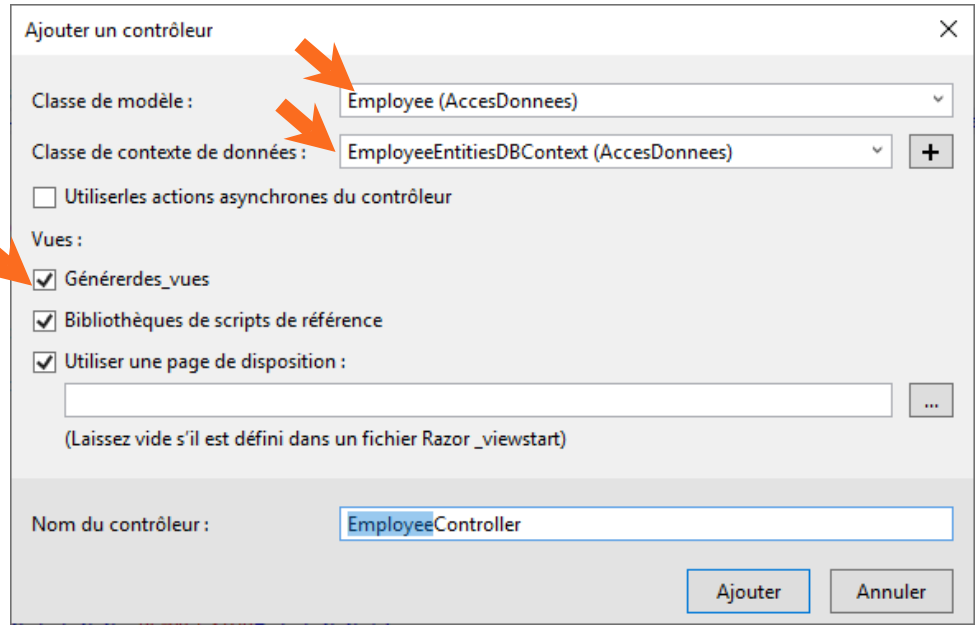
Exemple – Solution avec deux projets

Ajout d'un contrôleur Employee au projet 10-DemoEmployeeDepartementEF2Projets :

Faire un clic droit sur le dossier «Controllers» du projet >> Ajouter >> Contrôleur.

Sélectionner le modèle Contrôleur MVC 5 avec vues, utilisant Entity Framework.





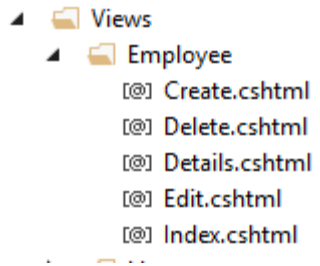
Exemple – Solution avec deux projets

Visual studio vas créer automatiquement dans le contrôleur **Employee** les méthodes d'actions suivantes :

- Index
- Create (GET, POST)
- Edit (GET, POST)
- Delete (GET, POST)
- Details

De plus, pour chaque méthode d'action de contrôleur Visual studio ajoute une vue. Au final, on doit trouver dans le dossier **Views/Employee** les vues suivantes :

- Index.cshtml
- Create.cshtml
- Edit.cshtml
- Delete.cshtml
- Details.cshtml

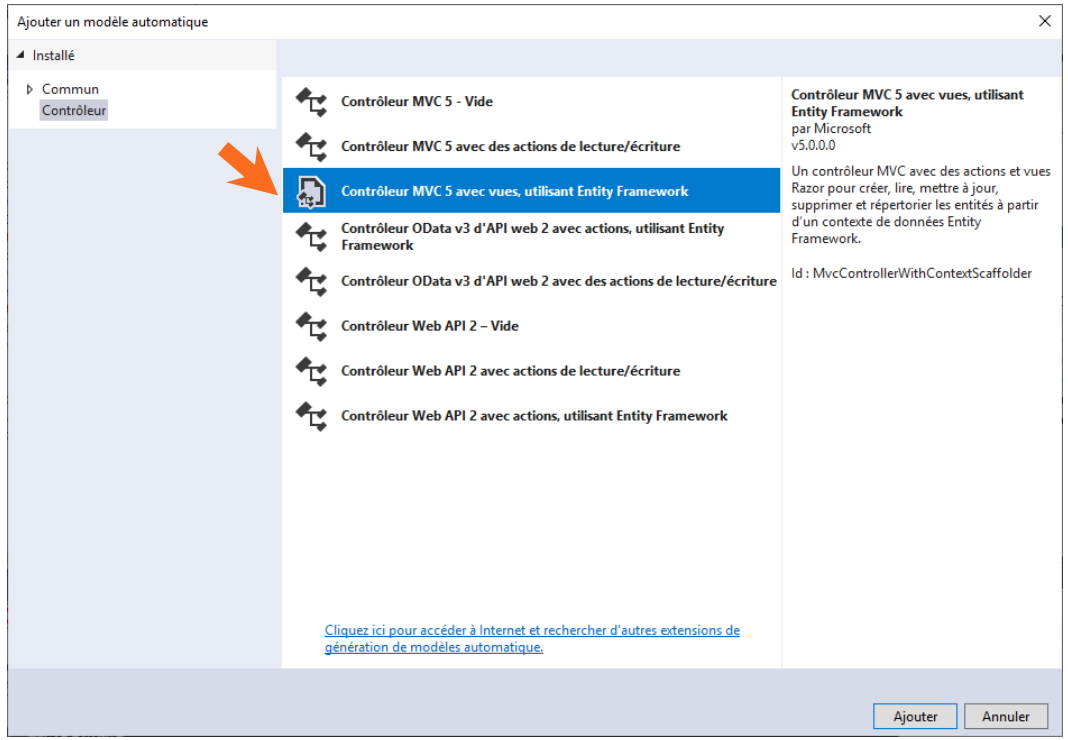


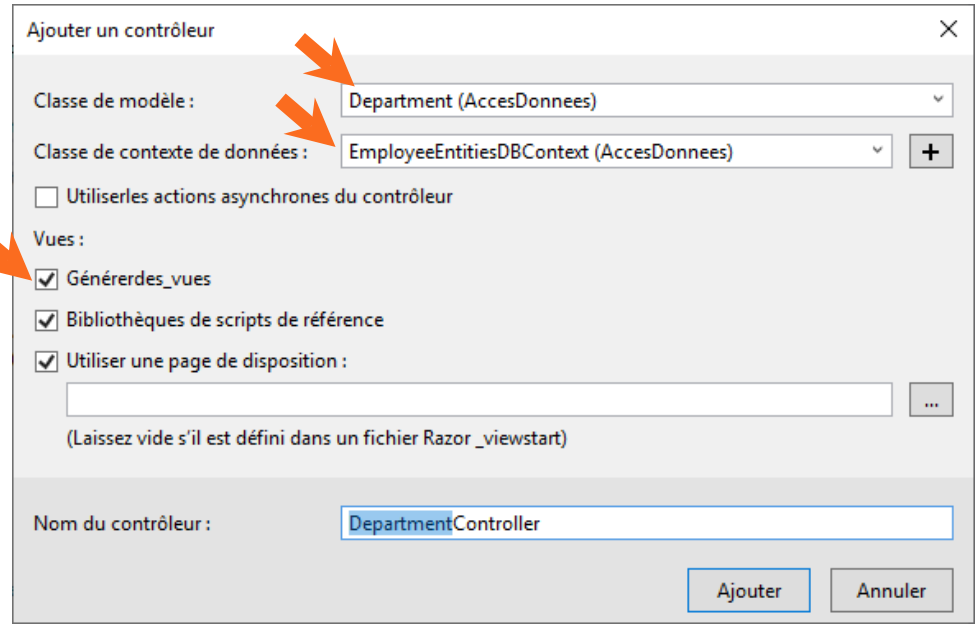
Exemple – Solution avec deux projets

Ajout d'un contrôleur Department au projet 10-DemoEmployeeDepartementEF2Projets :

Faire un clic droit sur le dossier «Controllers» du projet >> Ajouter >> Contrôleur.

Sélectionner le modèle Contrôleur MVC 5 avec vues, utilisant Entity Framework.





Exemple – Solution avec deux projets

Visual studio vas créer automatiquement dans le contrôleur **Department** les méthodes d'actions suivantes :

- Index
- Create (GET, POST)
- Edit (GET, POST)
- Delete (GET, POST)
- Details

De plus, pour chaque méthode d'action de contrôleur Visual studio ajoute une vue. Au final, on doit trouver dans le dossier **Views/Department** les vues suivantes :

- Index.cshtml
- Create.cshtml
- Edit.cshtml
- Delete.cshtml
- Details.cshtml