I. Installer l'outil Entity Framework core

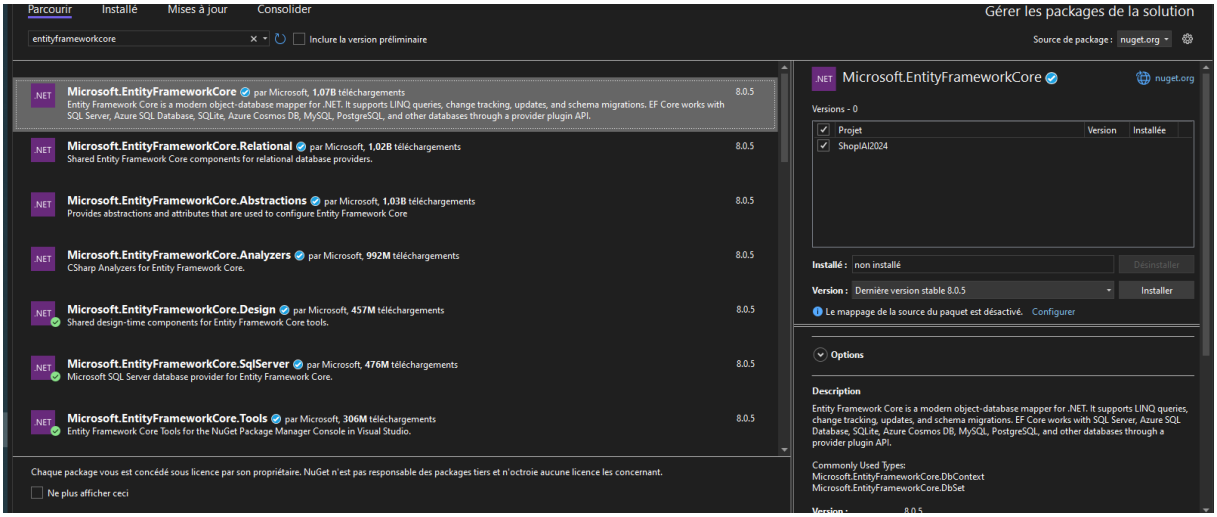
```
dotnet tool install --global dotnet-ef --version 8.*
```

II. Dépendances

- Microsoft.EntityFrameworkCore
- Microsoft.EntityFrameworkCore.Tools
- Pomelo.EntityFrameworkCore.MySql
- Microsoft.VisualStudio.Web.CodeGeneration.Design

Pour installer les packages

- Rechercher dans la barre de recherche dans l'onglet "Parcourir"
- Sélectionner le package dans la liste
- Sélectionner les projet pour lesquels installer le package en le cochant dans le bloc à droite
- Sélectionner la version dans la liste déroulante "Version"
- Cliquer sur **"Installer"**
- Accepter les licences



III. Connexion de base de données

1. Création du contexte de base de données

Exemple :

```
public class MyProjectDbContext : DbContext
{
    public MyProjectDbContext(DbContextOptions<MyProjectDbContext>
options) : base (options)
    { }

    public DbSet<Produit> Produits => Set< Produit >();
}
```

2. Ajout aux services

Exemple : Cas de MySql

```
            // Récupérer la chaine de connexion.
string connString =
builder.Configuration.GetConnectionString("ShopAppConnection");

            // Add services to the container.
builder.Services.AddDbContext<ShopAppDbContext>(options =>
      {
      options.UseMySql(connString,ServerVersion.AutoDetect(connString));

                });
```

3. Ajout migration et mise à jour de la base de données

Dans Visual Studio

- Ouvrir Outils ➔ Gestionnaire de package Nuget ➔ Console du gestionnaire de package
- Exécuter la commande « **Add-Migration « nomMigration »** » ou la commande **dotnet ef migrations add « nomMigration »**

  Si erreur par rapport au chemin du projet en exécutant la commande **dotnet ef migrations add « nomMigration »**, se positionner sur le dossier du projet avec la commande DOS **cd « Dossier »**
- Puis la commande «**update-database** » pour appliquer la migration sur la base de données :
  - A chaque modification de la structure du projet il est possible de générer une migration et de l'appliquer sur la base afin de la mettre à jour
-

4. Quelques annotations pour la validation des champs de formulaire

- **Required** : indique que la propriété est un champ obligatoire
- **NotMapped** : Exclut une propriété ou une classe du DbContext
- **Table** : Permet de donner un nom spécifique en Bd à une table
- **Column :** Permet de donner un nom spécifique à cette propriété dans la table
- **MaxLength** : définit une longueur maximale pour un champ en base de données.
-

- **DisplayName** : définit le texte à utiliser dans les champs de formulaire et les messages de validation
- **StringLength** : définit une longueur minimale ou maximale pour un champ de chaîne.
- **Range**: donne une valeur maximale et minimale pour un champ numérique
- **Bind** : répertorie les champs à exclure ou inclure lors de la liaison de valeurs de paramètre ou de formulaire à des propriétés de modèle
- **ScaffoldColumn** : permet de masquer les champs des formulaires de l'éditeur
-