Blazor

///// Installer les packages NuGet :

dotnet add package Microsoft.EntityFrameworkCore --version 7.0
dotnet add package Npgsql.EntityFrameworkCore.PostgreSQL --version 7.0

dotnet add package Microsoft.EntityFrameworkCore --version 6.0.0
dotnet add package Npgsql.EntityFrameworkCore.PostgreSQL --version 6.0.0

dotnet add package Microsoft.EntityFrameworkCore.Design --version 7.0.0
dotnet add package Microsoft.EntityFrameworkCore.Tools --version 7.0.0

//installation de l'outil EntityFrameworkCore

dotnet tool install --global dotnet-ef

// creation d'une migration

dotnet ef migrations add NomDeVotreMigration

// mis à jour de la base de donnée

dotnet ef database update

// listes des outils installés

dotnet tool list -g

Doc

Entity Framework (EF) est un cadre de mappage objet-relationnel (ORM) fourni par Microsoft. Il simplifie l'accès aux données dans vos applications en vous permettant de travailler avec des objets et des classes plutôt qu'avec des tables de base de données et des requêtes SQL. EF fournit un ensemble d'API et d'outils pour interagir avec les bases de données relationnelles, et il prend en charge divers moteurs de base de données.

Voici un bref aperçu des concepts et sujets clés liés à Entity Framework :

```
DbContext :
    La classe DbContext est cruciale dans Entity Framework et représente une session avec la base de données.
    Elle est responsable de la gestion des connexions à la base de données, du caching et de la fourniture d'un ensemble d'API pour interagir avec la base de données.

csharp
```

public class VotreDbContext : DbContext
{

public DbSet VosEntites { get; set; }

```
// Autres configurations et propriétés DbSet...
```

}

Entité :

```
Une entité dans Entity Framework représente une
classe de modèle de données qui correspond à une
table de base de données.
Les propriétés de la classe d'entité
correspondent généralement aux colonnes de la
table de base de données.
```

csharp

public class VotreEntite
{
public int Id { get; set; }
public string Nom { get; set; }
// Autres propriétés...
}

DbSet :

```
DbSet est une collection d'entités qui représente
une table de base de données ou un résultat de
requête.
Il vous permet d'effectuer des opérations CRUD
```

```
(Créer, Lire, Mettre à jour, Supprimer) sur les
entités.
```

csharp

public DbSet VosEntites { get; set; }

Migrations :

```
Les migrations sont utilisées pour faire évoluer
le schéma de la base de données au fil du temps
en fonction des changements apportés à votre
modèle de données.
Vous pouvez créer et appliquer des migrations
pour mettre à jour le schéma de la base de
données.
```

bash

dotnet ef migrations add NomDeVotreMigration
dotnet ef database update

LINQ to Entities :

```
Entity Framework prend en charge Language-
Integrated Query (LINQ) pour interroger les
données.
LINQ vous permet d'écrire des requêtes de manière
sécurisée en utilisant C# ou VB.NET.
```

csharp

var query = from entite in dbContext.VosEntites
where entite.Nom == "UnNom"
select entite;

Fluent API :

```
Fluent API fournit une alternative pour
configurer votre modèle, les relations et
d'autres aspects d'Entity Framework.
Elle est souvent utilisée dans la méthode
OnModelCreating de votre classe DbContext.
```

csharp

protected override void OnModelCreating(ModelBuilder modelBuilder)
{
modelBuilder.Entity().Property(e => e.Nom).IsRequired();
// Autres configurations...
}

Fournisseurs de base de données :

```
Entity Framework prend en charge divers
fournisseurs de bases de données tels que SQL
Server, SQLite, PostgreSQL, etc.
Vous pouvez spécifier le fournisseur dans la
chaîne de connexion et le configurer dans votre
fichier Startup.cs ou Program.cs.
```

csharp

services.AddDbContext(options =>
options.UseSqlServer(Configuration.GetConnectionString("VotreConnexionDb")));

Opérations asynchrones :

```
Entity Framework prend en charge les opérations asynchrones, vous permettant d'effectuer des opérations sur la base de données de manière asynchrone.
```

csharp

```
var entite = await dbContext.VosEntites.FindAsync(id);
```

Ceci est seulement un aperçu, et il y a beaucoup plus à apprendre sur Entity Framework, y compris des sujets avancés tels que les relations, les transactions, l'optimisation des performances, etc. Lorsque vous travaillez avec Entity Framework, il est important de comprendre la version spécifique que vous utilisez, car EF a évolué au fil du temps avec différentes versions (EF6, EF Core). Consultez la documentation officielle de Microsoft pour la version que vous utilisez pour des informations plus détaillées et spécifiques à la version.