RIL 2020 - MODULE ASP.NET CORE

MISE EN PRATIQUE : EF CORE, REPOSITORY PATTERN & TESTS UNITAIRES

Objectif : A partir d'une solution .NET Core, créer une table dans une base de données avec les scripts de migrations nécessaires, lui adjoindre un repository à placer sous tests unitaires.

Durée : 2h30

Exercice :

1. Récupération de la solution prévue pour exécuter l'exercice.
    a. Vérifier que la solution passe la compilation
    b. Vérifier l'ensemble des fichiers de configuration au niveau des chaines de connexion qui doivent correspondre aux instances SQL (MySQL/MariaDB) installées sur les postes (user, password de connexion)
    c. Vérifier/Installer les outils de migrations avec la commande suivante (PowerShell) :

        dotnet tool install --global dotnet-ef

2. Vous devez créer une table en mode « Code First » avec un script de migration dédié.
    a. La table doit porter le nom « Animal » et doit comporter les champs suivants :
        i. « id » pour la clé primaire (type int)
        ii. « name » (type string, obligatoire, taille maximum 50 caractères)
        iii. « weight » (type double)
        iv. « age » (type int)
        v. « species » (type string)
        vi. « gender » (type int)
        vii. « height » (type double)
        viii. « owner » (type string, obligatoire)
3. Création du repository pour la table créée
    a. Un repository ReadOnly
        i. Méthodes spécifiques à ajouter :
            1. GetAllByOwner
            2. GetAllByOwnerAsync
            3. GetAllBySpecies
            4. GetAllBySpeciesAsync
        ii. Tests unitaires à fournir sur le repository
            1. Find (par id)
            2. FindAsync
            3. GetAll
            4. GetAllAsync
            5. Les méthodes spécifiques demandées plus haut

4. BONUS :
    a. Repository du CRUD Animal & Tests Unitaires
    b. Sur un des projets WEB, faire un CRUD avec le repository créé

RAPPEL :

1. Base de données et migrations
    a. Respecter les conventions établies pour l'exercice
    b. Les commandes de migrations doivent être exécutées dans le répertoire du projet qui effectue les opérations de migration.
    c. Les commandes à utiliser sont les suivantes
        i. **dotnet-ef -v migrations add <script_name>** (créer un script de migration)
        ii. **dotnet-ef -v migrations remove** (effacer le dernier script de migration)
        iii. **dotnet-ef -v database update <script_name>** (appliquer un script de migration à la base de données)
        iv. **dotnet-ef -v database update** (appliquer tous les scripts de migration à la base de données)
2. Code (Repository, classes, etc…)
    a. Respecte les conventions de code établies pendant la formation.