BoVoyage- Projet 2

Étude de la couche métier

Objectifs

- Réaliser la base de données
- Développer une application console C# fonctionnelle
- Travailler en équipe en utilisant GIT

A faire

- A partir du diagramme des classes UML fourni
    - Créer la base de données correspondante
    - Ecrire les classes métier de l'application console
- Ensuite, réaliser une application console
    - Isoler la couche métier et la couche d'accès aux données
    - Penser aux validations métier
    - Utiliser les classes mis à disposition pour faciliter le code de l'application

Critères d'évaluations

Vous serez évalué en binôme, à partir de ce que vous aurez mis à disposition sur votre repository GIT.

- **Base de données (3 points)**
    - La base de données est-elle complète ?
    - Les conventions de nommages sont-elles respectées ?
    - Les relations entre les tables correspondent-elles au diagramme des classes ?
- **GIT (3 points)**
    - Le repository est-il fonctionnel ?
    - Les descriptions des commit (validations) sont-elles claires ?
    - GIT a-t-il été utilisé fréquemment pour échanger et synchroniser le code ?
- **Codage (5 points)**
    - Le code compile-t-il ?
    - Retrouve-t-on les couches applicatives ?
    - Le code est-il optimisé (utilisation de LINQ, EF) ?
    - Les classes mises à disposition ont-elles été utilisées ?
- **Qualité du code (5 points)**
    - Les conventions de nommage et de codage sont-elles respectées ?
    - Le code est-il lisible (bon nommage des variables, paramètres et méthodes) ?
    - Découpage du code en méthodes
- **Respect du cahier des charges (4 points)**
    - Retrouve-t-on dans l'application les fonctionnalités principales du cahier des charges ?
    - Ces fonctionnalités sont-elles fonctionnelles ?
    - L'application gère-t-elle les validations métier ?

Aide-mémoire

Conventions de nommage C#
Pour les noms de classes, méthodes, propriétés, constantes : PascalCase
Pour les noms de variables, champs, paramètres : camelCase

camelCase = premier mot avec une **minuscule** puis chaque mot avec une majuscule
PascalCase = premier mot avec une **majuscule** puis chaque mot avec une majuscule

Conventions de nommage SQL
Les tables ont des noms au pluriel en général

Pour faire le **mapping entre C# et SQL**, voici quelques attributs utiles :
[ForeignKey("nom de la propriété C# qui correspond au champ de la table qui fait la relation ")]
[Table("nom de la table")]
[Column("nom du champ dans la table")]

Dll fournie
Sur le dossier de partage, il y a un fichier dll qui offre plusieurs outils :

- Aide à la saisie pour la console
- Aide à l'affichage de liste
- Classe de base pour les exceptions métier
- Mécanisme de menu et élément de menu
- Base d'un module pour l'application

Pour l'utiliser, créer un dossier _Lib dans le projet, y copier le fichier dll.
Puis au niveau du projet, ajouter une référence en choisissant le fichier copié.

Autres conseils
Pour utiliser Entity Framework, penser à installer le packaget NuGet sur le projet.
Pensez au SaveChanges pour l'ajout, modification et suppression !!