ECF
Développer des composants d'accès aux données

Table des matières

Critères d'évaluation

Les compétences suivantes seront évaluées :

- Développer des composants d'accès aux données en utilisant le design pattern Data Access Object (DAO) et le design pattern Singleton
  - ✓ La programmation en utilisant le design pattern Data Access Object
  - ✓ L'utilisation des preparedStatement
  - ✓ L'utilisation du design pattern Singleton
  - ✓ Les exceptions
  - ✓ Les logs

- Git :
  - ✓ La gestion des branches
  - ✓ La normalisation des noms des commits

1. Ajout au cahier des charges

Vous allez ajouter une persistance de données à votre projet clients/prospects qui se fera avec une base de données MySQL et qui remplacera les collections.

La persistance de données se fera en utilisant le design pattern Data Access Object (DAO), ainsi que le design pattern Singleton.

1. Etape 1 : Conception

- A partir du cahier des charges du projet clients/prospects, rédiger le dictionnaire de données et concevoir le MCD : La BDD comportera au moins 4 tables (client, prospect, adresse, contrat)
  Vous pouvez ajouter d'autres tables si cela vous semble nécessaire ou judicieux

Le développement ne commencera qu'après validation de la partie conception

2. Etape 2 : Développement

- Créer une base de données MySQL
- Créer une branche Git dans votre projet Java clients/prospects dédiée à cette version.
  N'oubliez pas de modifier votre README
- Si vos attributs identifiant de vos classes métier sont de type int, les passer en type Integer
- Supprimer toute la gestion des collections des classes métiers Client et Prospect, Adresse…
- Créer un package DAO

- Dans le package DAO, coder une classe de connexion à cette base de données
  - ✓ Utiliser le Design Pattern Singleton
- Dans le package DAO, créer une classe pour chaque table de votre BDD
- Dans ces classes, coder des méthodes ayant le même nom dans toutes les classes pour :
  - ✓ Lire toute la table (findAll)
  - ✓ Lire un enregistrement en fournissant l'identifiant (findById)
  - ✓ Insérer un enregistrement (create)
  - ✓ Modifier un enregistrement (save)
  - ✓ Supprimer un enregistrement (delete)

- Vous devez donc avoir au moins 5 méthodes dans vos classes.
- Vous pouvez ajouter des méthodes si cela vous semble nécessaire ou judicieux.
- Pour la classe des contrats, ajouter la méthode findByIdClient afin de pouvoir récupérer les contrats d'un client
- Utiliser des preparedStatement SQL dès que nécessaire
- Coder une transaction pour au moins la méthode delete dans les classes.
- N'oubliez pas de générer des messages dans le fichier log en cas de problème avec la base de données

3. En option

- Créer une branche Git dédiée à cette amélioration
  Remplacer les méthodes « create » et « save » par une seule méthode « save » qui, suivant si elle reçoit un identifiant nul ou pas, créera ou mettra à jour un enregistrement dans la base de données
- Coder le Design Pattern DAO dans votre application avec de la généricité et du polymorphisme
- Coder les Design Pattern Factory et AbstractFactory dans votre application et ajouter une partie DAO NoSql avec MongoDB ou avec une gestion de fichiers

2. Travail demandé

1. Conception

- Déposer le MCD au format looping sur Mètis dans :

  Concevoir et développer une application sécurisée organisée en couches - Développer les composants d'accès aux données SQL et NoSQL - ECF Développer des composants d'accès aux données – Conception

2. Développement

- Le développement ne commencera qu'après validation de la partie conception
- Déposer le projet sur Mètis sous la forme d'un zip avant jeudi 22 janvier 2026 à 17h dans :

  Concevoir et développer une application sécurisée organisée en couches - Développer les composants d'accès aux données SQL et NoSQL - ECF Développer des composants d'accès aux données – Programmation

- L'accès au dépôt GitHub me sera toujours autorisé