Examen Architecture Dot Net, AOP et Design patterns

Durée : 3H00

A- Partie Programmation Orientée Aspect et Design patterns

On souhaite créer une application qui permet d'effectuer un traitement quelconque. Dans notre cas, ce traitement consiste à effectuer un simple calcul représenté par la somme suivante :

$$S(n) = \frac{1}{1!} + \frac{2}{2!} + \cdots + \frac{n}{\mathrm{n!}}$$

Dans cette application, nous devrions séparer les différents aspects métier et techniques en utilisant le principe de la programmation orientée aspect et les design patterns appropriés. Cette application devrait être également fermée à la modification et ouverte à l'extension. On vous fournit une première version de cette application dans laquelle nous avons implémenté uniquement l'aspect métier. L'application fournie se compose d'une interface ICalculateur, une implémentation de cette interface CalculateurImpl, une application de Test et un fichier de configuration config.txt.

Travail demandé :

1. Analyser et Tester la version fournie de cette application.
2. Créer un aspect en utilisant AspectJ, qui permet de journaliser la durée d'exécution de la méthode *calculer()*.
3. Créer un autre aspect qui permet de sécuriser cette application en obligeant l'utilisateur à saisir un login et un mot de passe.
4. Créer un aspect qui représente un cache des appels de la méthode *calculer ()*. Cet aspect devrait intercepter les appels de cette méthode. Ensuite vérifie si le somme relative à au nombre n fourni n'est pas stockée préalablement de une collection de type Map. Si c'est le cas, il retourne cette somme sans avoir besoin de recalculer la somme. Sinon, il fait appel à la méthode *calculSomme()* pour récupérer le résultat et le stocke dans son cache (Collection Map) avant de le retourner.
5. En utilisant le design pattern Proxy ou Decorator, proposer une autre solution à la question 4 sans utiliser AspectJ et sans modifier le code source de l'application de base.

B- Partie Architecture .Net

On souhaite créer une application web basée sur le Framework Asp.Net Version 5. Cette application devrait permettre de gérer des filières appartenant à des départements. Un département peut créer plusieurs filières et une filière appartient à un seul département. Chaque département est défini par un identifiant numérique, le nom du département et le nom du chef de département. Une filière est définie par un identifiant numérique, le nom de la filière et la durée de la formation.

L'application devrait permettre de réaliser les spécifications fonctionnelles suivantes :

- Gérer les départements :
- Gérer les filières :

L'application devrait respecter aussi les spécifications techniques suivantes :

- Le serveur d'application utilisé est IIS Express
- Les données sont stockées dans une base de données relationnelle de type SQL Server Express
- L'application se compose de trois couches :
  - La couche DAO est basée sur Entity Framework pour le mapping objet relationnel. Cette couche est principalement formée par les deux entités Departement et Filiere et la classe de qui définit le contexte de persistance MyDbContext basé sur Entity Framework.
  - La couche Web est basée sur Asp.net MVC 6
  - L'injection des dépendances est basée sur Microsoft Dependency Injection

Travail à réaliser :

Rendre un rapport au format PDF et Le Projet contenant les réponses aux questions :

1- Etablir l'architecture technique du projet
2- Etablir un diagramme de classes montrant les entités et la couche métier de l'application.
3- Créer un projet ASP.net 5 (Web Application)
4- Couche DAO
   a. Créer les deux entités Departement et Filiere.
   b. Créer la classe du contexte de persistance MyDbContext.
5- Déployer MyDbContext et les deux implémentations de la couche métier dans Startup.cs
6- Couche Web : Créer les contrôleurs et les vues qui permettent de :
   a. Chercher les filières dont le nom contient un mot clé.
   b. Saisir et à ajouter une nouvelle filière.
   c. Afficher les filières d'un département sélectionné dans une one de liste déroulante.