

Bonus : Diagramme de classes et Principes de conception

Ahmed

November 20, 2025

Question 1 : Diagramme de classes du module de gestion des cours

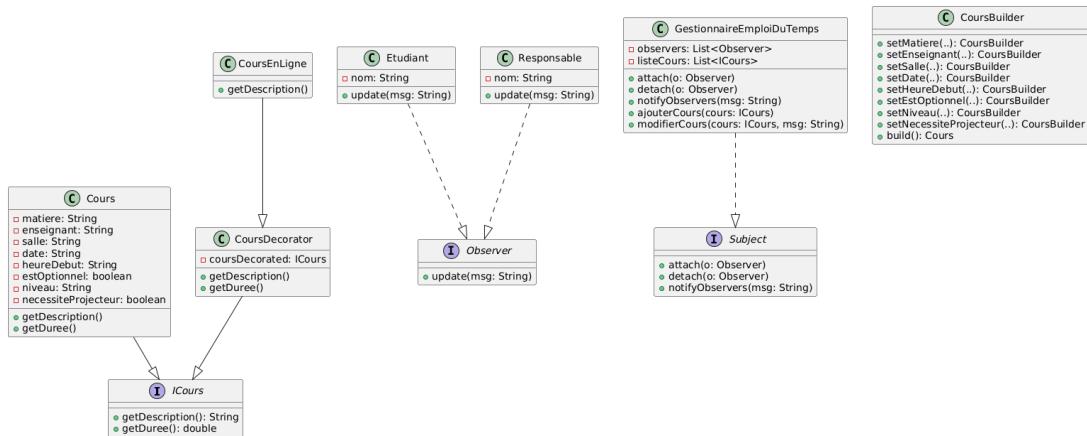


Figure 1: Diagramme de classes UML du module de gestion des cours.

Question 2 : Respect des principes de conception logicielle

L'analyse des principes SOLID appliqués à ce code est la suivante :

- **Single Responsibility Principle (SRP)** : Partiellement respecté. La classe `GestionnaireEmploiDuTemps` gère à la fois la liste des cours et la notification des observateurs. Pour être totalement conforme, on pourrait séparer ces responsabilités en deux classes distinctes.
- **Open/Closed Principle (OCP)** : Respecté. Les cours peuvent être étendus via des décorateurs (`CoursEnLigne`, `CoursEnAnglais`, etc.) sans modifier les classes existantes.
- **Liskov Substitution Principle (LSP)** : Respecté. Les décorateurs et les classes dérivées peuvent remplacer `ICours` sans altérer le comportement attendu.
- **Interface Segregation Principle (ISP)** : Respecté. Les interfaces `ICours` et `Observer` sont simples et ciblées.
- **Dependency Inversion Principle (DIP)** : Respecté. La classe `GestionnaireEmploiDuTemps` dépend de l'interface abstraite `Observer` plutôt que des classes concrètes.

Conclusion : Le code est bien structuré, flexible et extensible grâce aux patrons Builder, Observer et Decorator. Le seul petit manquement est la responsabilité multiple dans la classe `GestionnaireEmploiDuTemps`, ce qui viole légèrement le principe SRP.