

## BONUS 2 — Analyse des principes SOLID

### 1. SRP — Single Responsibility Principle

Respecté.

Chaque classe a une seule responsabilité : Cours représente les données, CoursBuilder construit un Cours, les décorateurs ajoutent des fonctionnalités, le gestionnaire gère les notifications et les observateurs ne font que recevoir des messages.

### 2. OCP — Open Closed Principle

Respecté.

Le code est ouvert à l'extension mais fermé à la modification grâce au pattern Decorator. De nouveaux comportements peuvent être ajoutés sans toucher aux classes existantes.

### 3. LSP — Liskov Substitution Principle

Respecté.

Les décorateurs et les cours implémentent tous ICours et sont interchangeables sans casser le comportement.

### 4. ISP — Interface Segregation Principle

Respecté.

Les interfaces ICours, Observer et Subject sont petites, cohérentes et ne forcent pas l'implémentation de méthodes inutiles.

### 5. DIP — Dependency Inversion Principle

Respecté.

Le gestionnaire dépend de l'interface Observer plutôt que des classes concrètes. Les décorateurs dépendent aussi de l'abstraction ICours.

## Conclusion

Le projet respecte les principes SOLID, ce qui garantit une bonne modularité et une architecture extensible.