UQÀM

Faculté des sciences
Département d'informatique
Cours: INF5153 - Génie logiciel: conception - Hiver 2025
Enseignant : Skander Jabouzi

Projet de session – L2
Système de Gestion de Commandes pour un Restaurant
26 février 2025

Objectif :

Développer un **système de gestion des commandes** pour un restaurant en appliquant les **principes SOLID** et en intégrant **les patrons de conception (créationnels et structurels)**.

Exigences du Projet :

- **Gestion des commandes** : Un client peut commander un ou plusieurs plats, avec des options de personnalisation.
- **Gestion des menus et promotions** : Possibilité de créer des **menus composés** et d'y appliquer des remises.
- **Gestion du stock** : Suivi des ingrédients et mises à jour en fonction des commandes.
- **Système de paiement** : Prise en charge de plusieurs modes de paiement (carte, cash, mobile).
- **Interfaces utilisateurs** : Différentes interfaces pour les **serveurs, la cuisine et la caisse**.
- **Base de données simplifiée** : Utilisation d'une **liste Java en mémoire** au lieu d'une base de données complète.

Concepts Appliqués :

Principes SOLID

Le projet doit respecter **un minimum** de principes SOLID, sans restrictions spécifiques.

Les étudiants devront **justifier dans leur documentation** les principes SOLID appliqués.

Patrons de Conception Utilisés :

- **Patrons Créationnels**

Patron	**Utilisation dans le système**
Factory Method	Création de plats selon leur type (entrée, plat principal, dessert).
Abstract Factory	Génération de **menus régionaux** (exemple : menu italien, asiatique...).
Singleton	Gestion unique du **système de paiement et de l'accès au stock**.
Prototype	Duplication rapide d'un plat pour **faciliter les modifications sans recréer un objet**.
Builder	Construction progressive d'un **menu self-service** où le client peut sélectionner différents plats et boissons.

- **Patrons Structurels**

Patron	**Utilisation dans le système**
Facade	Interface unique pour simplifier la **gestion des commandes et paiements**.

Composite	Permet aux **menus fixes** d'être composés de plusieurs plats.
Decorator	Ajouter dynamiquement des **suppléments aux plats** (exemple : fromage, sauce).
Bridge	Séparation entre **l'interface utilisateur** (serveurs, caisse, cuisine) et la logique métier.
Adapter	Permet la compatibilité avec **différents systèmes de paiement et de facturation**.

Diagrammes UML

Les étudiants doivent fournir **uniquement** des **diagrammes de classes**, un par **module principal** :

• **Gestion des commandes** → **Composite, Decorator et Builder (pour le self-service)**

• **Gestion des stocks** → **Singleton**

• **Gestion des paiements** → **Adapter et Bridge**

• **Système principal** → **Facade**

Code et Résultats Attendus

Le code doit être bien structuré en **modules indépendants** et respecter les principes SOLID.

• **Exécution attendue** : Simulation d'une prise de commande, personnalisation d'un plat, mise à jour du stock et paiement.

• **Résultats affichés** : Détails de la commande validée, mise à jour des stocks et confirmation du paiement.

Barème d'Évaluation

1. Présentation du projet (30%)

Clarté des **diagrammes UML** et explication des **concepts appliqués**.

2. Exactitude du modèle UML (40%)

Respect des exigences et **intégration correcte des patrons**.

3. Qualité du code (20%)

Lisibilité, **modularité** et application des **principes SOLID**.

4. Documentation (10%)

Explication des **choix de conception et interactions entre les modules**.

Livrables

Code source organisé en modules.

Diagrammes UML (**classes uniquement**).

Documentation expliquant l'architecture et les patrons utilisés.

Présentation (optionnelle) : Démonstration du fonctionnement du système.