Projet : Gestion du restaurant "La Bella Tavola"

📌 Contexte

Le restaurant **La Bella Tavola** est un établissement italien renommé, servant des plats traditionnels comme les pizzas, les pâtes fraîches et les desserts typiques. Son gérant souhaite moderniser la gestion du restaurant avec une application logicielle permettant de gérer :

- **Le menu** (plats, boissons, prix).
- **Les commandes** (prises par les serveurs, envoi en cuisine).
- **Le stock d'ingrédients** (mise à jour automatique après chaque commande).
- **Les employés** (serveurs, cuisiniers, gérant).
- **La persistance des données** via une **base de données MySQL**.

Le développement du projet sera divisé en **trois grandes étapes (milestones)**.

📌 Milestone 1 : Mise en place des fonctionnalités de base (Menu & Commandes)

🎯 Objectif principal

Mettre en place la structure initiale de l'application avec une gestion du menu et des commandes **sans persistance** (stockage temporaire en mémoire).

🔷 Attendus détaillés

1. **Définition des classes principales**
    - `Plat` (nom, prix, type : pizza, pâtes, dessert, boisson).
    - `Commande` (liste de plats, total de la commande).
    - `Menu` (liste des plats proposés).
2. **Gestion des commandes**
    - Un serveur peut **ajouter un plat** à une commande.
    - Un serveur peut **afficher le détail** d'une commande.

- L’application doit **calculer automatiquement le total** de la commande.

3. **Premier test de structure**
   - Ajouter plusieurs plats au menu et les afficher.
   - Simuler des commandes et vérifier le calcul des prix.

📌 Livrable attendu

✔ Une **première version fonctionnelle** de l’application en **mode console**, permettant d’afficher le menu et de créer une commande avec des plats variés.

📌 Milestone 2 : Gestion avancée (Stocks, Employés et Base de Données MySQL)

🎯 Objectif principal

Ajouter une **base de données MySQL** pour stocker les plats, les commandes et les stocks. Introduire la gestion des employés et du stock d’ingrédients.

🔷 Attendus détaillés

1. **Connexion à une base de données MySQL**
   - Création des tables `plats`, `commandes`, `stocks`, `employes`.
   - Intégration de **JDBC** pour communiquer avec la base de données.
   - Sauvegarde et récupération des données du menu depuis MySQL.
2. **Gestion des stocks**
   - Chaque plat est composé de **plusieurs ingrédients**.
   - Le stock est **mis à jour automatiquement** après une commande.
   - Si un ingrédient est **en rupture**, la commande du plat concerné est refusée.
   - Une classe `Stock` gère l’ajout, la suppression et la vérification des quantités d’ingrédients.
3. **Gestion des employés**
   - Création de la classe `Employe` avec ses sous-classes :

        - `Serveur` (prend les commandes et les envoie en cuisine).
        - `Cuisinier` (prépare les plats et met à jour le stock).
        - `Gérant` (gère le stock et surveille l'activité).
    - Gestion des employés stockée en base de données.

📌 Livrable attendu

✔ Une application **connectée à une base de données MySQL**, avec gestion des plats et des commandes.

✔ Un système **de gestion des stocks et des employés** opérationnel.

📌 Milestone 3 : Finalisation et fonctionnalités avancées

🎯 Objectif principal

Finaliser l'application avec une interface plus intuitive et ajouter des fonctionnalités avancées comme **un menu interactif en console** et la **persistance des commandes en base de données**.

🔷 Attendus détaillés

1. **Finalisation de la base de données**
    - Stocker **l'historique des commandes** en base de données.
    - Optimiser les requêtes SQL pour une meilleure performance.
2. **Ajout d'un menu interactif en console**
    - Options disponibles :
        - **Afficher le menu** (données chargées depuis MySQL).
        - **Passer une commande** (vérification du stock et ajout à la base).
        - **Afficher l'état du stock** et le modifier (ajout/retrait d'ingrédients).
        - **Consulter l'historique des commandes**.
3. **Tests et robustesse**

- Effectuer des tests approfondis pour s'assurer du bon fonctionnement.
- Vérifier la **cohérence des données** après redémarrage de l'application.

📌 Livrable attendu

✔ Une application complète et interactive en **console**, connectée à **MySQL**, permettant de gérer un restaurant de manière réaliste.

📌 Critères d'évaluation

- **Respect des principes de la POO** (encapsulation, héritage, polymorphisme).
- **Qualité du code** (modularité, clarté et commentaires).
- **Connexion robuste avec MySQL** (gestion des requêtes, optimisations).
- **Interface utilisateur fonctionnelle et intuitive**.
- **Gestion correcte des erreurs et exceptions**.

📌 Consignes finales

- Le projet devra être présenté sous la forme d'un **projet Java structuré** avec des classes bien définies.
- Un rapport d'environ **2 pages** devra expliquer :
  - L'organisation du code.
  - Les choix techniques (héritage, collections, exceptions, persistance).
  - Les tests réalisés.