SmartShop

Contexte :

SmartShop est une application web de gestion commerciale destinée à MicroTech Maroc, distributeur B2B de matériel informatique basé à Casablanca. L'application permettra de gérer un portefeuille de 650 clients actifs avec un système de fidélité à remises progressives et des paiements fractionnés multi-moyens par facture. Le système assurera une traçabilité complète de tous les événements financiers via un historique immuable et optimisera la gestion de la trésorerie.

Notes importantes :

- Cette application est purement backend REST (API uniquement)
- Pas d'interface graphique
- Les tests et démonstrations se feront via un API tester (Postman ou Swagger)
- Authentification par HTTP Session (login/logout)
- Pas de JWT, Pas de Spring Security
- Gestion des rôles : ADMIN (employé MicroTech qui utilise l'application SmartShop) et CLIENT (entreprises clientes qui achètent auprès de MicroTech)

EXIGENCES FONCTIONNELLES

1. Gestion des Clients

- Créer un client
- Consulter les informations d'un client
- Mettre à jour les données d'un client (informations personnelles)
- Suppression d'un client
- Suivre automatiquement :
    - Statistiques : Nombre total de commandes + Montant cumulé : total cumulé des commandes confirmées
    - Date de première et dernière commande
    - Consulter l'historique des commandes en affichant la liste des commandes effectuées par un client spécifique :
        - Identifiant de la commande
        - Date de création de la commande
        - Montant total TTC
        - Statut de la commande (PENDING, CONFIRMED, CANCELED, REJECTED)

2. Système de Fidélité Automatique

- Calcul automatique du niveau basé sur l'historique client :
  - **BASIC** : Client par défaut (0 commande)
  - **SILVER** : À partir de 3 commandes OU 1,000 cumulés
  - **GOLD** : À partir de 10 commandes OU 5,000 cumulés
  - **PLATINUM** : À partir de 20 commandes OU 15,000 cumulés
- Mise à jour du niveau après chaque commande confirmée
- Application des remises selon le niveau actuel :
  - SILVER : 5% si sous-total ≥ 500
  - GOLD : 10% si sous-total ≥ 800
  - PLATINUM : 15% si sous-total ≥ 1200

Sous-total = Montant HT de la commande (avant remises)

EXPLICATIONS DÉTAILLÉES :

PARTIE 1 : Comment on OBTIENT un niveau

Le niveau est calculé automatiquement selon le TOTAL des commandes et dépenses du client :

- Nombre total de commandes passées (confirmées par l'admin)
- Montant total dépensé depuis son inscription (la création du client)

PARTIE 2 : Comment on UTILISE ce niveau

Une fois le niveau acquis, il donne droit à des remises SUR LES FUTURES COMMANDES si elles atteignent un certain montant

EXEMPLE COMPLET : Historique d'un Client

Amine s'inscrit → Niveau : BASIC

Janvier – Février : Acquisition du niveau SILVER

Commande 1 : 250DH	**Commande 2 : 350DH**	**Commande 3 : 450DH**
• Niveau actuel : BASIC • Remise : 0% (client BASIC n'a pas de remise) • Après confirmation par l'ADMIN : 1 commande, 250DH cumulé → reste BASIC	• Niveau actuel : BASIC • Remise : 0% • Après confirmation par l'ADMIN : 2 commandes, 600DH cumulé → reste BASIC	Niveau actuel : BASIC (pas encore validée) Remise : 0% Après confirmation par l'ADMIN : 3 commandes, 1050DH cumulé DEVIENT SILVER

Mars – Avril : Utilisation des avantages SILVER

Commande 4 : 600DH	**Commande 5 : 3500DH**	**Commande 6 : 900DH**
• Niveau actuel : SILVER • Remise : 5% = -30DH (SILVER + commande ≥ 500DH) • Prix final : 570DH • Après confirmation : 4 commandes, 1650DH cumulé → reste SILVER	• Niveau actuel : SILVER • Remise : 5% = -175DH • Prix final : 3325DH • Après confirmation par l'ADMIN : 5 commandes, 5150DH cumulé → DEVIENT GOLD	• Niveau actuel : GOLD • Remise : 10% = -90DH (GOLD + commande ≥ 800DH) • Prix final : 810DH • Après confirmation par l'ADMIN : … etc

3. Gestion des Produits

- **Ajouter des produits**
- **Modifier** les informations produits
- **Supprimer** des produits (soft delete si commandes existantes)
- **Consulter la liste des produits** avec filtres et pagination

Remarque : Si un produit est utilisé dans des commandes existantes, et qu'on aimerait le supprimer il est marqué comme "deleted" (soft delete) Il n'apparaît plus dans la liste des produit mais reste visible dans les anciennes commandes

4. Gestion des Commandes

- **Créer une commande** multi-produits avec quantités
- **Valider les prérequis** :
  - Disponibilité du stock pour chaque produit
- **Appliquer les remises cumulatives** :
  - Remise fidélité selon niveau client
  - Code promo PROMO-XXXX (+5% si valide)
- **Calculer automatiquement** :
  - Sous-total HT : somme de (prix HT × quantité de produit)
  - Montant remise totale (code promo ou niveau de fidélité)
  - Montant HT après remise = Sous-total HT - Montant remise
  - TVA 20% (configurable) = Montant HT après remise × 20%
  - Total TTC = Montant HT après remise + TVA

Note : La TVA se calcule sur le montant APRÈS remise (standard au Maroc)

Exemple : Sous-total 1,000 DH - Remise 100 DH = 900 DH → TVA 180 DH → Total 1,080 DH

- **Mettre à jour après validation** :
  - Décrémenter le stock produits
  - Actualiser statistiques client (totalOrders, totalSpent)
  - Recalculer niveau fidélité
- **Gérer les statuts** :
  - PENDING : en attente validation
  - CONFIRMED : commande validée par ADMIN (après paiement complet)
  - CANCELED : annulée manuellement par ADMIN
  - REJECTED : refusée (stock insuffisant)

5. Système de Paiements Multi-Moyens

Moyens de Paiement Acceptés

ESPÈCES	CHÈQUE	VIREMENT
• Limite légale : 20,000 DH maximum par paiement (Art. 193 CGI) • Paiement immédiat • Statuts possibles : Encaissé • Nécessite un reçu	• Peut être différé (date d'échéance future) • Statuts possibles : En attente / Encaissé / Rejeté • Nécessite : numéro, banque, échéance	• Paiement immédiat ou différé • Statuts possibles : En attente / Encaissé / Rejeté • Nécessite : référence, banque

Une commande peut être payée en plusieurs fois avec différents moyens de paiement.

Règle importante : Une commande doit être totalement payée (montant_restant = 0) avant qu'un ADMIN puisse la valider et la faire passer au statut CONFIRMED.

Exemple de paiement fractionné :

Commande de 10,000 DH		
Paiement 1 - Le 05/11/2025	Paiement 2 - Le 08/11/2025	Paiement 3 - Le 12/11/2025
Montant : 6,000 DH Moyen : ESPECES Référence : REÇU-001 Statut : Encaissé Restant dû : 4,000 DH →Commande reste PENDING	Montant : 3,000 DH Moyen : CHÈQUE Référence : CHQ-7894561 Banque : BMCE Bank Échéance : 20/11/2025 Statut : En attente d'encaissement Restant dû : 1,000 DH →Commande reste PENDING	Montant : 1,000 DH Moyen : VIREMENT Référence : VIR-2025-11-12-4521 Banque : Attijariwafa Bank Statut : Encaissé Restant dû : 0 DH →Commande peut maintenant être validée par ADMIN → CONFIRMED

6. Règles Métier Critiques

Validation stock : quantité_demandée ≤ stock_disponible

- Arrondis : tous les montants à 2 décimales
- Codes promo : format strict PROMO-XXXX, usage unique possibl @Pattern(regexp = "PROMO-[A-Z0-9]{4}")
- Taux de TVA : 20% par défaut (paramétrable via configuration)

Validations métier

- Une commande sans client ou sans article est refusée
- Une commande ne peut être validée (CONFIRMED) par l'admin que si totalement payée
- Les messages d'erreur doivent être clairs

EXIGENCES TECHNIQUES :

1. Architecture & Stack Technique

- Type d'application : Backend REST API uniquement (pas de frontend)
- Tests et simulation : Via Postman ou Swagger
- Framework : Spring Boot
- Java : Version 8 ou plus
- API : REST avec JSON
- Base de données : PostgreSQL ou MySQL
- ORM : Spring Data (JPA/Hibernate)
- Tests Unitaires : JUnit, Mockito
- Architecture en couches (Controller, Service, Repository, Entity, DTO, ...)
- Validation des données avec annotations
- Utilisation des interfaces et implémentation
- Gestion centralisée des exceptions
- Lombok et Builder Pattern pour simplifier la gestion des entités
- MapStruct pour la conversion entre les entités, DTO et View Models
- Concepts Java fondamentaux : Stream API, Java Time API, Lambda expressions Authentification : HTTP Session
- Pas de JWT, Pas de Spring Security, Session simple avec login/logout

2. Modèle de Données

- **User :** id, username, password, role (ADMIN/CLIENT)
- **Client:** id, nom, email, niveau de fidélité
- **Product** : id, nom, prix unitaire, stock disponible
- **Commande:** id, client, liste d'articles, date, sous-total, remise, TVA, total, code promo, statut, montant_restant.
- **OrderItem :** id, produit, quantité, prix unitaire, total ligne
- **Paiement:** id, id_commande, numero_paiement : Numéro séquentiel du paiement pour cette facture (1, 2, 3...), montant, type_paiement, date_paiement : Date où le client a effectué le paiement, date_encaissement : Date d'encaissement effectif.

Note : Vous pouvez ajouter d'autres attributs pertinents selon votre analyse

3. Enums du Système

- **UserRole:** représente le rôle de l'utilisateur :
  - ADMIN : Employé de MicroTech (gestion complète du système)
  - CLIENT : Entreprise cliente (consultation uniquement)
- **CustomerTier :** représente le niveau de fidélité du client :
  - BASIC
  - SILVER
  - GOLD
  - PLATINUM

- **OrderStatus**: représente le statut d'une commande :
  - PENDING : commande en attente de validation par l'ADMIN
  - CONFIRMED : commande validée par ADMIN (après paiement complet)
  - CANCELED : commande annulée manuellement par ADMIN (uniquement si PENDING)
  - REJECTED : commande refusée pour cause de stock manquant
- **PaymentStatus :** représente le statut d'un paiement :
  - EN_ATTENTE : paiement reçu mais non encore encaissé
  - ENCAISSÉ : montant effectivement reçu
  - REJETÉ : paiement rejeté

4. Gestion des statuts de commande :

Transitions automatiques (gérées par le système) :

- PENDING → REJECTED : si stock insuffisant au moment de la création de la commande

Transitions manuelles (via endpoint API – ADMIN uniquement) :

- PENDING → CONFIRMED : validation par ADMIN (après vérification du paiement complet)
- PENDING → CANCELED : annulation par ADMIN
- Statuts finaux : CONFIRMED, REJECTED, CANCELED (aucune modification de la commande sera possible)

Matrice de permission

CLIENT peut uniquement :	ADMIN peut tout faire :
Se connecter Consulter SES PROPRES données : • Profil et niveau de fidélité • Historique des commandes • Statistiques (nombre de commandes, montant cumulé) Consulter la liste des produits (lecture seule) NE PEUT PAS créer, modifier, supprimer quoi que ce soit NE PEUT PAS voir les données des autres clients	Toutes les opérations (CRUD complet) Voir tous les clients Créer des commandes pour n'importe quel client Enregistrer le paiement Valider, annuler et rejeter les commandes

5. Gestion des Erreurs

- La classe @ControllerAdvice centralise la gestion des exceptions.
- Les erreurs doivent être traduites en codes HTTP cohérents :
    - 400 → erreur de validation
    - 401 → non authentifié
    - 403 → accès refusé (permissions insuffisantes)
    - 404 → ressource inexistante
    - 422 → règle métier violée (stock insuffisant, commande déjà validée, etc.)
    - 500 → erreur interne
- Chaque réponse JSON doit inclure :
    - Un timestamp,
    - Le code HTTP,
    - Le type d'erreur,
    - Le message explicatif,
    - et le chemin de la requête

Modalité Pédagogique

Un brief individuel

- Date de lancement: 24/11/2025
- Date de limite : 28/11/2025
- Durée : 5 jour

Modalité d'évaluation

45 min réparties comme suit :

- 05 minutes : Présentation + démonstration des fonctionnalités de l'application
- 05 minutes : Explication du code et de son organisation
- 20 minutes : évaluation des savoirs(Q/A)
- 15 minutes : Mise en situation

Livrables attendus

- Lien github vers le code source complet
- Le diagramme de classes UML (format image)
- Projet JIRA à montrer le jour de l'évaluation
- Un bon README

Critères de réussite

- L'application démarre sans erreur et se connecte correctement à la base
- Les validations, remises, TVA et stock sont correctement gérés
- Les erreurs sont traitées proprement et envoyées sous format JSON cohérent
- L'architecture (Controller-Service-Repository-DTO-Mapper) est claire et propre

Compétences techniques visées

- C1 - N2 : Installer et configurer son environnement de travail en fonction du projet
- C3 - N2 : Développer des composants métier
- C4 - N2 : Contribuer à la gestion d'un projet informatique
- C6 - N2 : Définir l'architecture logicielle d'une application
- C7 - N2 : Concevoir et mettre en place une base de données relationnelle
- C8 - N2 : Développer des composants d'accès aux données SQL et NoSQL
- C9 - N2: Préparer et exécuter les plans de tests d'une application

Compétences transversales visées

- C1 - N2 : Planifier le travail à effectuer individuellement et en équipe afin d'optimiser le travail nécessaire à l'atteinte de l'objectif visé N2
- C6 - N2 : Présenter un travail réalisé en synthétisant ses résultats, sa démarche et en répondant aux questions afin de le restituer au commanditaire
- C8-N2: Interagir dans un contexte professionnel de façon respectueuse et constructive pour favoriser la collaboration

Consignes :

Avant le début de la soutenance :

- Préparer votre IDE (Ouvrir le projet)
- Préparer votre diagramme de classe et JIRA
- Préparer la base de données avec au minimum 5 enregistrements dans chacune des tables de la base de données
- Ouvrir votre dépôt GitHub
- Préparer et ouvrir des slides de présentation simples et claires

Déroulement de la soutenance :

- Introduction : Commencez par une brève présentation du projet, son objectif, son utilité ainsi que les technologies
- Avoir une collection sur postman de toutes les endpoints demandés, ou bien une documentation d'apis via Swagger.
- Expliquez l'architecture globale du projet, notamment la structure des entités, les relations entre elles et les autres couches
- Démonstration avec votre Api Tester : Création , Ajout , modification, calcul automatique ...
- Évaluation des savoirs : Le formateur évaluateur vous posera des questions pour évaluer votre degré de maîtrise des concepts et technologies abordés
- Mise en situation : on vous demandera de coder une méthode ou d'apporter une modification à votre code (une logique métier, introduire une nouvel implémentation d'un service, etc.)