Challenge Technique en Java:

Gestion d'une pizzeria

Description :

Vous gérez une pizzeria dans un food court italien. Les clients commandent leur pizza via une tablette sur leur table. La pizzeria qui livre les commandes le plus rapidement est payée. Vous êtes en concurrence avec toutes les autres pizzerias (autres équipes) du food court.

Le food court met à votre disposition une API dont la documentation est disponible sur Moodle. Pour accéder à la liste des commandes en attente, utilisez le chemin GET /orders. Pour vous identifier à l'API, utilisez votre clé secrète (une par pizzeria, fournie par le professeur) dans l'en-tête Authorization de toutes vos requêtes HTTP.

Pour chaque pizza, vous avez besoin d'ingrédients spécifiques. Vous pouvez consulter la liste des ingrédients disponibles via le chemin GET /market. Les recettes de chaque pizza sont disponibles sur le chemin GET /recipes.

Pour préparer vos pizzas, vous devez disposer des ingrédients nécessaires dans votre espace de stockage. Vous pouvez consulter votre stock via le chemin GET /stock. Il est crucial de gérer votre stock avec soin pour éviter les ruptures et répondre rapidement aux commandes. Notez que votre stock est limité à 20 emplacements et que les ingrédients périmés ne sont plus utilisables.

Pour jeter un ingrédient périmé, utilisez le chemin DELETE /stock/{id} avec l'ID de l'ingrédient. Vérifiez régulièrement les dates de péremption pour éviter le gaspillage.

Pour acheter des ingrédients, utilisez le chemin POST /market/{id}/buy avec l'ID de l'ingrédient. Le coût des ingrédients sera déduit de votre porte-monnaie, et tenez compte du temps de livraison des ingrédients dans votre stock.

Pour débuter la préparation d'une pizza, utilisez le chemin POST /pizzas en fournissant l'ID de la recette et l'ID de la commande. Notez que le temps de préparation de la pizza peut varier selon les recettes.

Une fois la pizza prête, vous pouvez la livrer avec le chemin POST /pizzas/{pizzaId}/deliver. Si la commande a déjà été honorée par une autre pizzeria, vous recevrez une erreur. Votre pizza reste livrable pendant un certain temps à une autre commande avant de devenir périmée.

Les clients du food court peuvent fixer librement le prix de leur commande. Chaque pizzeria décide si elle accepte ou refuse de préparer la commande en fonction du prix proposé.

Devenez la pizzeria la plus rentable en créant un système automatisé de préparation de pizza en Java. Adoptez de bonnes stratégies pour gagner de l'argent.

Exigences :

Rédiger un plan de tests fonctionnels qui couvre au minimum 80 % des fonctionnalités de l'application en détaillant à la fois les cas standards (happy path) et les scénarios d'erreur (inclure des tests qui simulent des entrées invalides afin de vérifier que l'application affiche des messages d'erreur adaptés).

Barème de notation – Livrable de conception

Item	Détails	Points
Diagramme UML	Le diagramme UML initial est correct. Le diagramme UML final est correct et reflète le contenu du code final.	8
Plan de test	Tests à exécuter sur le programme, résultats attendus	4
Résultats des tests	Tests exécutés sur le livrable final, résultats obtenus	4
Collaboration/planning	Planification et répartition du travail entre les membres de l'équipe	4

Barème de notation – exigences Fonctionnelles

Item	Détails	Points
Exigence	Lecture de l'API et authentification	2
Exigence	Commande d'ingrédients avec succès	2
Exigence	Préparation d'une pizza avec succès	2
Exigence	Livraison d'une pizza avec succès	2
Exigence	Automatisation de toute la chaine de logistique	2
Exigence	Étude de la rentabilité et choix stratégiques	2
Respect des exigences	Interaction avec l'utilisateur à la console : affichage des tâches en cours par le programme	2
Défis	- Suppression automatique des ingrédients périmés (2 points) - Equipe la plus rentable avec une stratégie avancée (5 points)	

Barème de notation – exigences Techniques

Voir Moodle

La qualité du rendu de chaque élément exercera une influence sur les points accordés.

- Absence totale de rendu pour un item: 0 points
- Rendu partiel ou bâclé d'un item: moitié des points
- Rendu complet et de qualité de l'item: tous les points

Plagiat entre les équipes : note 0 au module Challenges Techniques

Un seul commit sur git : note 0 au module Challenges Techniques