TP Mini-Projet en Java :

Application de Gestion de Marché Local du Finistère

MarketLocal29

Vidéo du TP disponible sur le drive.

***👩‍💻 Niveau de difficulté : ★★★☆☆**

Contexte :

Le Finistère est une région riche en producteurs locaux offrant une variété de produits frais et artisanaux.
Pour promouvoir l'économie locale et faciliter l'accès à ces produits, vous êtes chargé de développer une application de gestion de marché local.
Cette application permettra aux producteurs de gérer leurs produits et aux clients de passer des commandes facilement.

Description du projet :

Ce projet consiste à créer une application de gestion de marché local pour promouvoir l'économie locale du Finistère. L'application permettra de gérer les producteurs locaux, les produits qu'ils proposent, et les commandes des clients.
L'objectif est de consolider vos connaissances en POO, gestion des exceptions, threads, MVC, DAO, et connexion à une base de données MySQL avec Hibernate.
Vous devrez également versionner votre projet avec git et le déposer sur GitHub.

A - Objectifs du projet :

1. Mettre en place une application de gestion de marché local.
2. Implémenter les fonctionnalités de gestion des producteurs, produits, et commandes.
3. Valider les entrées des utilisateurs et gérer les erreurs.
4. Utiliser les threads pour simuler des commandes asynchrones.
5. Appliquer le modèle MVC pour structurer l'application.
6. Accéder à une base de données MySQL via JDBC et Hibernate.

B - Compétences attendues :

1. **Programmation Orientée Objet (POO)** : Capacité à utiliser l'héritage, l'encapsulation, l'abstraction et le polymorphisme pour modéliser des objets du monde réel.
2. **Gestion des Exceptions et Threads** : Maîtrise de la gestion des erreurs et de la programmation multithread.

3. **Modèle-Vue-Contrôleur (MVC)** : Compréhension de la séparation des préoccupations et capacité à structurer l'application en utilisant le modèle MVC.
4. **Accès à la Base de Données** : Utilisation de JDBC et Hibernate pour interagir avec une base de données MySQL.
5. **Versionnement avec Git** : Utilisation de git pour versionner le code et gestion du dépôt sur GitHub.

C - Compétences développées ou renforcées :

1. **Développement Back-End** : Acquisition de compétences en développement back-end pour créer des applications Java robustes et scalables.
2. **Architecture Logicielle** : Renforcement des compétences en architecture logicielle, notamment en utilisant le modèle MVC.
3. **Accès aux Données** : Développement de compétences pour interagir avec des bases de données relationnelles via JDBC et Hibernate.
4. **Gestion des Threads** : Compréhension et utilisation des threads pour la programmation concurrente.
5. **Collaboration et Versionnement** : Pratique de la collaboration et de la gestion de version avec git et GitHub.

D - Structure du Projet :

Fichiers et Dossiers :

`src/` **: Dossier contenant les sources du projet.**

- `model/` **: Dossier contenant les classes métiers.**

    - `Producer.java` : **Classe représentant un producteur.**
    - `Product.java` : **Classe représentant un produit.**
    - `Order.java` : **Classe représentant une commande.**
    - `Market.java` : **Classe principale gérant le marché.**
- `dao/` : **Dossier contenant les classes d'accès aux données.**
    - `ProducerDAO.java` : **DAO pour les producteurs.**
    - `ProductDAO.java` : **DAO pour les produits.**
    - `OrderDAO.java` : **DAO pour les commandes.**
- `controller/` : **Dossier contenant les contrôleurs.**
    - `MarketController.java` : **Contrôleur pour les interactions.**
- `view/` : **Dossier contenant les vues (console pour ce TP).**
    - `Main.java` : **Classe principale lançant l'application.**
- `utils/` : **Dossier contenant les utilitaires.**
    - `Exceptions.java` : **Fichier contenant les exceptions personnalisées.**

  `resources/` : **Dossier contenant les ressources (fichiers de configuration, etc.).**

- `hibernate.cfg.xml` : **Fichier de configuration pour Hibernate.**

E - Énoncé Technique :

Le projet consiste à créer une application de gestion de marché local en Java. Voici les principales étapes et fonctionnalités du projet :

1. Modélisation UML :

- **Diagramme de Cas d'Utilisation** : Identifiez les cas d'utilisation principaux tels que Ajouter un producteur, Supprimer un

producteur, Ajouter un produit, Passer une commande, etc.
Créez un diagramme de cas d'utilisation avec UMLet.

- **Diagramme de Classes** : Identifiez les classes principales et leurs relations.
  Créez un diagramme de classes avec UMLet.

2. Implémentation :

- **Classes Métiers**
- **MVC (Model-View-Controller) :**
  - **Controller** : Gérer les interactions.
  - **View** : Interface utilisateur en console (pour ce TP).
  - **Model** : Classes métiers et DAO.
- **Threading :** Utiliser les threads pour simuler des commandes asynchrones.

F - Livrables Attendus :

1. **Code Source** : Fichiers Java, configuration Hibernate, scripts SQL pour créer la base de données.
2. **Documentation** : Explication des différentes parties du code, choix de conception, problèmes rencontrés.
3. **Diagrammes UML** : Cas d'utilisation et diagrammes de classes.
4. **Capture(s) d'écran ou vidéo(s)** : Montrant l'application en action.
5. **Dépôt GitHub** : Projet versionné et déposé sur GitHub.

G - Critères d'Évaluation :

- ☐ **Fonctionnalité de l'application** : Gestion des producteurs, produits, et commandes.
- ☐ **Qualité du code** : Structure, modularité, et lisibilité.
- ☐ **Design de l'interface utilisateur** : Esthétique et facilité d'utilisation.
- ☐ **Réactivité de l'application** : Gestion asynchrone des commandes.
- ☐ **Expérience utilisateur** : Clarté des instructions et feedback après les actions.
- ☐ **Respect des bonnes pratiques** : Développement logiciel et versionnement avec git.

Date de Rendu :

👩‍💻 **Niveau de difficulté : ★★★☆☆**

Ce TP a un niveau de difficulté de 3 étoiles sur 5. Voici pourquoi :

- **Implémentation du MVC** : La séparation des préoccupations en utilisant MVC demande une bonne compréhension de l'architecture logicielle.
- **Accès aux Données** : La connexion à une base de données via JDBC et Hibernate nécessite une bonne connaissance des frameworks.
- **Gestion des Threads** : La programmation concurrente avec des threads peut être complexe.
- **Modélisation UML** : La création de diagrammes UML demande une réflexion approfondie sur la modélisation des données.
- **Versionnement avec Git** : La gestion de versions avec git et la collaboration via GitHub sont essentielles pour les projets en équipe.

Dans l'ensemble, ce projet offre une opportunité d'apprentissage intéressante pour les étudiants ou les développeurs qui cherchent à approfondir leurs compétences en développement Java et en gestion de projets logiciels.