



Système de Gestion d'Inventaire avec Java RMI

Projet de Système Distribué

École : SupMTI
Cycle d'Ingénierie 2A GI

Binôme :
CHIKHI Ayyoub & ELJAMGHILI Omar

Professeur :
Mohammed OUANAN

Version 1.0

30 décembre 2024

Table des matières

1	Introduction	2
1.1	Contexte du Projet	2
1.2	Objectifs	2
2	Architecture du Système	3
2.1	Vue d'Ensemble	3
2.2	Diagrammes UML	3
2.2.1	Diagramme de Classes	3
2.2.2	Diagramme de Cas d'Utilisation	4
2.3	Technologies Utilisées	4
3	Implémentation	5
3.1	Structure du Projet	5
3.2	Base de Données	5
3.2.1	Schéma de la Base de Données	5
3.3	Interface Utilisateur	5
3.3.1	Écran de Connexion	5
3.3.2	Interface Principale	6
3.3.3	Liste des Produits	7
3.3.4	Gestion des Produits	7
3.4	Système de Logging	10
4	Fonctionnalités	12
4.1	Gestion des Produits	12
4.2	Authentification	12
5	Tests et Validation	13
5.1	Tests Fonctionnels	13
5.2	Démarrage du Serveur	13
6	Conclusion et Perspectives	14
6.1	Bilan	14
6.2	Améliorations Futures	14
6.3	Code Source	14

Chapitre 1

Introduction

1.1 Contexte du Projet

Le projet de système de gestion d'inventaire est une application distribuée qui illustre l'utilisation de Java RMI (Remote Method Invocation) dans un contexte pratique. Cette application permet la gestion complète d'un inventaire de produits avec une interface graphique moderne développée en JavaFX, offrant une expérience utilisateur fluide et intuitive.

Le système est conçu pour répondre aux besoins quotidiens des entreprises en matière de gestion de stock, permettant un suivi en temps réel des produits, des modifications d'inventaire, et la génération de rapports détaillés.

1.2 Objectifs

Les objectifs principaux du projet sont :

- Développer une application client-serveur avec Java RMI permettant une communication efficace entre les différents composants du système
- Implémenter une interface utilisateur intuitive avec JavaFX pour faciliter la gestion des produits
- Gérer la persistance des données avec MySQL pour assurer la durabilité des informations
- Assurer la sécurité avec un système d'authentification robuste
- Maintenir un historique des opérations via un système de logging complet

Chapitre 2

Architecture du Système

2.1 Vue d'Ensemble

L'architecture du système repose sur le modèle client-serveur utilisant Java RMI. Cette architecture se compose de trois couches principales :

- La couche présentation (Client JavaFX)
- La couche métier (Serveur RMI)
- La couche données (Base de données MySQL)

2.2 Diagrammes UML

2.2.1 Diagramme de Classes

Le diagramme de classes ci-dessous illustre la structure statique du système, montrant les relations entre les différentes classes et interfaces.

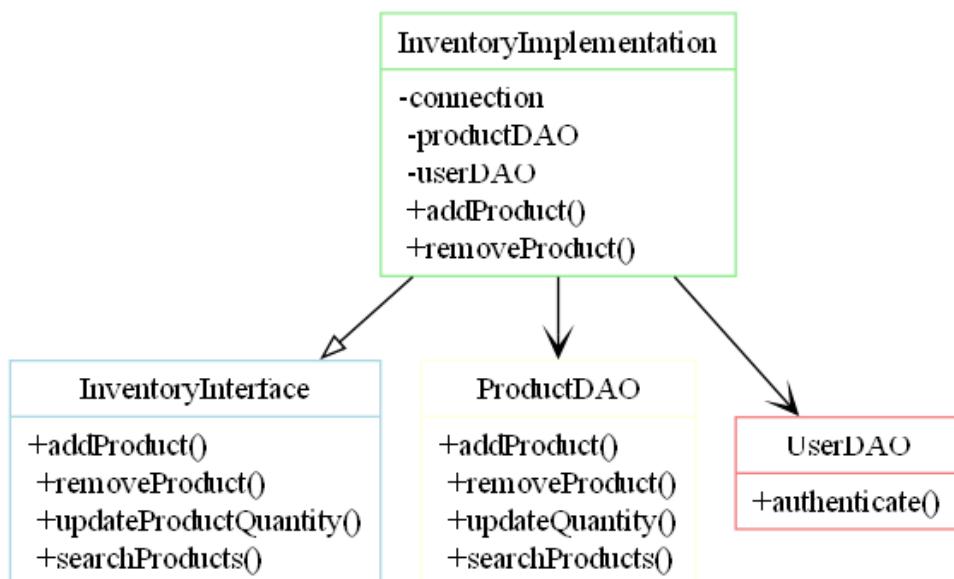


FIGURE 2.1 – Diagramme de classes du système

2.2.2 Diagramme de Cas d'Utilisation

Ce diagramme présente les différentes interactions possibles entre les utilisateurs et le système.

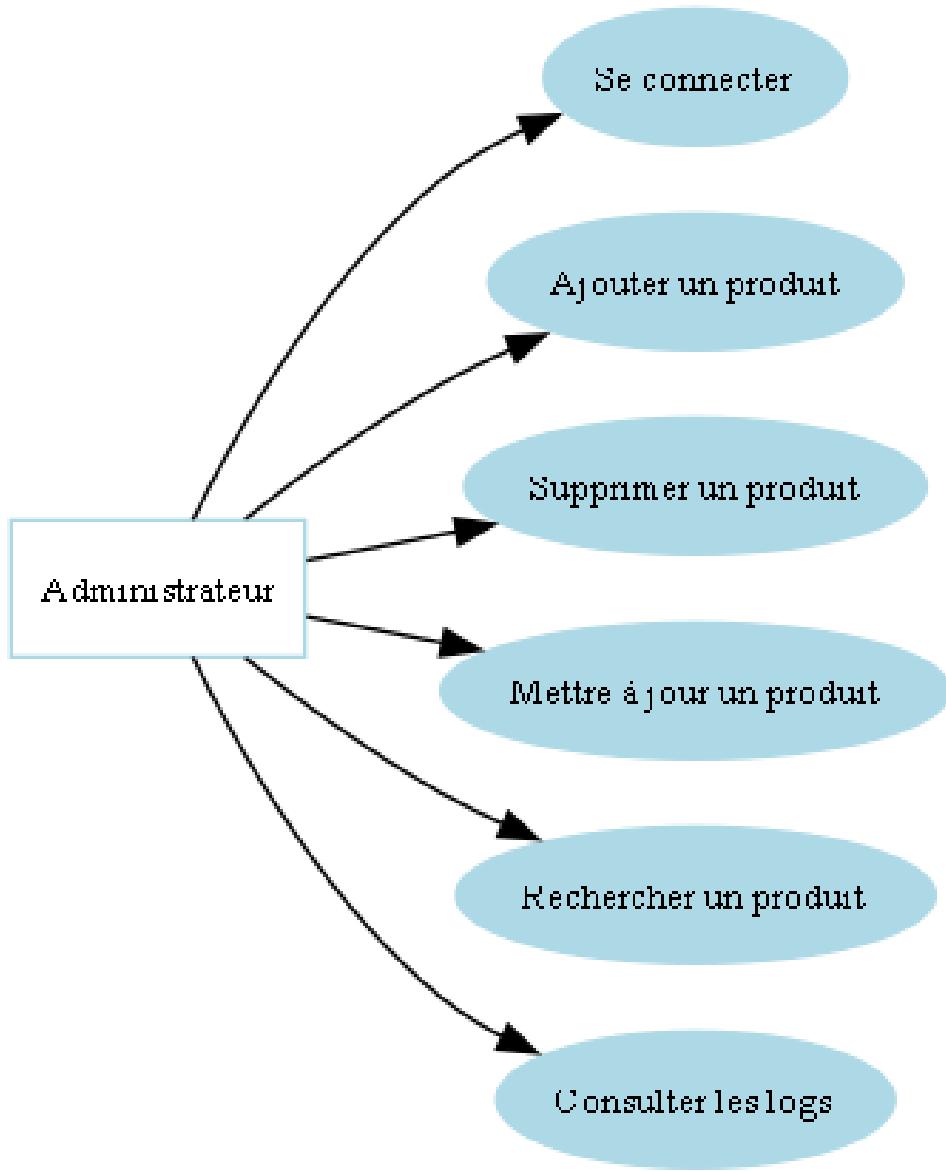


FIGURE 2.2 – Diagramme de cas d'utilisation

2.3 Technologies Utilisées

- **Java 21.0.5 LTS** : Plateforme de développement principale
- **JavaFX SDK 23.0.1** : Framework pour l'interface graphique
- **MySQL** : Système de gestion de base de données relationnelle
- **Java RMI** : Technologie pour la communication distribuée

Chapitre 3

Implémentation

3.1 Structure du Projet

La structure du projet est organisée de manière modulaire pour faciliter la maintenance et l'évolution du code. Voici une vue d'ensemble de l'organisation des fichiers et dossiers :

FIGURE 3.1 – Structure du projet

3.2 Base de Données

3.2.1 Schéma de la Base de Données

La base de données utilise deux tables principales pour stocker les informations des produits et des utilisateurs :

```
1 CREATE TABLE products (
2     id INT AUTO_INCREMENT PRIMARY KEY,
3     name VARCHAR(100) NOT NULL,
4     category VARCHAR(50) NOT NULL,
5     quantity INT NOT NULL,
6     price DOUBLE NOT NULL
7 );
8
9 CREATE TABLE users (
10    id INT AUTO_INCREMENT PRIMARY KEY,
11    username VARCHAR(50) UNIQUE NOT NULL,
12    password VARCHAR(255) NOT NULL
13 );
```

3.3 Interface Utilisateur

3.3.1 Écran de Connexion

L'écran de connexion est la première interface que l'utilisateur rencontre. Il permet une authentification sécurisée avec un nom d'utilisateur et un mot de passe.

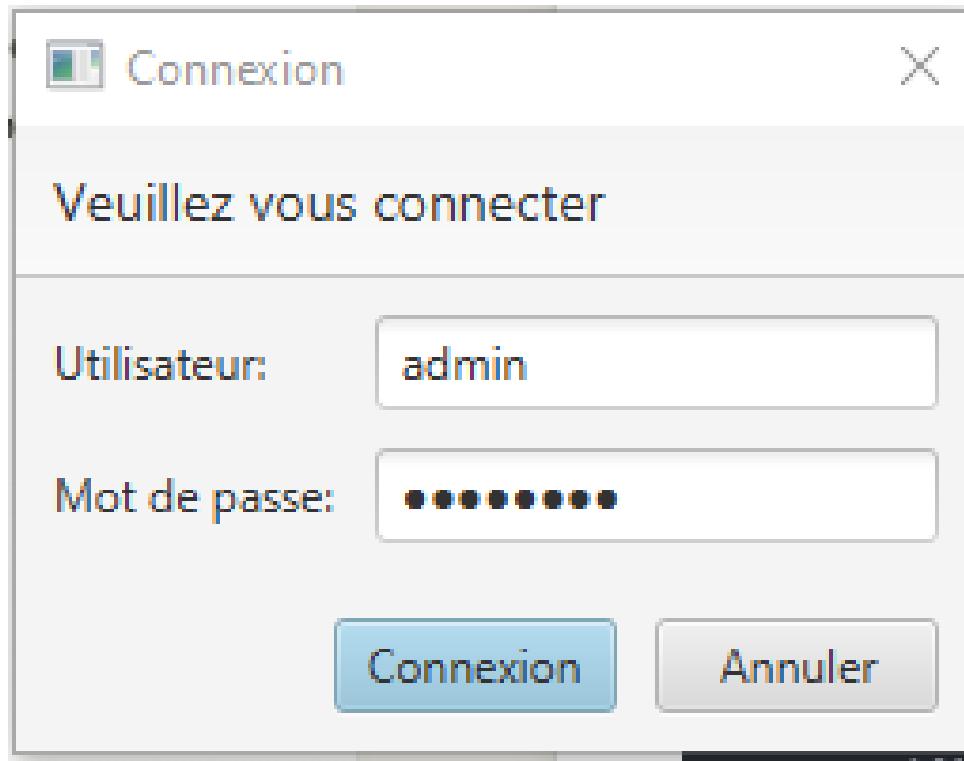


FIGURE 3.2 – Interface de connexion

3.3.2 Interface Principale

L'interface principale offre une vue complète de l'inventaire et permet d'accéder à toutes les fonctionnalités du système.

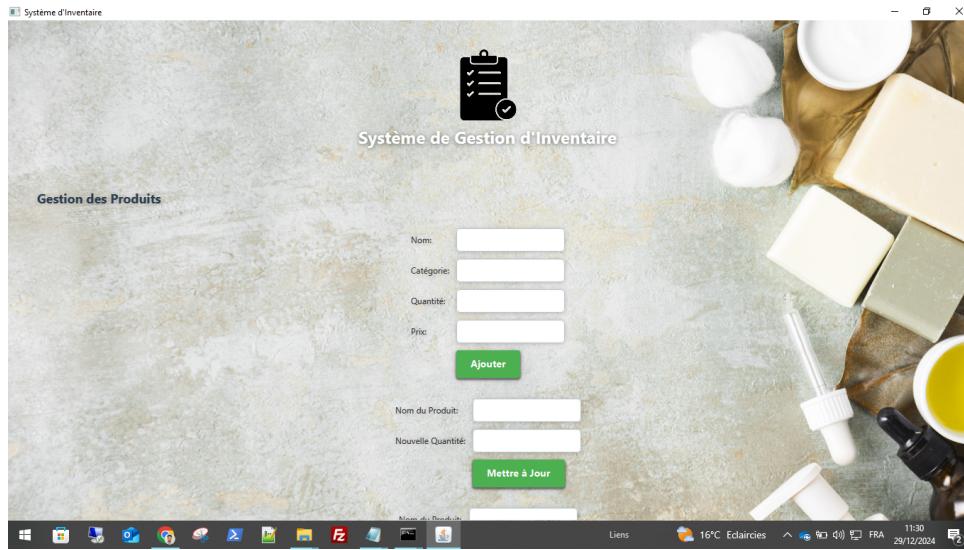


FIGURE 3.3 – Interface principale - Vue générale

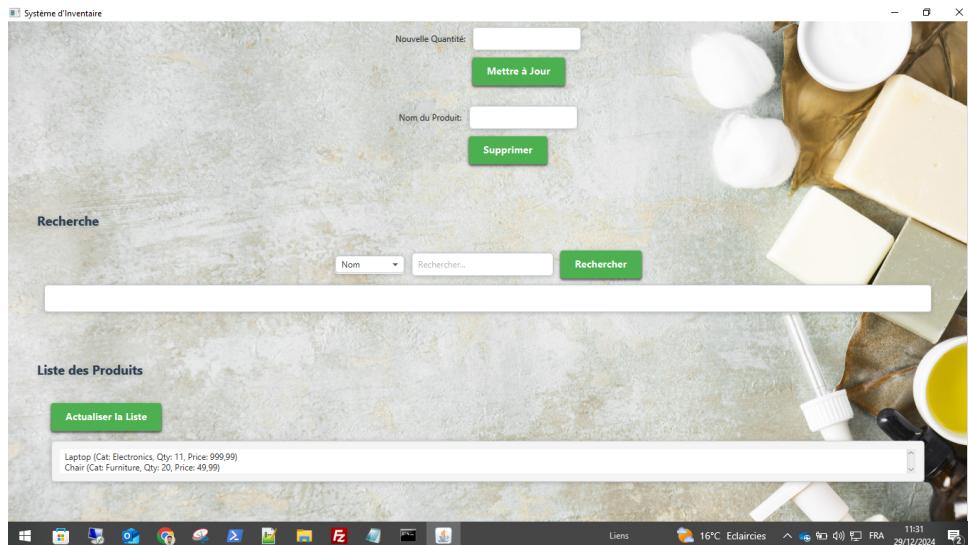


FIGURE 3.4 – Interface principale - Gestion des produits

3.3.3 Liste des Produits

L'interface principale présente une vue complète de tous les produits disponibles avec une fonction d'actualisation en temps réel.

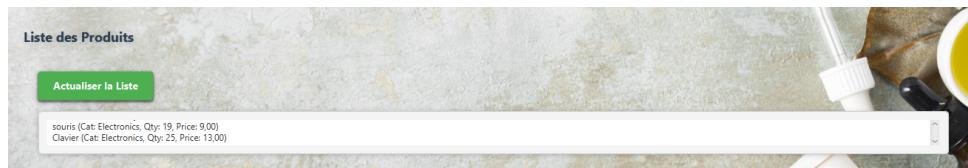


FIGURE 3.5 – Liste complète des produits

3.3.4 Gestion des Produits

Ajout de Produit

L'interface d'ajout permet de saisir les informations d'un nouveau produit :

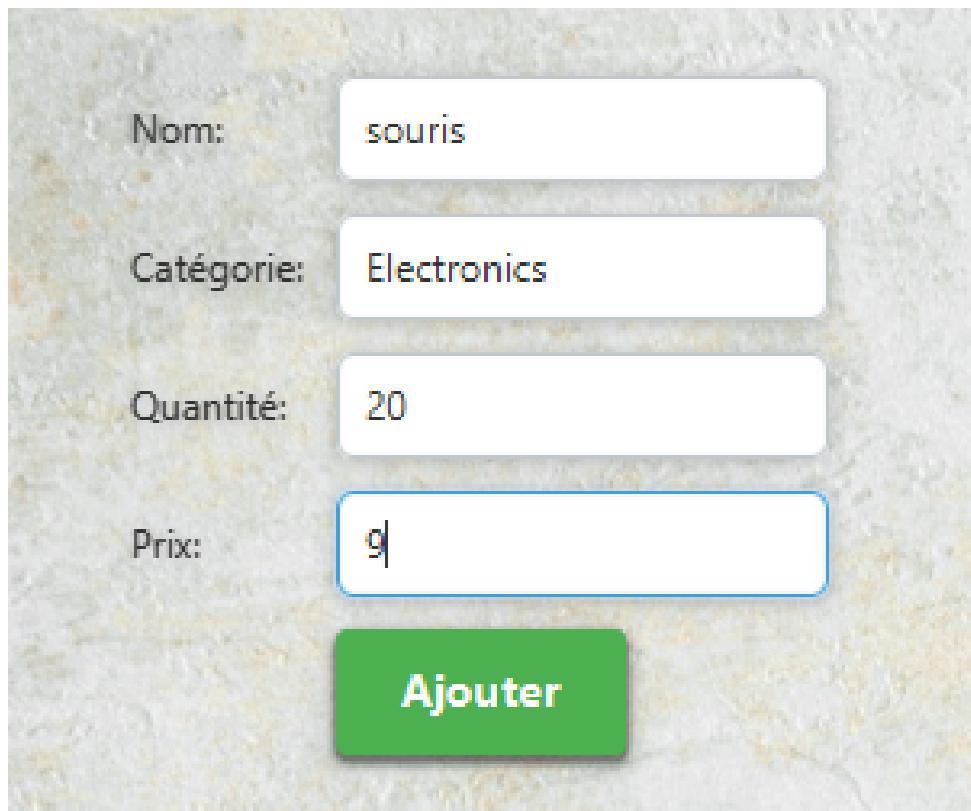


FIGURE 3.6 – Interface d'ajout de produit

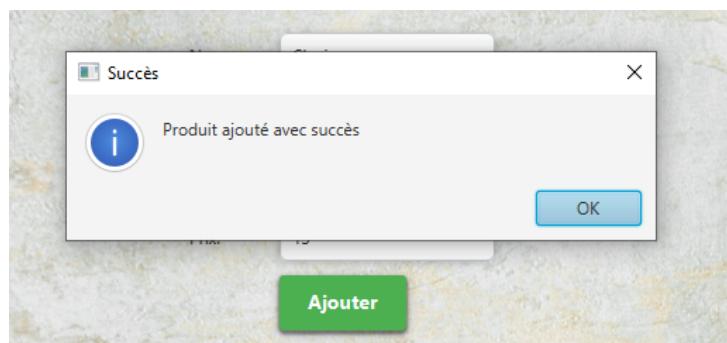


FIGURE 3.7 – Confirmation d'ajout réussi

Modification de Quantité

Le système permet la mise à jour des quantités en stock :



FIGURE 3.8 – Interface de modification de quantité

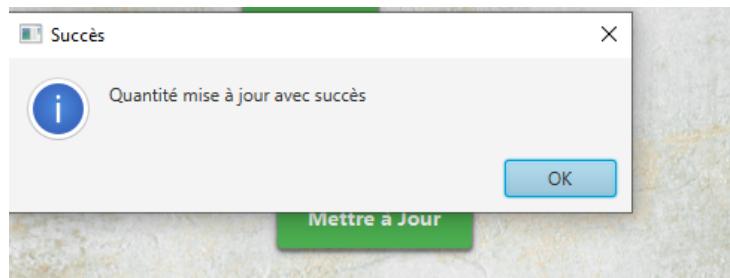


FIGURE 3.9 – Confirmation de modification réussie

Recherche de Produits

Le système offre trois types de recherche :

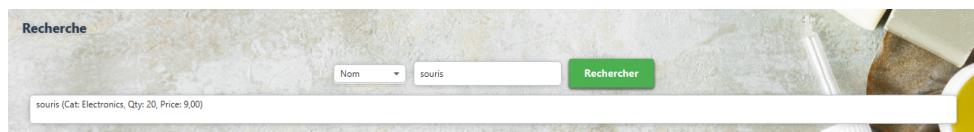


FIGURE 3.10 – Recherche par nom

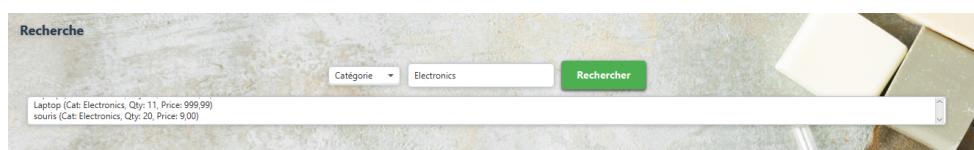


FIGURE 3.11 – Recherche par catégorie



FIGURE 3.12 – Recherche par niveau de stock

Suppression de Produit

La suppression se fait en plusieurs étapes pour éviter les erreurs :

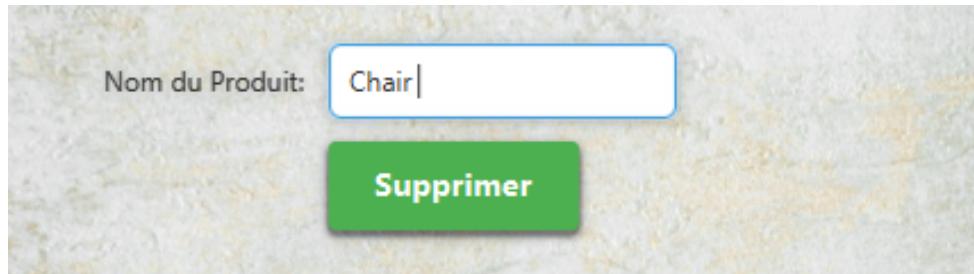


FIGURE 3.13 – Interface de suppression

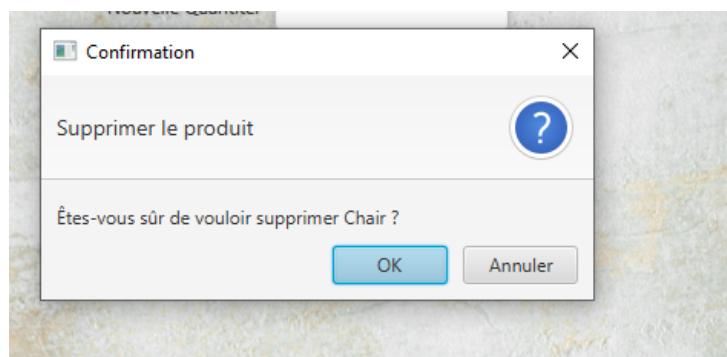


FIGURE 3.14 – Demande de confirmation

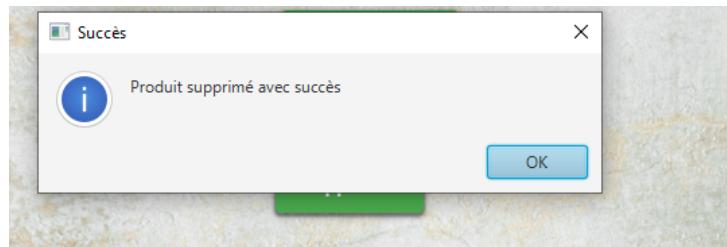


FIGURE 3.15 – Confirmation de suppression réussie

3.4 Système de Logging

Le système de logging enregistre toutes les opérations importantes pour assurer la traçabilité :

```

1  @Override
2  public void addProduct(String name, String category,
3      int quantity, double price)
4      throws RemoteException {
5          try {
6              productDAO.addProduct(name, category, quantity, price);
7              Logger.log("Produit ajout : " + name +

```

```
8     ", Cat gorie : " + category +
9     ", Quantit : " + quantity +
10    ", Prix : " + price);
11 } catch (SQLException e) {
12     throw new RemoteException(
13         "Erreur lors de l'ajout du produit", e);
14 }
15 }
```

Chapitre 4

Fonctionnalités

4.1 Gestion des Produits

Le système offre une gestion complète des produits avec les opérations CRUD suivantes :

- **Ajout de nouveaux produits** : Interface intuitive avec validation des données
- **Modification des quantités** : Mise à jour en temps réel des stocks
- **Suppression de produits** : Processus sécurisé avec confirmation
- **Recherche multicritères** : Recherche par nom, catégorie ou niveau de stock

4.2 Authentification

Le système d'authentification assure la sécurité des accès :

- Gestion des credentials avec cryptage des mots de passe
- Sessions utilisateur sécurisées
- Validation des données de connexion

Chapitre 5

Tests et Validation

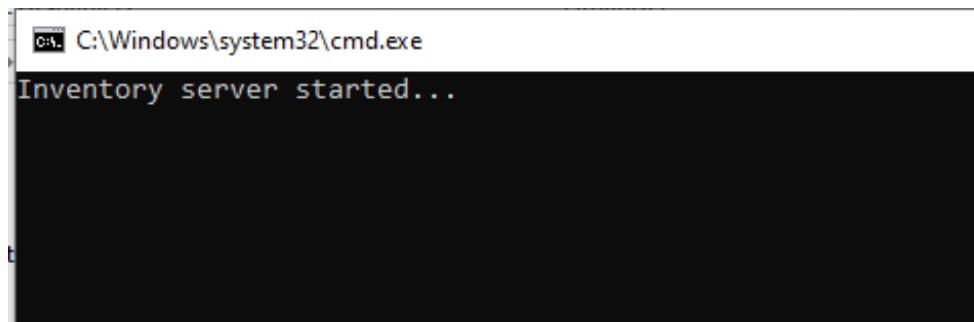
5.1 Tests Fonctionnels

Les tests suivants ont été effectués pour valider le système :

- Tests d'authentification et de sécurité
- Validation des opérations CRUD sur les produits
- Tests de performance du système de recherche
- Vérification du système de logging

5.2 Démarrage du Serveur

Le serveur affiche des informations de diagnostic lors du démarrage :



A screenshot of a Windows Command Prompt window titled 'C:\Windows\system32\cmd.exe'. The window contains the text 'Inventory server started...'. The background of the window is black, and the text is white.

FIGURE 5.1 – Console du serveur en exécution

Chapitre 6

Conclusion et Perspectives

6.1 Bilan

Le projet a atteint ses objectifs principaux :

- Architecture distribuée robuste et efficace
- Interface utilisateur moderne et intuitive
- Système de gestion des données sécurisé
- Logging complet des opérations

6.2 Améliorations Futures

Les perspectives d'évolution incluent :

- Ajout de tableaux de bord avec graphiques statistiques
- Implémentation d'un système de notifications en temps réel
- Export des données vers différents formats (PDF, Excel)
- Gestion avancée des droits utilisateurs
- Interface responsive pour supports mobiles

6.3 Code Source

Le code source complet du projet est disponible publiquement sur GitHub :

<https://github.com/ayyoubchikhi/InventoryRMIProject>

Le dépôt contient :

- Code source complet de l'application
- Scripts de base de données
- Fichiers de configuration
- Documentation détaillée d'installation