



# Client Serveur



## Système de Gestion de Stock et de Facturation

### Description

Ce projet a pour objectif de développer un système informatique complet pour la **gestion du stock de marchandises** et la **préparation des factures** des ventes effectuées par un vendeur.

Le système est conçu en **architecture client-serveur** et utilise le protocole de communication **RMI (Remote Method Invocation)** pour permettre l'interaction entre les clients et le serveur.

### Fonctionnalités

-  Gestion des produits en stock :
  - Ajout, modification et suppression de produits
  - Consultation du stock disponible
-  Gestion des ventes :
  - Création de factures clients
  - Mise à jour automatique du stock lors des ventes

- 📁 Persistance des données (stock et factures)
  - 🔄 Communication client-serveur via RMI
- 

## Architecture

- **Backend (Serveur)** : Java avec RMI, gestion des données persistantes
- **Frontend (Client)** : Java RMI Client (ou possibilité de GUI avec JavaFX/Swing)
- **Base de données** : Fichier local / SGBD (au choix selon l'implémentation)
- **Communication** : Protocole RMI

### ##Prérequis Avant lancement

- \*Créer une base de données JDBC sur le port 3306 avec un outil comme XAMP
- \*Après la connexion a JDBC créer les deux bases à l'aide des fichiers SQL fournis dans le ZIP

## Lancement de l'application

- \*Lancer le fichier SiegeRMI pour lancer le serveur du siège
- \*Lancer le fichier MagasinRMI pour lancer le serveur du magasin
- \*Lancer le client avec ClientApp

Vous devriez ensuite voir l'interface du client

BricoMerlin - Client

Articles + Ajouter Stock Facturation Analyse

Recherche d'articles

Référence :  Rechercher par référence

Famille :  Rechercher par famille

Panier / Commande

Réf :  Qté :  Ajouter au panier Supprimer du panier

Total : 0.00 €

Objectif :

Permet à l'utilisateur de :

rechercher un article par sa référence ou sa famille,

consulter les détails de l'article (nom, prix, stock, famille),

ajouter des articles à un panier,

gérer le panier (supprimer, totaliser),

valider une commande (envoi au serveur + création ticket JSON).

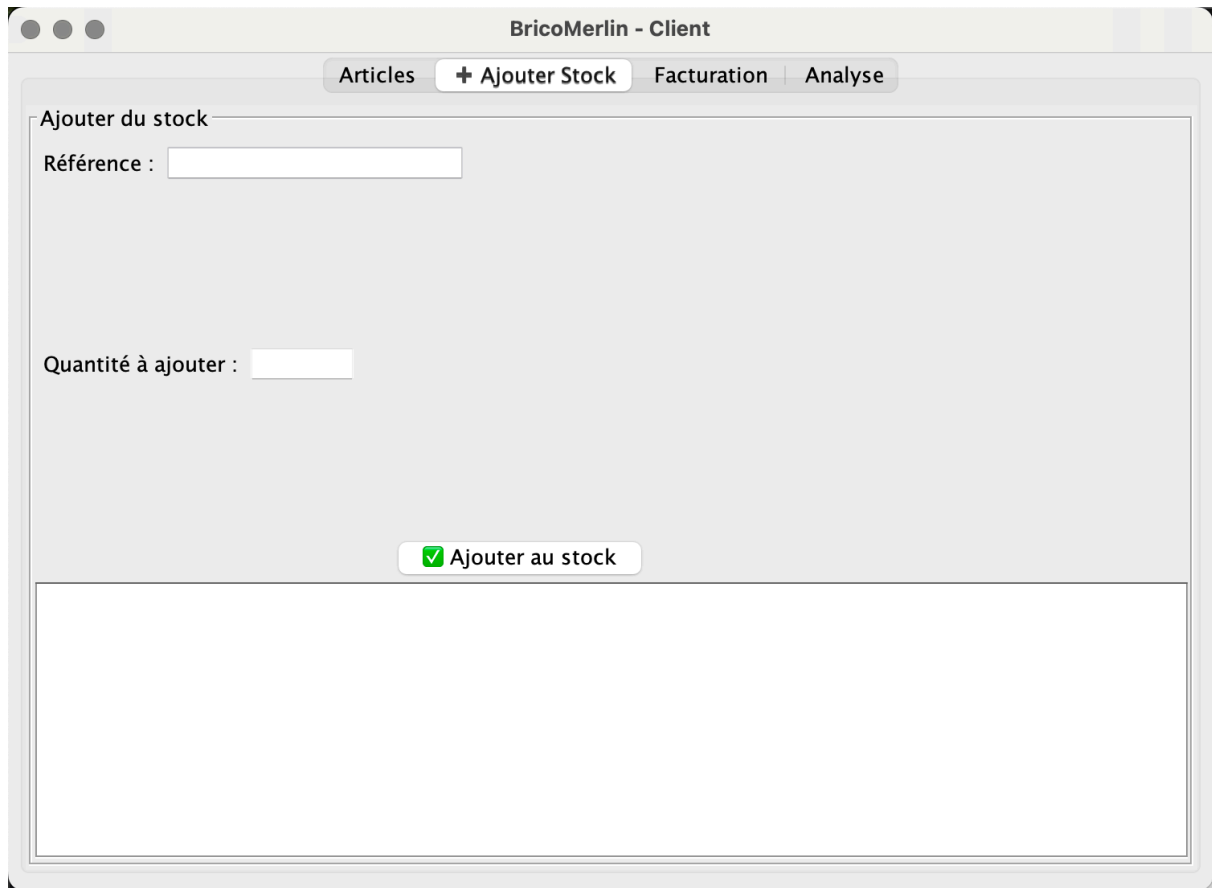
Fonctionnalités principales :

Recherche par référence (ex : "Pot de peinture").

Recherche par famille (ex : "Peinture").

Création d'un panier temporaire avec quantité, nom, prix total.

Envoi de la commande au serveur avec mise à jour du stock.



The screenshot shows a web application window titled "BricoMerlin - Client". It features a tabbed interface with four tabs: "Articles", "+ Ajouter Stock" (which is the active tab), "Facturation", and "Analyse". The "Ajouter Stock" tab contains a form with the following elements:

- A section header "Ajouter du stock".
- A label "Référence :" followed by a text input field.
- A label "Quantité à ajouter :" followed by a text input field.
- A green checkmark icon and the text "Ajouter au stock" at the bottom of the form.
- A large empty rectangular box at the bottom of the tab.

Onglet 2 : Ajouter Stock

Objectif :

Permet d'augmenter manuellement le stock d'un article déjà existant en base de données.

Fonctionnalités :

L'utilisateur saisit une référence d'article existante.

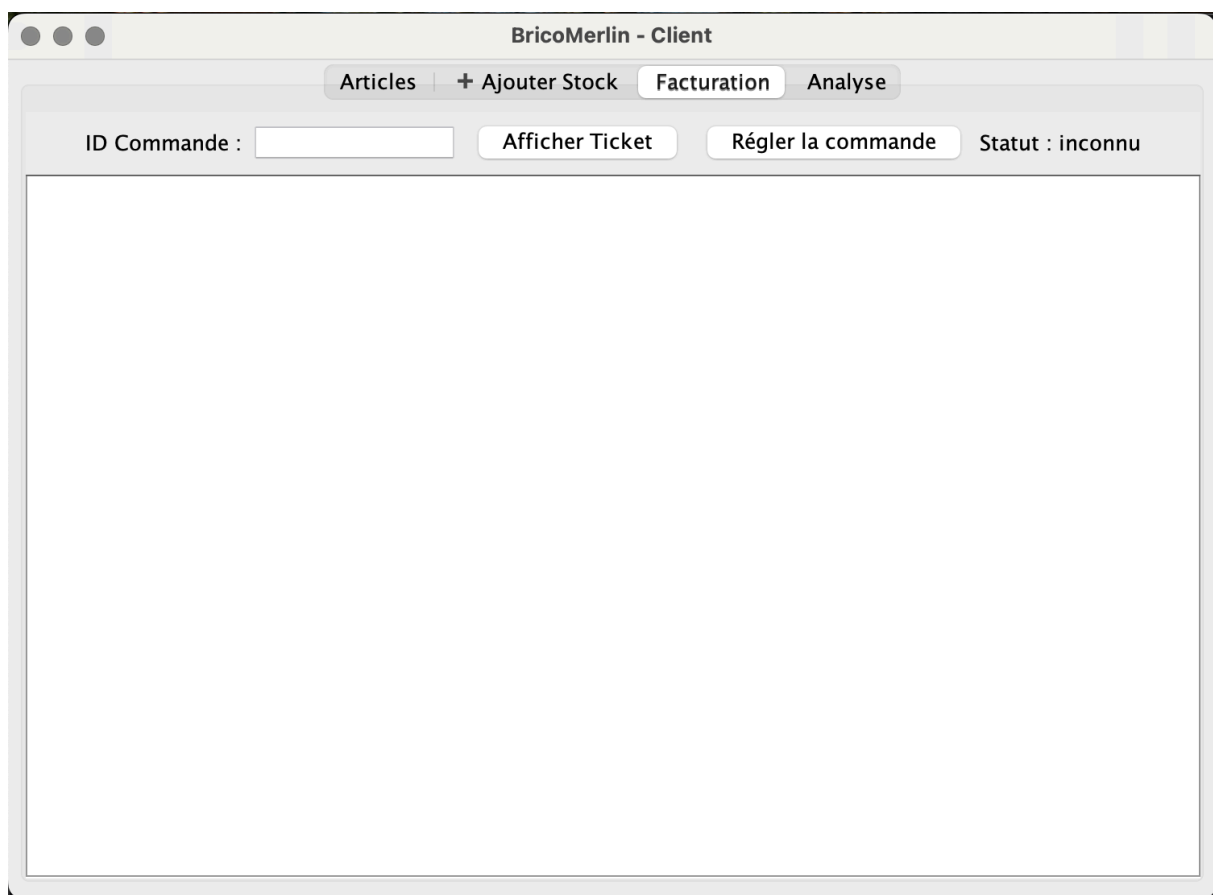
Il indique la quantité à ajouter au stock.

En cliquant sur Ajouter au stock, l'application :

appelle le serveur via RMI (ajouterStockProduit()),

met à jour la base de données,

affiche un message de succès ou d'erreur dans la zone de texte.



Onglet 3 : Facturation

Objectif :

Permet à l'utilisateur de :

consulter le ticket de caisse d'une commande existante,

vérifier le statut de paiement,

effectuer le règlement d'une commande encore "En Attente".

Fonctionnalités :

L'utilisateur saisit un ID de commande.

En cliquant sur Afficher Ticket :

le ticket est chargé depuis un fichier JSON (dans le dossier factures/),

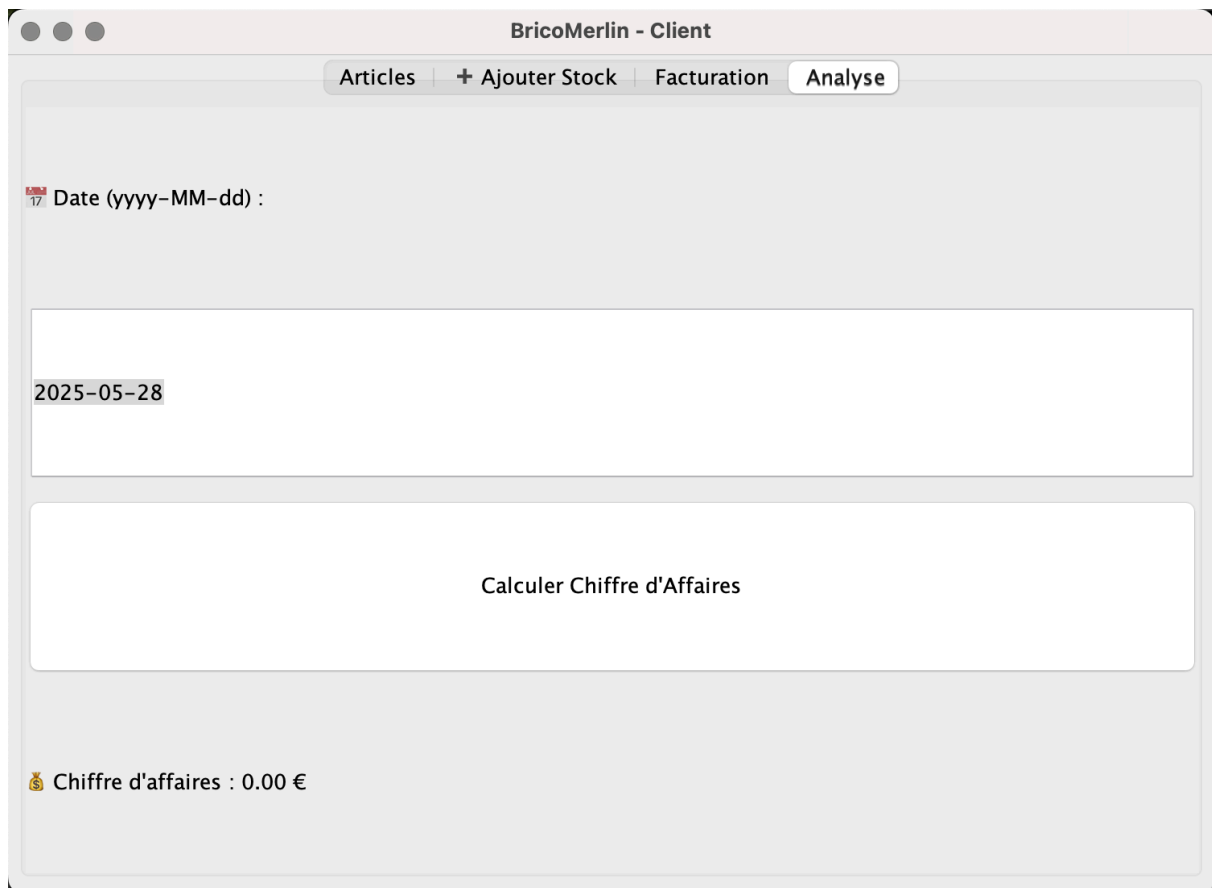
le détail complet s'affiche : articles, quantités, prix, total.

le statut de paiement est récupéré via le serveur (getStatutPaiement()).

En cliquant sur Régler la commande :

le statut est mis à jour dans la base (statut\_paiement = 'Payée'),

le message de succès ou d'échec est affiché.



#### Onglet 4 : Analyse

##### Objectif :

Permet à l'utilisateur d'analyser l'activité du magasin en calculant le chiffre d'affaires (CA) généré à une date précise.

##### Fonctionnalités :

L'utilisateur saisit une date au format yyyy-MM-dd.

En cliquant sur Calculer Chiffre d'Affaires :

l'application appelle le serveur via RMI (`calculerChiffreAffaires(date)`),

le serveur interroge la base de données commandes et additionne le champ `total_prix` de toutes les commandes passées ce jour-là,

le CA est affiché dans l'interface.