




# Architecture : Client — Serveur

— Il vaut mieux accéder à la doc via ce lien :

 [Architecture : Client — Serveur](#)

— Auteurs : MASTER 1 — MIAGE UPJV (2024-2025)

- Anisse Gaia Imerzoukene
- Abdoul-Hakim Alio Alassane

— Sujet :

[Projet Client-Serveur 2024-2025 \(Alternants\).pdf](#)

[clent-servers.schemas.drawio](#)

I] [Introduction](#)

II] [Solution — theory](#)

1. [Architecture](#)

III] [Solution — how we built it](#)

1. [Schema](#)

## I] Introduction

— Features

### 1. Consultation du stock d'un article

- Rechercher un article par sa référence

- Afficher ses informations (quantité en stock, prix unitaire, etc.)

Recherche de Produits

ID Produit :

ID Famille :

Rechercher Produit

Rechercher par Famille

ID	ID Famille	Nom	Prix	Quantité
1	1	Smartphone	599.99	20

## 2. Recherche d'articles par famille

- Rechercher des articles appartenant à une famille donnée
- Retourner uniquement les références dont le stock n'est pas nul

Recherche de Produits

ID Produit :

ID Famille :

Rechercher Produit

Rechercher par Famille

ID	ID Famille	Nom	Prix	Quantité
9	2	Novel	12.99	100
10	2	Cookbook	24.99	40
11	2	Science Textbook	39.99	30
12	2	History Book	18.99	20
13	2	Fantasy Book	15.5	60
14	2	Self-Help Guide	13.49	50
15	2	Children Storybook	9.99	70

Message

7 produits trouvés !

OK

## 3. Achat d'un article

- Vérifier la disponibilité de l'article en stock
- Déduire la quantité achetée du stock

Système de Gestion de Stock

Produits Créer Produit Créer Famille Factures Rapports **Commandes**

Ajouter un produit à la commande

ID Produit :  Quantité :

Commande courante

ID Produit	Nom	Quantité	Prix unitaire	Total
1	Smartphone	4	599.99	2399.96
3	Bluetooth Speaker	2	49.99	99.98
15	Children Storybook	1	9.99	9.99

Résumé de la commande

Articles dans la commande : 3  
Montant total : 2509.93€

#### 4. Paiement d'une facture

- Permettre à un client de payer une facture

Système de Gestion de Stock

Produits Créer Produit Créer Famille Factures Rapports **Commandes**


Ajouter un produit à la commande

ID Produit :  Quantité :

Commande courante

ID Produit	Nom	Quantité	Prix unitaire	Total
1	Smartphone	4	599.99	2399.96
3	Bluetooth Speaker	2	49.99	99.98
15	Children Storybook	1	9.99	9.99

Message

 Commande confirmée avec succès !

Résumé de la commande

Articles dans la commande : 3  
Montant total : 2509.93€

#### 5. Consultation d'une facture

- Afficher une facture (ticket de caisse)

Système de Gestion de Stock

Produits Créer Produit Créer Famille Factures Rapports Commandes

Recherche de Facture

ID Facture : 1 Rechercher Facture

Détails de la facture :

ID Facture : 1  
Date : 2025-06-30  
Montant total : 2509.93€  
Méthode de paiement : Card  
Statut : Payé

Payer Facture

ID Facture : 1 Payer Facture

## 6. Calcul du chiffre d'affaires à une date donnée

- Calculer le chiffre d'affaires en fonction des factures enregistrées pour cette date

Système de Gestion de Stock

Produits Créer Produit Créer Famille Factures Rapports Commandes

Calcul des Revenus

Date (aaaa-MM-jj) : 2025-06-30 Calculer Revenus

Total des Revenus : 2509,93€

Sauvegarder Factures

Sauvegarder Factures sur le Serveur

## 7. Ajout d'un produit en stock

- Ajouter un certain nombre d'exemplaires d'un produit existant dans le catalogue

#### **8. Mise à jour des prix**

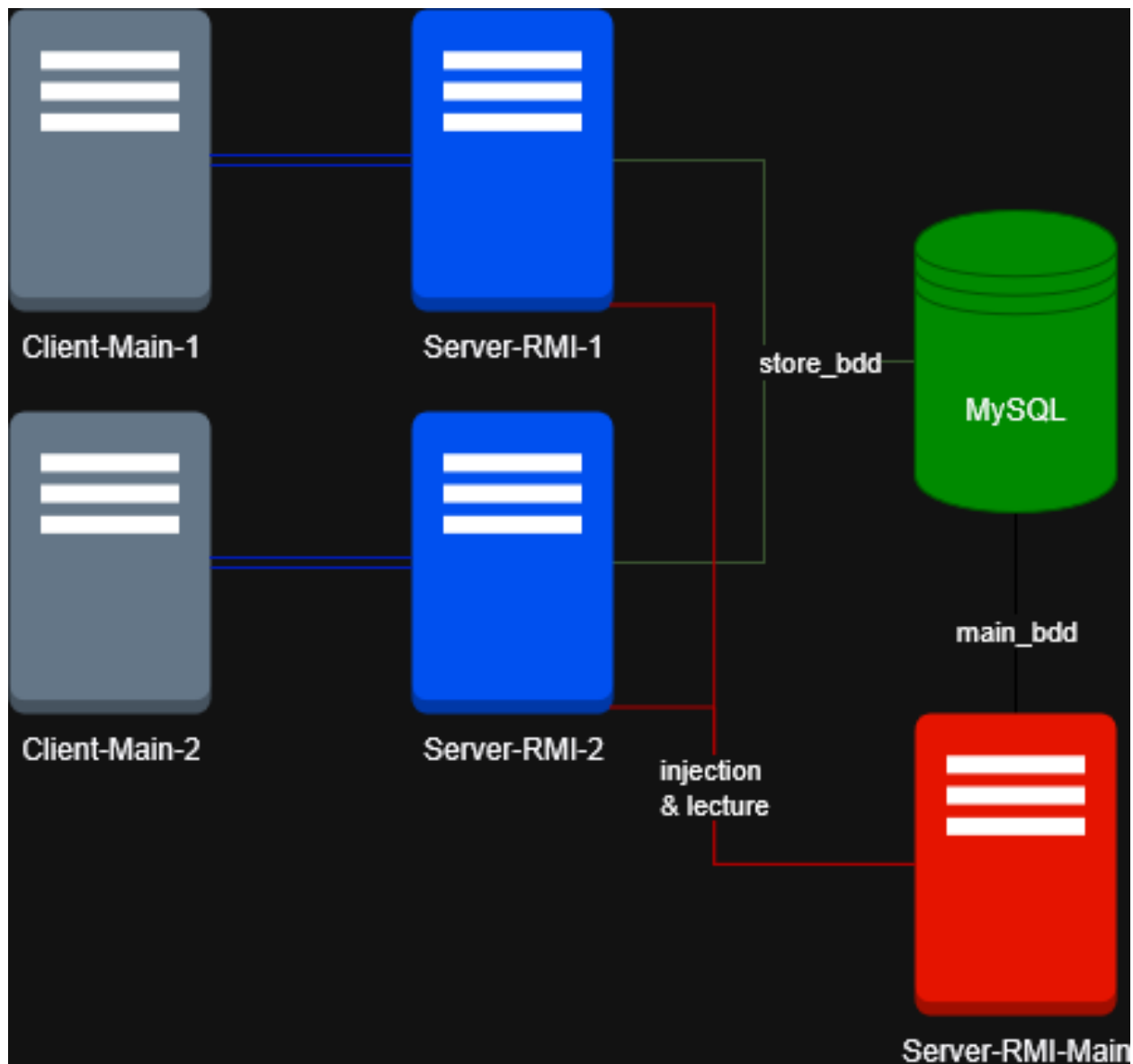
- Mettre à jour les prix des articles chaque matin via le serveur du siège

#### **9. Sauvegarde des factures**

- Enregistrer toutes les factures sur le serveur du siège chaque soir

## **II] Solution — theory**

### **1. Architecture**



Hypothétiquement, nous pourrions lancer plusieurs servers RMI pour chacun des magasins et un server RMI main. L'architecture ainsi que le code du server main sont construits pour accueillir un nombre  $n$  de magasins de manière dynamique.

— Application magasin :

- Appel au RMI "ServerMain" (StockService) pour réaliser des opérations sur la database "store\_bdd". Ces opérations sont applicatives.

— Transmission au server central :

- Le RMI "ServerMain" fait des appels au RMI "ServerShopManager" (ShopService). Le RMI des magasins se déclarent, ce qui ajoute un magasin géré par le main.

## III] Solution — how we built it

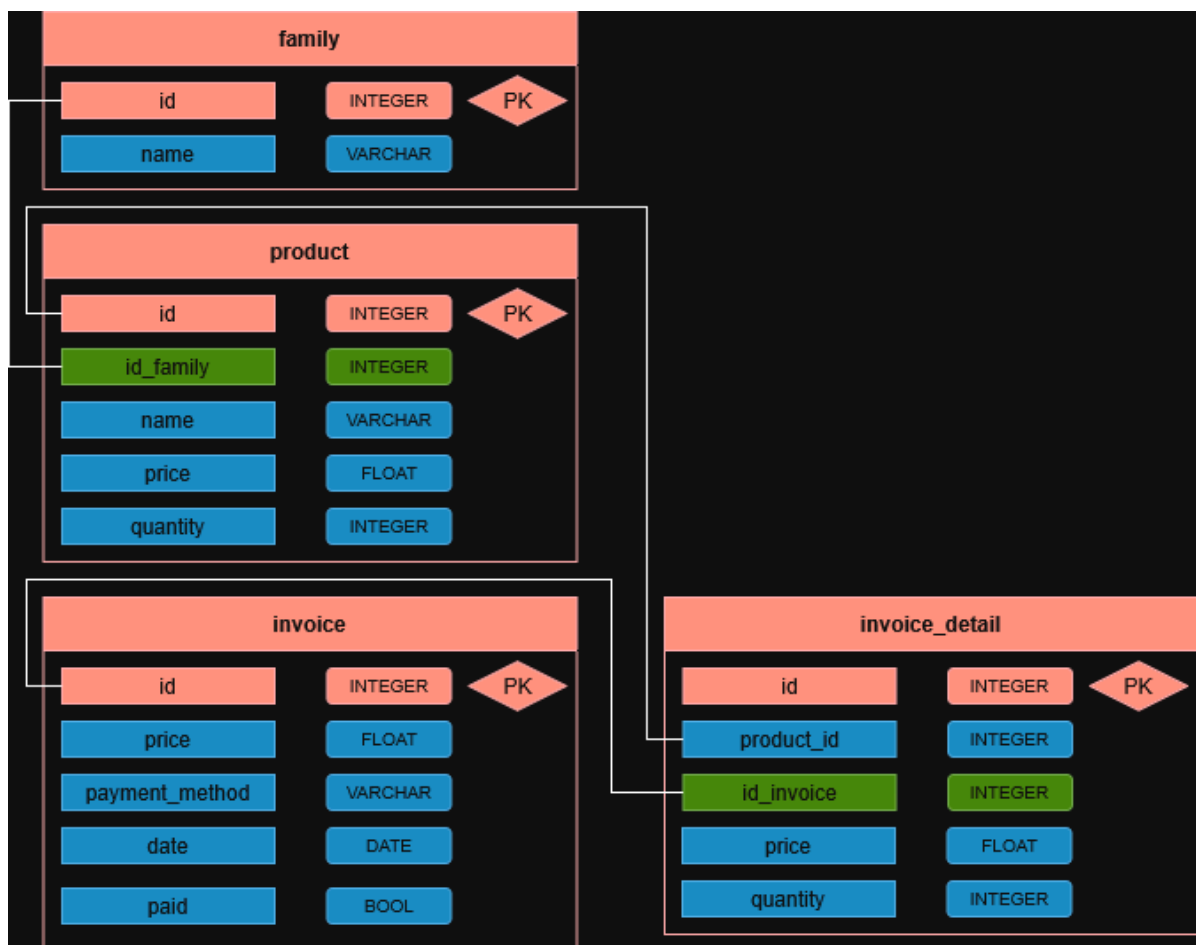
### 1. Schema

— Schemas :

→ `"/resources/init.sql"`

→ `"/resources/init-centre.sql"`

1. — Schema : Magasin



Schema de la database d'un magasin, tous les magasins partagent le même schema mais pas la même database. Il y a une unique instance MySQL hébergeant plusieurs base de données, une pour la base centrale et d'autres pour chaque magasin respectivement.

## 2. — Schema : Maison mère



— Schema du serveur principale, on stocke les produits, les magasins de notre écosystème ainsi que les factures. Les deux tables "database", "invoice\_detail"



constituent une évolution potentielle.

- Product : Cette table sera transmise au magasin.
- Shop\_invoice : Les magasins envoient leurs factures dans cette table au travers du RMI "ServerShopManager".