ECOLE MAROCAINE DES SCIENCES DE L'INGENIEUR
Membre de HONORIS UNITED UNIVERSITIES

HONORIS UNITED UNIVERSITIES

Rapport Finale du projet

Architecture des composants d'entreprise

5e annee – Ingenierie Informatique et Reseaux

Sous le thème

Réalisation d'une Application de Gestion des Devis et Factures

Réalisée par :

- MAJGHIROU Mohamed Riyad
- AZZAM Mohamed

ANNEE UNIVERSITAIRE: 2025-2026

Table des matières

Chapitre 1

Introduction

Ce projet consiste à concevoir et développer une application complète permettant de gérer les devis, les factures, les clients et les produits d'une entreprise. Le système vise à offrir une interface simple et un backend performant capable d'automatiser et centraliser le suivi commercial.

L'application s'appuie sur :

— un backend REST développé avec **Spring Boot**,

— une base de données **MySQL Aiven**,

— une interface cliente en **HTML/CSS/JavaScript**,

— un environnement de déploiement basé sur **Docker**.

Ce rapport présente les objectifs, l'architecture, la conception, l'implémentation, les résultats obtenus et les perspectives d'amélioration.

Chapitre 2

Objectifs du Projet

2.1 Objectif général

Développer une application de gestion permettant la création, la gestion et le suivi des devis et des factures de manière fiable et centralisée.

2.2 Objectifs spécifiques

— Gérer les clients : création, mise à jour, suppression, historique.
— Gérer les produits : ajout, prix, description, mise à jour.
— Générer des devis détaillés avec calcul automatique des montants.
— Convertir un devis en facture.
— Générer des factures conformes.
— Calculer des statistiques commerciales.
— Produire des documents PDF professionnels.
— Intégrer une authentification sécurisée.
— Fournir un environnement d'installation universel via Docker.

Chapitre 3

Spécifications Fonctionnelles

3.1 Gestion des Clients

— Ajouter un client avec ses informations.
— Modifier ou supprimer un client.
— Visualiser l'historique des devis et factures.

3.2 Gestion des Produits

— Enregistrer un produit (nom, prix, description).
— Modifier ou supprimer un produit.

3.3 Gestion des Devis

— Créer un devis avec liste de produits et quantités.
— Calcul automatique des totaux HT, TVA et TTC.
— Changement de statut : en cours, validé, refusé.

3.4 Gestion des Factures

— Générer des factures à partir d'un devis.
— Définir le mode de paiement.
— Suivre le statut : payé, en attente, en retard.

3.5 Statistiques

— Chiffre d'affaires global.
— Nombre de devis et factures.
— Statistiques mensuelles.

Chapitre 4

Architecture Générale

L'application adopte une architecture en couches :
— **Controller :** expose les endpoints REST.
— **Service :** contient la logique métier.
— **Repository :** interface avec MySQL via JPA.
— **Entity :** représentation objet des tables.
— **DTO/Mapper :** transfert de données et conversion.

4.1 Architecture technique

— **Backend :** Spring Boot 3, Java 17.
— **Base de données :** MySQL hébergé sur Aiven Cloud.
— **Frontend :** HTML/CSS/JS.
— **Déploiement :** Docker avec build multi-stages.

Chapitre 5

Conception

5.1 Modèle de données

Les principales entités :
— `Client`
— `Produit`
— `Devis`, `DevisDetail`
— `Facture`, `FactureDetail`
— `User` (authentification)
Relations :
— Un client possède plusieurs devis/factures.
— Un devis contient plusieurs produits (relation 1-N).
— Une facture dérive d'un devis.

5.2 Flux principal

1. Création d'un client.
2. Ajout de produits.
3. Création d'un devis.
4. Validation → conversion en facture.
5. Génération PDF.

Chapitre 6

Implémentation

6.1 Backend

6.1.1 Contrôleurs

Chaque ressource possède son contrôleur : `ClientController`, `ProduitController`, `DevisController`, `FactureController`, etc.

6.1.2 Services

La logique métier est isolée dans les classes : `ClientService`, `DevisService`, `FactureService`, etc.

6.1.3 Sécurité

Un endpoint d'authentification permet de valider un utilisateur via `UserRepository`.

6.1.4 PDF

La classe `PdfService` génère les documents PDF des devis et factures.

6.2 Base de données

Le projet utilise JPA/Hibernate avec synchronisation automatique (`ddl-auto=update`). Les scripts SQL d'insertion sont fournis dans le dossier `resources`.

6.3 Conteneurisation

6.3.1 Dockerfile

Build multi-stage :
— Compilation Maven.
— Image finale légère JRE.

6.3.2 docker-compose.yml

Permet le lancement automatique du backend.

Chapitre 7

Résultats et Validation

7.1 Fonctionnalités réalisées

— CRUD complet : clients, produits, devis, factures.
— Génération automatique des PDF.
— Transformation devis → facture.
— Visualisation des statistiques commerciales.
— API REST opérationnelle.
— Déploiement Docker fonctionnel.

7.2 Performances

Le backend répond rapidement aux requêtes grâce à Spring Boot et MySQL cloud.

7.3 Limites actuelles

— Pas encore de système avancé de rôles (admin/user).
— Frontend statique sans framework moderne.
— Manque de tests unitaires.

Chapitre 8

Conclusion

Ce projet a permis de concevoir une solution complète et fonctionnelle de gestion des devis et factures. L'architecture Spring Boot, associée à MySQL Aiven et Docker, garantit une application robuste, portable et facilement déployable.

Les objectifs initiaux ont été atteints : une API stable, une gestion complète des ressources, un système PDF, un ensemble de statistiques et un déploiement conteneurisé.

Des améliorations futures incluent :

— authentification avancée,
— interface moderne en React ou Vue.js,
— ajout de tests automatiques,
— optimisation UI/UX.