CAHIER DE CHARGES

Application de gestion de quincaillerie

Canisius

IFU 00292389M | Ouagadougou, 0022670612916, canisiushien@gmail.com

Cahier des charges – Application de gestion de quincaillerie

1. CONTEXTE ET OBJECTIFS

- Objectif général : concevoir, développer et déployer une application de gestion de quincaillerie permettant d'améliorer la gestion des stocks, des ventes, des clients et des fournisseurs, avec une interface utilisateur simple et accessible pour des utilisateurs non techniques.
- Domaine fonctionnel : articles, stocks, ventes, facturation, clients, fournisseurs, rapports, sécurité et administration.
- Portée : solution complète (frontend web, backend API, base de données, déploiement et sauvegardes) accessible sur serveur local ou cloud selon les besoins du maître d'ouvrage.

2. PORTEE FONCTIONNELLE DETAILLEE

2.1 Gestion des articles

- Création d'articles : assignation d'un identifiant, désignation, référence, catégorie, unité de mesure, seuil d'alerte, prix d'achat, prix de vente, fournisseur principal (optionnel), code barre (optionnel), poids, dimensions, fabricant, SKU, codes internes, description.
- Modification et suppression d'articles.
- Catégorisation des produits (par exemple : quincaillerie générale, matériaux, consommables, outils, peinture, etc.).
- Gestion des prix : prix d'achat, prix de vente, éventuels remises ou marges par article.
- Alertes : seuil de stock minimum par article, notification en cas de dépassement ou de rupture imminente pour chaque article.

2.2 Gestion des stocks

- Entrées de stock : réception de marchandise, association avec article, quantité, date, fournisseur, numéro de lot si applicable.
- Sorties de stock : ventes, retours, déstockages, inventaire.
- Historique des mouvements : journal des entrées/sorties avec horodatage, utilisateur et motif (réception, vente, retour, inventaire, déstockage).
- État du stock en temps réel : quantité disponible, quantité réservée (si appliqué), valeur du stock.
- Inventaire périodique et synchronisation des stocks si nécessaire.
- Gestion des transferts entre entrepôts (optionnel si multi-emplacements).

2.3 Gestion des ventes

- Enregistrement des ventes : création d'une vente avec items, quantités, prix facturé, remise éventuelle, taxes (si applicable), mode de paiement (espèce, mobile money, chèque, TPE).
- Génération automatique de factures : numéro de facture, date, client, articles, TVA, total.
- Impression et export des factures : PDF imprimable, export Excel si nécessaire.
- Calcul automatique des totaux : sous-totaux, remises, TVA, total et éventuels écarts.
- Gestion des statuts de vente (en cours, payée, partiellement payée, annulée).
- Gestion des paiements et échéances associées (si besoin).

2.4 Gestion des clients et des fournisseurs

- Gestion des fiches ou registres clients : nom, coordonnées (adresse, téléphone, email), conditions de paiement, notes.
- Gestion des fiches ou registres fournisseurs : nom, coordonnées, conditions de paiement, délai de livraison.
- Historique des transactions par client/fournisseur : date de la transaction, montant total convenu, montant reçu de la transaction, référence de transaction, type (ventes, achats, retours, paiements, ajustements), commentaire.
- Recherche et filtrage rapide par nom, téléphone, email ou identifiant.

2.5 Rapports et statistiques

- Rapports périodiques d'achats et ventes : journalier, mensuel, annuel.
- État des stocks : niveaux par articles, valeur du stock.
- Chiffre d'affaires et bénéfices : par période, par catégorie, par client/fournisseur.
- Rapports personnalisables (filtres : date, catégorie, fournisseur, client).
- Export des rapports : PDF imprimable et Excel.
- Tableaux de bord et indicateurs clés (KPI) affichés sur la page d'accueil (ventes quotidiennes, stock faible, etc.).

2.6 Sécurité et accès

- Authentification ou connexion au logiciel : identifiant (email ou login) et mot de passe.
- Gestion des rôles : administrateur, vendeur (et éventuellement autres rôles selon besoin).
- Autorisations par rôle : accès lecture/écriture, modification, suppression, gestion des utilisateurs (pour l'administrateur).
- Journal d'audit : enregistrement des actions sensibles (création/suppression/modification d'articles, transferts de stock, facturation, gestion des utilisateurs).

3. CONTRAINTES TECHNIQUES ET ARCHITECTURE

3.1 Architecture cible

- Frontend ou interfaces utilisateurs : Application Web en Angular.

- Backend ou logique métier : API REST en Spring Boot.
- Base de données : PostgreSQL.
- Déploiement : option serveur local ou cloud (à déterminer avec le maître d'ouvrage).
- Sauvegarde : sauvegarde automatique des données (fréquence à définir : quotidienne, multiplicité, rétention).

3.2 Contraintes non fonctionnelles

- Disponibilité et performance : architecture scalable pour supporter une taille d'inventaire et un volume de transactions croissants.
- Sécurité : protections standard (hashage des mots de passe, TLS, gestion des sessions, validation côté serveur et côté client).
- Accessibilité : interface claire, navigation inter page intuitive, écran responsive.
- Maintenabilité : code structuré, documentation technique, tests unitaires et d'intégration, procédures de déploiement.
- Compatibilité : prêt pour une utilisation sur postes Windows/macOS via navigateur web moderne (Google chrome, Mozilla Firefox, Opera, MS Edge, etc.).
- Multilingue : préférence pour une solution prête à localisation si nécessaire (mais en français par défaut).

3.3 Environnements et déploiement

- Environnement de développement : IDE/stack recommandés, conventions de branchement et workflow (Git).
- Environnement de test : jeux de données simulés (articles, stocks, ventes, clients, fournisseurs).
- Environnement de production : déploiement sur serveur local ou cloud (à préciser), mécanismes de sauvegarde et restauration.
- Plan de continuité : sauvegardes récurrentes, plan de restauration, management des incidents.

4. RÉFÉRENTIELS ET QUALITÉ

- Méthodologie : agile Scrum avec livrables itératifs et livraisons incrémentales.
- Définition de "done" (Definition of Done): fonctionnalités testées, code couvert par tests unitaires et d'intégration, documentation utilisateur et opérateur, déploiement en environnement cible.
- Assurance qualité : tests fonctionnels, tests d'intégration, tests de sécurité de base (OWASP top 10), revue de code.
- Documentation:
    - Dossier techniques (architecture, schémas, API specs).
    - Guide utilisateur et aide en ligne (intégrés dans la solution).
    - Procédures d'installation et de déploiement.

- Guide de gestion des sauvegardes et de restauration.

5. LIVRABLES ATTENDUS

- Spécifications fonctionnelles détaillées (SFD) et leur traçabilité vers les exigences métier.
- Spécifications techniques (architecture, schémas de données, API).
- Maquettes/UX pour les écrans clés (portail article, stock, vente, client/fournisseur, rapports).
- Code source balisé et structuré.
- Jeux de tests (unitaires, d'intégration, scénarios fonctionnels).
- Script de déploiement et configurations d'infrastructure.
- Documentation utilisateur et opérateur.
- Plan de maintenance et support.

Ces livrables demeurent la propriété du maître d'œuvre. Mais peuvent être transmis au maître d'ouvrage en fonction du mode d'acquisition du logiciel.

6. PLAN DE TRAVAIL ET LIVRABLES PAR PHASE

Phase 1 – Analyse et design (2-4 semaines)

- Ateliers avec le maître d'ouvrage pour finaliser les besoins.
- Définition du modèle de données et des flux métiers.
- Élaboration du dossier technique détaillé et des API contracts.
- Maquettes des écrans.

Phase 2 – Développement de l'OS base (4-8 semaines)

- Implémentation des modules : articles, stocks, ventes, clients/fournisseurs, rapports.
- Mise en place de l'authentification et des rôles.
- Intégration PostgreSQL et API Spring Boot.
- Développement frontend Angular.

Phase 3 – Tests et qualité (2-4 semaines)

- Tests unitaires et d'intégration.
- Tests fonctionnels par scénarios.
- Vérifications de sécurité et performance.
- Correction des anomalies.

Phase 4 – Déploiement et mise en production (1-2 semaines)

- Déploiement sur l'environnement choisi (local ou cloud).
- Mise en place des sauvegardes et de la surveillance.
- Transfert de connaissances et formation des utilisateurs clés.

Phase 5 – Exploitation et maintenance (Ongoing)

- Support, enhancements, et évolutions post-déploiement.

7. CRITÈRES D'ACCEPTATION

- Conformité aux exigences fonctionnelles et non fonctionnelles.
- Taux de couverture des tests (sécurité et qualité).
- Performance acceptable pour les charge attendues (temps de réponse des pages, temps de génération des factures).
- Exploitabilité en environnement cible (local ou cloud).
- Documentation complète et formation fournie.

8. RISQUES ET ATTÉNUATION

- Risque : complexité de l'intégration multi-entrepôts ou multi-contrats. Atténuation : démarrer par un périmètre initial et planifier les extensions.
- Risque : surcoût lié à la migration des données existantes. Atténuation : définir une stratégie de migration et un plan de données initial.

9. ANNEXES

- Diagrammes (use cases, classes, architecture).
- Modèles de données (schémas PostgreSQL).
- API contracts (endpoints, méthodes, schémas JSON, exemples).
- Exercices de migration et règles d'audit.

10. VALIDATION ET SIGNATURES

- Parties prenantes : Maître d'ouvrage, prestataire développeur.
- Dates de validation et signatures.