#### 

# **DOCUMENT DE CONCEPTION DETAILLE**

#### **Introduction**

* Objectif du Document : Ce document vise à décrire en détail l'architecture, les fonctionnalités et les composants du système de gestion des ventes pour le supermarché afin de guider le développement, la livraison et la maintenance.
* Portée du Document : Il couvre les aspects techniques de l'interface utilisateur, de la logique métier, de la sécurité, de la base de données, des performances, de la maintenance et de l'évolutivité du système.

#### **Architecture Générale**

* Architecture à Niveaux : L'architecture est basée sur une approche à trois niveaux comprenant le frontend (interface utilisateur), le backend (logique métier et services), et la base de données (stockage des données).
* Composants Majeurs : Les composants clés incluent le serveur web, le système de gestion de base de données, les API RESTful pour les services backend, les composants d'interface utilisateur (pages, widgets, formulaires) et les services de sécurité.
* Flux de Données : Le flux de données commence par les requêtes des utilisateurs via l'interface utilisateur, qui sont traitées par les services backend, accèdent à la base de données pour récupérer ou modifier les données, puis renvoient les résultats à l'interface utilisateur pour affichage.

#### **Interface Utilisateur (UI)**

* Pages et Fonctionnalités : Chaque page de l'interface utilisateur est détaillée avec ses fonctionnalités associées, comme la gestion des produits, des clients, la visualisation des rapports, etc.

#### **Backend et Services**

* Services RESTful : Chaque service backend est décrit en détail avec ses Endpoint, les méthodes HTTP supportées (GET, POST, PUT, DELETE), les paramètres requis, les réponses attendues, et les actions effectuées par chaque Endpoint.
* Logique Métier : Chaque service backend est accompagné d'une description détaillée de la logique métier qu'il implémente, y compris les règles de traitement des données, les validations, les calculs, etc.

#### **Base de Données**

* Schéma de Données :
* Modèle Conceptuel de Données (MCD)

Le Modèle Conceptuel de Données représente de manière abstraite les entités, les attributs et les relations du système de gestion des ventes du supermarché.

* Entité Produit :

Attributs : ID Produit (Clé primaire), Nom, Catégorie, Prix Achat, Prix Vente, Quantité en Stock, etc.

Relations : Une Vente peut concerner plusieurs Produits.

* Entité Client :

Attributs : ID Client (Clé primaire), Nom, Prénom, Sexe.

Relations : Un Client peut effectuer plusieurs Ventes.

* Entité Vente :

Attributs : ID Vente (Clé primaire), Date, ID Produit (Clé étrangère), Quantité Vendue, Prix de Vente, Montant Total, etc.

Relations : Une Vente concerne un seul Produit. Une Vente est réalisée par un Client.

* Entité Stock :

Attributs : IDStock (Clé primaire), IDProduit (Clé étrangère vers Produit), QuantiteStock (Quantité en stock), DateMiseAJour (Date de la dernière mise à jour du stock).

Ce Modèle Conceptuel de Données représente les principales entités et relations nécessaires pour le système de gestion des ventes du supermarché, en prenant en compte les interactions entre les Produits, les Clients, les Ventes et les stocks.

* Modèle Logique de Données (MLD)

Le Modèle Logique de Données se base sur le Modèle Conceptuel de Données pour définir la structure relationnelle de la base de données.

* Table Produit :

Colonnes :

IDProduit (Clé primaire, INT)

Nom (VARCHAR)

Type (VARCHAR)

PrixAchat (DECIMAL)

PrixVente (DECIMAL)

QuantiteStock (INT)

* Table Client :

Colonnes :

IDClient (Clé primaire, INT)

Nom (VARCHAR)

Prenom (VARCHAR)

Sexe (VARCHAR)

DateAchat (DATE)

* Table Vente :

Colonnes :

IDVente (Clé primaire, INT)

Date (DATE)

IDProduit (Clé étrangère, INT)

QuantiteVendue (INT)

PrixVente (DECIMAL)

MontantTotal (DECIMAL)

* Table Stock :

Colonnes :

IDStock (Clé primaire, INT, AUTO\_INCREMENT)

IDProduit (Clé étrangère vers Produit)

QuantiteStock (INT)

DateMiseAJour (DATE)

Ce Modèle Logique de Données traduit les entités du Modèle Conceptuel de Données en tables relationnelles avec des attributs spécifiques, des clés primaires et des clés étrangères pour représenter les relations entre les entités. Les types de données sont également précisés pour chaque colonne afin de définir la structure de la base de données.

* Modèle Physique de Données (MPD)

Le Modèle Physique de Données traduit le Modèle Logique de Données en une structure adaptée à un système de gestion de base de données spécifique, comme MySQL par exemple.

* Table Produit :

Colonnes :

IDProduit (Clé primaire, INT, AUTO\_INCREMENT)

Nom (VARCHAR(255))

Type (VARCHAR(100))

PrixAchat (DECIMAL(10, 2))

PrixVente (DECIMAL(10, 2))

QuantiteStock (INT)

* Table Client :

Colonnes :

IDClient (Clé primaire, INT, AUTO\_INCREMENT)

Nom (VARCHAR(255))

Prenom (VARCHAR(255))

Sexe (TEXT)

DateAchat (DATE)

* Table Vente :

Colonnes :

IDVente (Clé primaire, INT, AUTO\_INCREMENT)

Date (DATE)

IDProduit (Clé étrangère, INT)

QuantiteVendue (INT)

PrixVente (DECIMAL(10, 2))

MontantTotal (DECIMAL(10, 2))

* Table Stock :

Colonnes :

IDStock (Clé primaire, INT, AUTO\_INCREMENT)

IDProduit (Clé étrangère vers Produit)

QuantiteStock (INT)

DateMiseAJour (DATE)

Ce Modèle Physique de Données détaille la structure spécifique des tables et des colonnes pour notre base de données SQL Server. Les types de données sont précisés avec les longueurs et les précisions appropriées pour chaque colonne, et les clés primaires, les clés étrangères et les contraintes d'auto-incrémentation sont également spécifiées pour garantir l'intégrité des données et la cohérence du système de gestion des ventes.

#### **Sécurité**

#### 1. Authentification et Autorisation :

#### - Utiliser Flask-Login pour gérer l'authentification des utilisateurs. S’assurer que chaque utilisateur a des identifiants uniques et sécurisés pour se connecter au système.

#### - Implémenter des contrôles d'autorisation basés sur les rôles et les permissions pour limiter l'accès des utilisateurs aux différentes parties de notre application en fonction de leurs droits.

#### 2. Chiffrement des Données :

#### Utiliser des méthodes de chiffrement sécurisées, comme Flask-Bcrypt, pour stocker les mots de passe des utilisateurs de manière sécurisée dans la base de données SQL Server. S’assurer de ne jamais stocker les mots de passe en texte brut.

#### 3. Prévention des Attaques :

#### - Utiliser Flask-WTF pour la gestion sécurisée des formulaires afin de prévenir les attaques CSRF (Cross-Site Request Forgery).

#### - S’assurer de valider et de filtrer toutes les entrées utilisateur pour éviter les attaques d'injection SQL et les attaques XSS (Cross-Site Scripting).

#### 4. Gestion des Sessions :

#### Utiliser les fonctionnalités de session sécurisées de Flask pour gérer les sessions utilisateur de manière sécurisée. S’Assurer que les informations sensibles ne sont pas stockées dans les cookies ou les sessions côté client.

#### 5. Contrôles d'Accès :

#### Implémenter des contrôles d'accès au niveau des routes et des fonctionnalités de notre application pour garantir que seuls les utilisateurs autorisés peuvent accéder aux données et aux fonctionnalités sensibles.

#### 6. Audit et Journalisation :

#### Mettre en place des mécanismes d'audit et de journalisation pour surveiller et enregistrer les activités des utilisateurs, les tentatives d'accès non autorisées et les erreurs système. Cela vous permettra de détecter rapidement les problèmes de sécurité et d'y réagir.

#### En suivant ces bonnes pratiques de sécurité, vous pourrons renforcer la sécurité de notre application Flask avec SQL Server et protéger efficacement les données sensibles, l'accès aux fonctionnalités et l'intégrité du système contre les menaces potentielles.

#### **Performances**

* Stratégies de Performance :

Optimisation du Code :

- Exemple : Révision du code pour éviter les boucles inefficaces et optimiser les algorithmes utilisés pour le traitement des données.

- Explication : En optimisant le code, on réduit le temps d'exécution des opérations, on minimise la consommation de ressources et on améliore la réactivité globale du système.

* Tests de Performance : La méthodologie utilisée pour les tests de performance, les résultats obtenus, les améliorations apportées en fonction des résultats, sont détaillés pour garantir des performances optimales du système.

#### 

#### **Gestion des Données**

* Modèle de Données : Une analyse approfondie du modèle de données sera réalisée, incluant les relations entre les entités, les règles de normalisation, les contraintes d'intégrité, afin d'assurer une gestion efficace et cohérente des données dans notre système de gestion des ventes pour le supermarché.

#### Dans le cadre du projet de développement du système de gestion des ventes pour le supermarché, le traitement des données implique plusieurs étapes clés pour assurer une gestion fiable et cohérente des données :

#### Collecte des Données : Nous mettrons en place des mécanismes pour récupérer les données de vente à partir de diverses sources telles que les transactions en caisse, les commandes en ligne et d'autres canaux pertinents. Ces données seront intégrées de manière transparente dans notre système de gestion des ventes pour une utilisation efficace.

#### Traitement des Données : Nous établirons des procédures de traitement des données qui comprendront la validation, le nettoyage, la transformation et l'enrichissement des données. L'objectif est de garantir que nos données sont complètes, précises et cohérentes, ce qui est essentiel pour des analyses précises et fiables.

#### Stockage des Données : Nous choisirons l'infrastructure de stockage des données la mieux adaptée à nos besoins, qu'il s'agisse de bases de données relationnelles comme SQL Server ou d'autres solutions de stockage adaptées. Le stockage des données sera sécurisé, évolutif et conforme aux réglementations en matière de protection des données.

#### Recherche et Manipulation des Données : Nous mettrons en place des méthodes et des outils pour rechercher, extraire et manipuler les données en fonction des besoins spécifiques de notre système de gestion des ventes. Cela peut inclure l'utilisation de requêtes SQL pour récupérer des données spécifiques, la création de vues pour faciliter l'analyse et l'utilisation d'algorithmes de traitement des données pour identifier des tendances ou des modèles.

#### En décrivant ces étapes, nous assurerons une gestion cohérente et fiable de nos données tout au long du processus, ce qui est crucial pour des analyses efficaces, des prises de décision éclairées et une expérience utilisateur optimale.