

SmartShop 360 – Intégration E-commerce & Agent IA métier avec UI

1. Contexte métier :

SmartShop 360 est un e-commerçant B2C de taille moyenne, spécialisé dans la **Décoration & Cadeaux**, qui vend ses produits via deux canaux principaux : son propre site e-commerce et plusieurs marketplaces internationales.

Le problème actuel (Silos Métier) : Les équipes métier sont aujourd'hui aveugles car elles travaillent de manière cloisonnée :

- **L'équipe Ventes & Finance** (ERP/Site) pilote le Chiffre d'Affaires, les marges et les volumes de vente (Données structurées).
- **L'équipe Marketing & Qualité** (Web) surveille la réputation de la marque et les avis clients (Données non structurées).

L'objectif stratégique : La Direction souhaite briser ces silos pour obtenir une **vue consolidée à 360°**. Elle veut pouvoir répondre à des questions croisées complexes telles que : *"Quels sont nos best-sellers qui commencent à avoir une mauvaise réputation ?"* ou *"Quels produits de niche ont des fans très fidèles ?"*.

Pour y parvenir, le projet vise à :

1. **Unifier les données** de transactions (ventes) et d'avis produits (qualité) au sein d'un entrepôt de données commun.
2. **Construire une UI simple** (Tableau de bord) pour visualiser les indicateurs clés unifiés.
3. **Mettre en place un Agent IA métier** capable de dialoguer avec les utilisateurs (marketing/qualité) et d'interroger ces données intégrées en langage naturel.

2. Problème d'intégration à résoudre

La situation actuelle (Legacy en Silos) : Le système d'information actuel est fragmenté en deux mondes qui ne se parlent pas :

- **Le Monde Transactionnel (ERP/Ventes) :** Les données structurées (factures, produits, clients, dates) résident dans la base e-commerce interne, simulée ici par le dataset **Online Retail II**. <https://www.kaggle.com/datasets/mashlyn/online-retail-ii-uci>

- **Le Monde Réputationnel (Avis/Web)** : Les données non structurées (textes d'avis, sentiments) sont dispersées sur des plateformes externes, simulées par le dataset **ecommerce-product-reviews-sentiment**. [README.md](#) · [dipawidia/ecommerce-product-reviews-sentiment at main](#)

Le défi d'Urbanisation : Le problème technique majeur est l'**absence de référentiel commun**.

- Il n'existe aucun mécanisme robuste pour aligner les produits entre ces deux sources.
- *Exemple concret* : Le produit identifié ID_123 ("Mug Blue") dans l'ERP ne porte pas le même identifiant ni exactement le même nom sur les marketplaces externes.
- **Conséquence technique** : Il est impossible d'effectuer une jointure simple pour croiser le volume de ventes et la perception client.

Conséquences opérationnelles :

- **Métier** : Impossible de répondre rapidement à des questions corrélées comme:
 - *“Quels produits réalisent beaucoup de ventes mais reçoivent des avis majoritairement négatifs ?”*
 - *“Quels segments de clients sont rentables ET satisfaits ?”*
- **Inefficacité** : Les équipes dépendent d'un lourd travail manuel sous Excel, source d'erreurs et sans traçabilité.
- **Absence d'Intelligence** : Aucune capacité d'IA dans le SI actuel pour assister la décision, seulement des analyses post-mortem.

La Cible (To-Be) : Le projet doit sortir de ce mode "bricolage" en proposant une **architecture d'intégration** (ETL + Master Data Management) surmontée d'une **couche IA + UI**. Le cœur du problème à résoudre sera la création d'une **table de correspondance (Mapping)** pour réconcilier les deux sources de données.

3. Objectifs du projet :

3.1 Intégration de données (ETL & MDM)

- **Source 1 – Transactions (Kaggle – Online Retail II)**
 - Extraction depuis le fichier Excel/CSV.
 - **Nettoyage** : Filtrer les commandes annulées (codes 'C'), gérer les formats de dates.

- Chargement dans une table SALES_TRANSACTIONS (PostgreSQL)².
- **Source 2 – Avis clients (Hugging Face – ecommerce-product-reviews)**
 - Extraction via la librairie datasets ou fichier JSON.
 - Chargement d'un échantillon significatif (ex: 500-1000 avis) dans une table CUSTOMER_REVIEWS.
- **Master Data Management**
 - *Problème* : Les IDs produits ne correspondent pas nativement entre les deux fichiers.
 - **Consigne MDM** : Vous devez construire une **Table de Mapping** artificielle pour simuler un "Golden Record".
 - *Méthode* : Prenez le Top 50 des produits vendus (Source 1) et associez-les arbitrairement aux 50 produits les plus commentés (Source 2) via une table de jointure PRODUCT_MAPPING.
 - *Objectif* : Permettre les jointures SQL (ex: JOIN ... ON map.id_source1 = sales.prod_id).
- **Pattern Technique** : Pipeline ETL scripté en Python (Pandas + SQLAlchemy) vers la base intégrée.

3.2 Mini-orchestrateur IA (Agent SQL)

L'objectif est de créer un système de type "Chat with your Data" (RAG structuré).

- **L'Agent (Le Cerveau)**
 - Utiliser un LLM via API (Groq, Mistral, OpenAI) pour garantir la qualité de la génération SQL⁵.
 - *Note* : Un modèle local est autorisé s'il est assez performant pour du code SQL.
 - **Rôle** : L'agent agit comme un "Data Analyst" qui connaît le schéma de la base de données (noms des tables et colonnes).
- **Les Outils (Tools)**
 - **Tool SQL_Executor (Obligatoire)** : L'agent génère du code SQL (ex: SELECT product_name, AVG(sentiment) FROM...), le tool l'exécute sur Postgres et renvoie le résultat brut⁶.

- **Tool Analysis (Optionnel)** : Un script Python pour des calculs statistiques complexes que le SQL gère mal⁷.
- **Boucle d'Orchestration**
 1. **User** : *"Quels produits vendus à plus de 100 unités ont une note inférieure à 3/5 ?"*⁸
 2. **Agent** : Analyse la question \rightarrow Génère la requête SQL.
 3. **Tool** : Exécute la query sur la base unifiée.
 4. **Agent** : Récupère les données et formule la réponse en langage naturel.

3.3 UI Frontend / Dashboard (Rapid Application Development)

- **Technologie Recommandée : Streamlit ou Gradio**
 - Ces frameworks Python permettent de créer des dashboards interactifs en quelques lignes de code, vous permettant de vous concentrer sur l'architecture des données et l'IA plutôt que sur le CSS/HTML.
 - React/Vue.
- **Écrans attendus :**
 - **Dashboard "Data Quality"** : Affiche le taux de couverture du mapping (combien de produits unifiés ?) et les KPIs globaux¹⁰.
 - **Interface "Chat"** : Une zone de texte pour dialoguer avec l'Agent et visualiser ses réponses¹¹.
 - **Fiche Produit 360°** : Une vue détaillée combinant courbe des ventes (Source 1) et derniers avis (Source 2)

4. Livrables attendus :

4.1 Modélisation & Architecture (Livrables Urbaniste)

- **Cartographie du SI (Format ArchiMate ou UML) :**
 - **Vue "Métier"** : Représentez les processus existants (Vente sur site, Vente Marketplace, Analyse Qualité) et les acteurs.
 - **Vue "Transformation"** :

- Distinguez clairement le **SI Opérationnel** (Sources : ERP, Web) du **SI Décisionnel** (Cible : Data Warehouse).
- Mettez en évidence le composant "**Référentiel Produit**" (**MDM**) qui sert de pivot entre les silos.
- *Outils recommandés* : **Archi** (ArchiMate), PlantUML, ou Mermaid.
- **Diagrammes Techniques** :
 - **Diagramme de Flux de Données (DFD)** : Détaillez le pipeline ETL : Source \rightarrow Nettoyage \rightarrow **Mapping/Unification** \rightarrow Base Finale.
 - **Schéma d'Architecture Physique** : Conteneurs (PostgreSQL, App Python/Streamlit), flux vers l'API LLM externe.

4.2 Proof Of Concept (POC) Fonctionnel

Le code doit être fourni (repo Git) avec un README clair.

- **Le Backend & Data Pipeline** :
 - **Script ETL (etl.py)** : Ingestion des fichiers Kaggle/HuggingFace, création de la table de mapping (MDM simulé) et chargement dans PostgreSQL.
 - **Logique Agentique** : Implémentation de l'Agent capable de traduire une question naturelle en SQL valide (Text-to-SQL) et d'exécuter la requête.
- **L'Interface Utilisateur (UI)** :
 - Application **Streamlit** (ou Gradio/Web classique) fonctionnelle.
 - **Dashboard** : Visualisation des KPIs unifiés (Ventes + Avis).
 - **Chatbot Data** : Zone d'interaction où l'on voit la question de l'utilisateur, la requête SQL générée par l'IA (pour débogage/transparence), et la réponse finale.
- **Livrables Repository** :
 - Fichier requirements.txt ou Dockerfile.
 - README.md expliquant comment lancer la démo en 3 étapes maximum (ex: docker-compose up).

4.3 Documentation Urbanistique & Métier

Document synthétique (PDF/Markdown) justifiant vos choix :

- **Stratégie MDM (Master Data Management) :**
 - Expliquez comment vous avez géré le mapping dans le projet.
 - **Important :** Décrivez comment vous auriez procédé dans un contexte réel (ex: utilisation de codes EAN/GTIN, algorithmes de Fuzzy Matching sur les noms de produits).
- **Dictionnaire de Données :** Définition des tables clés de votre entrepôt (Unified_Product, Sales_Facts, Review_Facts).
- **Scénario de Démonstration :**
 - Script de la vidéo obligatoire (5 à 10 min) : Montrez l'ingestion, puis posez une question complexe à l'agent (ex: *"Quels produits vendus à >50 exemplaires ont une note < 3 ?"*) et analysez sa réponse.