DOCUMENTATION DE SUPPORT UTILISATEUR

Projet Amazon Reviews :

Analyse et classification des avis Clients

BONNAT Jonathan

Data Engineer - B2B-LP-DESFL

💬INTRODUCTION

OBJECTIF DU DOCUMENT

Ce document a pour objectif de fournir aux utilisateurs techniques et opérationnels l'ensemble des informations nécessaires pour exploiter, superviser et comprendre le fonctionnement du système d'analyse et de classification des avis clients.

Il décrit les composants principaux de l'architecture, les procédures d'exploitation, les modalités de support et les mécanismes de supervision à mettre en place pour assurer la continuité de service.

PUBLIC CONCERNÉ

Ce document s'adresse principalement :

- aux data engineers chargés d'exécuter, diagnostiquer ou relancer les traitements;
- aux équipes d'exploitation responsables de la supervision et du maintien en condition opérationnelle;
- aux analystes ou utilisateurs métiers intervenant ponctuellement via l'API;
- aux administrateurs en charge des accès et de la sécurité.

CONTEXTE ET PÉRIMÈTRE FONCTIONNEL

Le système couvre l'ensemble de la chaîne de traitement des avis : extraction depuis PostgreSQL, enrichissement et classification via un pipeline ETL/NLP, stockage structuré dans S3 et exposition des résultats via une API FastAPI.

Le présent document ne traite pas du développement du modèle ou du pipeline, mais exclusivement de l'usage, du support et du fonctionnement opérationnel de la solution en production.

RAPPELS DE L'ARCHITECTURE APPLICATIVE (API, S3, AIRFLOW)

L'architecture se compose des éléments suivants :

- Airflow : orchestration du pipeline ETL/NLP,
- PostgreSQL : base source des avis et métadonnées,
- S3 : Data Warehouse contenant les données enrichies et calculées au format Parquet,
- MongoDB : archivage des données brutes,
- FastAPI : service d'exposition pour consommation par le frontend.

DOCUMENTATION TECHNIQUE DE L'ARCHITECTURE

PRÉSENTATION GÉNÉRALE DU SYSTÈME

Le système d'analyse des avis clients s'appuie sur une architecture modulaire permettant d'extraire, transformer, enrichir et rendre accessibles des données de manière fiable.

L'ensemble repose sur cinq briques principales : un orchestrateur (Airflow), une base source (PostgreSQL), un stockage analytique (S3), un archivage (MongoDB) et un service d'exposition (FastAPI).

Chaque composant joue un rôle spécifique dans la chaîne de traitement afin d'assurer une automatisation complète, une bonne traçabilité et une restitution rapide des résultats.

DESCRIPTION DES COMPOSANTS TECHNIQUES

1. **Airflow (orchestration)**

   Airflow gère l'exécution du pipeline ETL/NLP. Il orchestre les tâches d'extraction, de transformation, de classification Zero-Shot, de génération du fichier Parquet et d'archivage MongoDB. Le DAG reflète la séquence opérationnelle du traitement.

2. **S3 (Data Lake)**

   S3 stocke les données enrichies et calculées produites par le pipeline, au format Parquet. La structure du bucket distingue les données calculées, rejetées et enrichies. Le fichier review_curated_XXX.parquet constitue la source unique pour l'API.

3. **PostgreSQL (source)**

   PostgreSQL est la base de données contenant les avis bruts, les produits et les utilisateurs. Elle sert de point d'entrée du pipeline d'extraction. L'accès s'effectue en lecture seule afin de préserver l'intégrité de la donnée source.

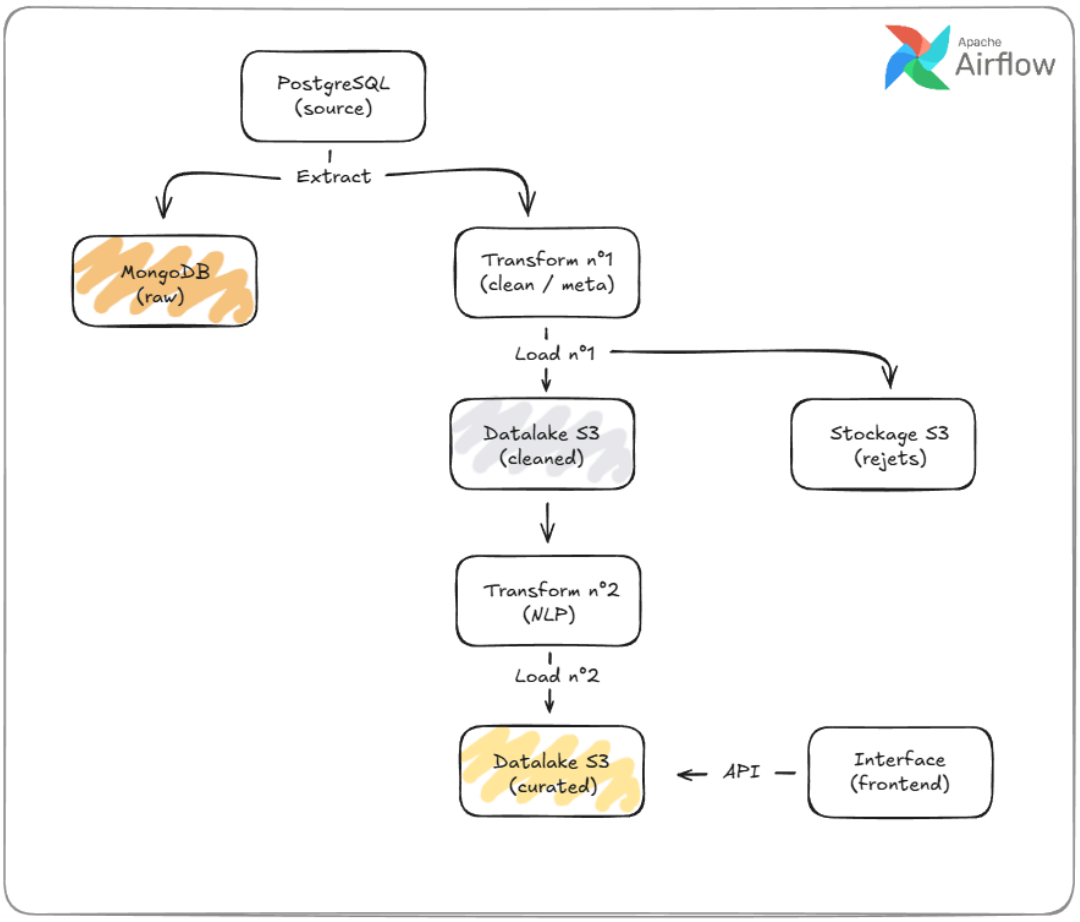
4. **MongoDB (archivage)**

   MongoDB stocke une copie JSON des avis bruts. Cette base sert d'archive pour la traçabilité et permet un éventuel retraitement sans relire PostgreSQL qui pourrait avoir changé.

5. **FastAPI (exposition)**

   FastAPI fournit une interface REST permettant au frontend d'accéder aux avis les plus pertinents. Il lit directement le fichier Parquet sur S3 pour offrir une réponse rapide et structurée.

SCHÉMA D'ARCHITECTURE



FLUX DE DONNÉES

Le fonctionnement global se déroule en plusieurs étapes :

- **Extraction** : Airflow lit les avis et métadonnées depuis PostgreSQL.
- **Stockage** : Les données brutes sont stockées dans MongoDB pour archivage
- **Transformation** : les données sont nettoyées et enrichies avec les métadonnées disponibles
- **Stockage** : les résultats nettoyés sont écrits au format Parquet dans S3 (dossier /cleaned)
- **Traitement NLP** : les données passent dans un modèle de classification zéro-shot et une catégorie et un score de confiance pondéré y est ajouté
- **Stockage** : les données calculées sont écrites au format Parquet dans S3 (cette fois ci dans /curated)
- **Exposition** : FastAPI lit le Parquet et expose les avis les plus pertinents au

frontend.

Ce flux garantit une mise à jour cohérente et centralisée du Data Lake.

ENDPOINTS DE L'API

- GET /health : Permet de vérifier que l'API est opérationnelle.
- GET /reviews/{product_id}/top : Retourne les avis les plus pertinents pour un produit donné, à partir du fichier Parquet sur S3.

PROCÉDURES D'EXPLOITATION

DEMARRAGE ET ARRET DES SERVICES

1. **Airflow**
    - Pour démarrer :
        - Lancer la commande *"docker compose up airflow-scheduler airflow-webserver airflow-worker"*
    - Pour arrêter :
        - Lancer la commande *"docker compose down airflow-scheduler airflow-webserver airflow-worker"*
    - Pour redémarrer :
        - Lancer la commande *"docker compose restart airflow-webserver airflow-scheduler airflow-worker"*
        - Pour vérifier l'état du service :
            - Lancer la commande *"docker compose ps"*

2. Base PostgreSQL
    - Pour démarrer :
        - Lancer la commande *"docker compose up -d postgres"*
    - Pour arrêter :

        - Lancer la commande *“docker compose stop postgres*
    - Pour redémarrer :
        - Lancer la commande *“docker compose restart postgres”*
    - Pour consulter les logs :
        - Lancer la commande *“docker compose logs postgres”*

3. MongoDB
    - Pour démarrer :
        - Lancer la commande *“docker compose up -d mongodb”*
    - Pour arrêter :
        - Lancer la commande *“docker compose stop mongodb”*
    - Pour redémarrer :
        - Lancer la commande *“docker compose restart mongodb”*
    - Pour tester la connexion :
        - Lancer la commande *“docker exec -it mongodb mongosh --eval "db.stats()"”*

4. FastAPI
    - Pour démarrer :
        - Lancer la commande *“docker compose up -d api”*
    - Pour arrêter :
        - Lancer la commande *“docker compose stop api”*
    - Pour redémarrer :
        - Lancer la commande *“docker compose restart api”*
    - Pour consulter les logs :
        - Lancer la commande *“docker compose logs -f api”*

LANCER UN RUN AIRFLOW

Cette procédure permet d’exécuter le pipeline complet de traitement des avis, que ce soit suite à une planification automatique ou dans le cadre d’une exécution manuelle.

Depuis Airflow :

1. Accéder à l'interface Airflow
2. Sélectionner le DAG *"review_pipeline"*
3. Cliquer sur *"Trigger DAG"*
4. Surveiller l'exécution (*"running"*, *"success"*, *"failed"*)
5. Si besoin, vérifier les logs (cf chapitre précédent)

Relancer une tâche :

1. Cliquer sur la tâche en erreur
2. Cliquer sur *"clear"*
3. Cliquer sur *"Confirmer"*

Airflow relancera la tâche automatiquement en tenant compte des dépendances.

Depuis Docker :

1. Lancer le run avec la commande *"docker exec -it airflow-scheduler airflow dags trigger review_pipeline"*
2. Vérifier les runs actifs avec la commande *"docker exec -it airflow-scheduler airflow dags list-runs -d review_pipeline"*

Validation :

L'exécution est considérée comme réussie si :

- toutes les tâches affichent un statut success;
- le fichier Parquet sur S3 est mis à jour;
- les données enrichies sont archivées dans MongoDB;
- l'API FastAPI peut lire le Parquet sans erreur.

VÉRIFIER LES LOGS AIRFLOW

La consultation des logs Airflow permet d'identifier l'origine d'un échec de tâche, de confirmer une exécution réussie ou de diagnostiquer un comportement anormal du pipeline. Cette procédure est essentielle dans le cadre du support quotidien et des interventions en cas d'incident.

Depuis Airflow :

1. Accéder à l'interface

2. Sélectionner le DAG *“review_pipeline”*
3. Cliquer sur la tâche dont on veut inspecter les logs
4. Cliquer sur *“Log”*

Depuis Docker :

- Scheduler : Lancer la commande *“docker compose logs -f airflow-scheduler”*
- Worker : Lancer la commande *“docker compose logs -f airflow-worker”*
- Webserver : Lancer la commande *“docker compose logs -f airflow-webserver”*

Les logs sont affichés directement dans l’interface et incluent :

- les opérations SQL (extraction depuis PostgreSQL),
- la progression de la transformation des données,
- les détails de la classification Zero-Shot,
- les opérations d’écriture S3,
- l’insertion dans MongoDB,
- les éventuelles erreurs ou exceptions.

Lors de la consultation des logs, vérifier notamment :

- Erreurs de connexion (PostgreSQL, S3, MongoDB)
- Erreurs de transformation (colonnes manquantes, valeurs invalides)
- Échecs de classification NLP (modèle non chargé, memory error)
- Échecs d’écriture dans S3 (droits IAM, fichier verrouillé)
- Échecs d’archivage (connexion MongoDB)

Actions en cas d’erreur :

- Erreur d’extraction PostgreSQL → vérifier la connexion, re-lancer task_extract_data
- Erreur S3 → vérifier les permissions, la clé IAM, la présence du bucket
- Erreur NLP → redémarrer le worker pour recharger les modèles
- Erreur MongoDB → vérifier que le conteneur MongoDB est actif
- Erreur de dépendance (Python) → redéployer l’image du worker

En cas d’erreur irrésolue → escalade au N2 / N3.

VÉRIFIER LA SANTÉ DE L'API

L'état de l'API peut être vérifié via le endpoint de santé, (en curl ou accès direct). La réponse attendue est : "code : 200, status: OK"

Si l'API ne répond pas ou avec un code d'erreur :

- vérifier que le conteneur api est actif avec la commande *"docker compose ps api"*
- consulter les logs avec la commande *"docker compose logs -f api"*
- s'assurer que le fichier Parquet sur S3 est disponible
- redémarrer le service si besoin avec la commande *"docker compose restart api"*

En cas d'erreur irrésolue → escalade au N2 / N3.

VÉRIFIER LE CONTENU DU DATA LAKE S3

Le fichier Parquet reviews_curated_XXX.parquet généré par le pipeline constitue la source unique pour l'API et le frontend. Il est donc essentiel de vérifier régulièrement que la donnée est correctement actualisée dans le bucket S3, constituant notre Data Lake.

Les étapes de vérification sont les suivantes :

1. Ouvrir le bucket configuré dans l'application (s3://…/datalake/).
2. Localiser le dernier fichier review_curated.parquet.
3. Vérifier la date et la taille du fichier (une date récente indique que le pipeline a tourné correctement , tandis qu'une taille anormalement faible peut signaler un problème d'extraction ou de transformation)
4. En cas de suspicion, relancer un run du DAG depuis Airflow.
5. Vérifier également la présence de fichiers de rejet (rejects/) ainsi que les logs, pour identifier de possibles anomalies dans le pipeline.

Une absence de mise à jour prolongée doit entraîner une analyse approfondie, car elle affecte la qualité et la véracité des informations servies au frontend.

PROCÉDURES PÉRIODIQUES

Afin d'assurer la stabilité opérationnelle du système, plusieurs actions de maintenance doivent être effectuées régulièrement.

Quotidiennes :

- Vérifier que le DAG Airflow s'est exécuté sans erreur.
- Contrôler que le fichier Parquet sur S3 est à jour.
- Vérifier la disponibilité de l'API (/health).
- Surveiller la consommation du Worker Airflow (mémoire / CPU).

Hebdomadaires :

- Examiner les logs Airflow et ceux de l'API pour identifier des anomalies récurrentes.
- Vérifier le volume des rejets ETL et analyser les tendances.
- Contrôler la volumétrie du bucket S3 pour détecter des croissances anormales.

Mensuelles :

- Purger les logs selon la politique de rétention définie.
- Vérifier que l'archivage MongoDB s'effectue correctement.
- Réaliser une revue des accès et permissions (RBAC).
- Mettre à jour les dépendances Docker si nécessaire.

Ces opérations participent à la pérennité et à l'efficacité du système en production, tout en prévenant des incidents futurs.

PLAN DE SUPPORT UTILISATEURS

Cette section décrit l'organisation du support ainsi que les modalités d'accès, de suivi et de résolution des incidents relatifs au système d'analyse des avis clients. Elle sert de référence aux équipes opérationnelles et aux utilisateurs techniques nécessitant une assistance.

TYPES D'UTILISATEURS

Le support couvre plusieurs profils d'utilisateurs, chacun avec un rôle défini :

- **Analystes / équipes métier** : ils consultent les données via l'API ou les outils BI connectés, et signalent tout dysfonctionnement d'accès ou de qualité de données.
- **Data Engineers / DataOps** : ils sont responsables de l'exécution des pipelines, du diagnostic technique et de la correction des erreurs liées à Airflow, S3, PostgreSQL ou MongoDB.
- **Administrateurs Systèmes / DevOps** : ils veillent à la disponibilité de l'infrastructure Docker, des services réseaux et des environnements d'exécution.
- **Administrateurs sécurité** : ils gèrent les accès, rôles et permissions, conformément au RBAC défini.

Chaque type d'utilisateur escalade les incidents selon le niveau d'intervention nécessaire.

MODALITÉS D'ACCÈS AU SYSTÈME

Les accès sont strictement définis selon le principe du "moindre privilège" :

- **API FastAPI** : accessible via HTTP/HTTPS. Certaines routes sont publiques (lecture seule), les autres nécessitent un token ou une permission spécifique.
- **Airflow** : accès restreint aux Data Engineers et DevOps pour consultation des DAG et des logs.
- **Data Lake S3**: accès en lecture uniquement pour l'API + lecture et écriture pour le pipeline Airflow + accès incident temporaire pour Data Engineers.
- **MongoDB** : accessible uniquement par le pipeline (écriture) et par les Data Engineers en cas d'investigation.
- **PostgreSQL** : accès lecture seule pour le pipeline, aucun accès utilisateur final.

Aucun accès ne doit être fourni sans validation explicite de l'équipe sécurité.

SLA (SERVICE LEVEL AGREEMENTS)

Pour garantir une exploitation stable, les niveaux de service suivants sont définis :

Disponibilité de l'API :

- Objectif : 99%
- Temps de réponse cible : < 150 ms pour l'endpoint /reviews/{id}/top

Pipeline Airflow :

- Exécution quotidienne garantie
- En cas d'échec, diagnostic sous 4 heures ouvrées

Données :

- Mise à jour quotidienne du fichier Parquet
- Temps maximal de retard accepté : 24 heures

Interventions :

- Correction d'incident mineur : sous 1 jour ouvré
- Incident majeur (API indisponible, pipeline bloqué) : sous 4 heures ouvrées
- Escalade vers niveau 3 si incident critique non résolu en 24 heures

GESTION DES TICKETS ET ESCALADE

Niveau 1 – Analyste / Support métier :

- collecte des symptômes, vérifie l'API via /health
- créé le ticket si reproduit

Niveau 2 – Data Engineer / DataOps :

- analyse Airflow, examine S3/MongoDB
- relance un run du DAG
- applique les correctifs simples

Niveau 3 – DevOps / Administrateur système :

- intervention sur Docker, réseau, ressources machine
- problèmes liés à la disponibilité globale du système

Procédure d'escalade :

- Si l'incident persiste après deux tentatives au niveau 2 → escalade niveau 3
- En cas d'incident critique (API down, données indisponibles) → escalade immédiate
- Le ticket est clôturé uniquement après validation par le demandeur

SUPERVISION & MONITORING

La supervision du système garantit le bon fonctionnement du pipeline d'analyse des avis, de l'API et des environnements de stockage. Elle permet de détecter précocement les anomalies et d'assurer la continuité de service.

OUTILS DE MONITORING

La supervision repose sur une combinaison de sources de logs et d'outils permettant d'assurer une visibilité complète sur l'état du système.

- **Logs Airflow** : point de référence pour analyser l'exécution des DAG.
- **Logs Docker** : essentiel pour diagnostiquer les erreurs de conteneurs.
- **Logs FastAPI** : permettent d'identifier les exceptions, erreurs S3 ou problèmes de lecture du Parquet.

Dans le futur**, Splunk** viendra centraliser les logs provenant d'Airflow, de l'API FastAPI et des conteneurs Docker. Les recherches programmées et tableaux de bord Splunk permettront d'identifier rapidement les erreurs et d'automatiser l'alerting. Il joue le rôle de plateforme unique d'observation, consolidant les logs et fournissant des alertes automatisées.

Splunk peut également servir de base d'audit en retraçant les accès, anomalies et comportements inhabituels sur Airflow, API et Docker.

MÉTRIQUES SUPERVISÉES

Les métriques supervisées sont les suivantes :

Pipeline Airflow :

- Statut des tâches (succès/échec)
- Temps d'exécution par étape
- Volume de rejets ETL
- Nombre de relances nécessaires

API FastAPI :

- Disponibilité (/health)
- Latence moyenne
- Taux d'erreurs (4xx / 5xx)
- Temps de lecture du fichier Parquet S3

Stockage S3 (Data Lake) :

- Date de dernière mise à jour du fichier Parquet
- Taille du fichier (détection d'anomalies)
- Volumétrie totale du bucket

MongoDB :

- Volume d'insertion
- Erreurs d'écriture

DASHBOARD PROPOSÉS

Plusieurs tableaux de bord peuvent être configurés dans Splunk afin de faciliter la supervision :

Dashboard 1 : Pipeline Airflow

- Statut des derniers runs du DAG
- Historique des durées d'exécution
- Taux de rejets ETL

Dashboard 2 : API

- Disponibilité temps réel
- Graphique du taux d'erreurs et exceptions
- Latence par endpoint

Dashboard 3 : Stockage et archivage

- Historique de mise à jour du Parquet
- Évolution de la volumétrie S3
- Suivi du nombre de documents archivés dans MongoDB

Ces tableaux de bord garantissent une visibilité immédiate sur la santé globale du système.

ALERTES ET SEUIL

Des alertes sont configurées dans Splunk via des "recherches programmées" utilisant des seuils définis :

Alertes critiques :

- API indisponible (/health en erreur plusieurs fois)
- DAG Airflow en échec
- Fichier Parquet non mis à jour depuis plus de 24 h
- Taux d'erreurs API > 10 % sur 10 minutes
- Rejets ETL > seuil défini
- Erreurs répétées de connexion S3 ou MongoDB
- DAG Airflow en échec persistant après 3 retries automatiques

Alertes d'attention :

- Durée d'exécution du DAG supérieure à la moyenne
- Croissance anormale du fichier Parquet
- Latence API > seuil nominal

Les alertes peuvent être transmises automatiquement aux équipes opérationnelles par

exemple par mail pour une intervention rapide.

🏁 CONCLUSION

Ce dossier d'accompagnement présente l'ensemble des éléments nécessaires à l'exploitation du système d'analyse et de classification des avis clients : procédures de lancement, vérification du pipeline Airflow, diagnostic via les logs, supervision de l'API et gestion des stockages. Il fournit également un cadre clair pour le support, la gestion des incidents et le maintien en condition opérationnelle.

Grâce à ces procédures et à la centralisation de la supervision via Splunk, les équipes techniques disposent d'un environnement fiable, traçable et simple à administrer. Ce document constitue enfin la base de référence pour les futures actions de maintenance évolutive détaillées dans le dossier suivant.