

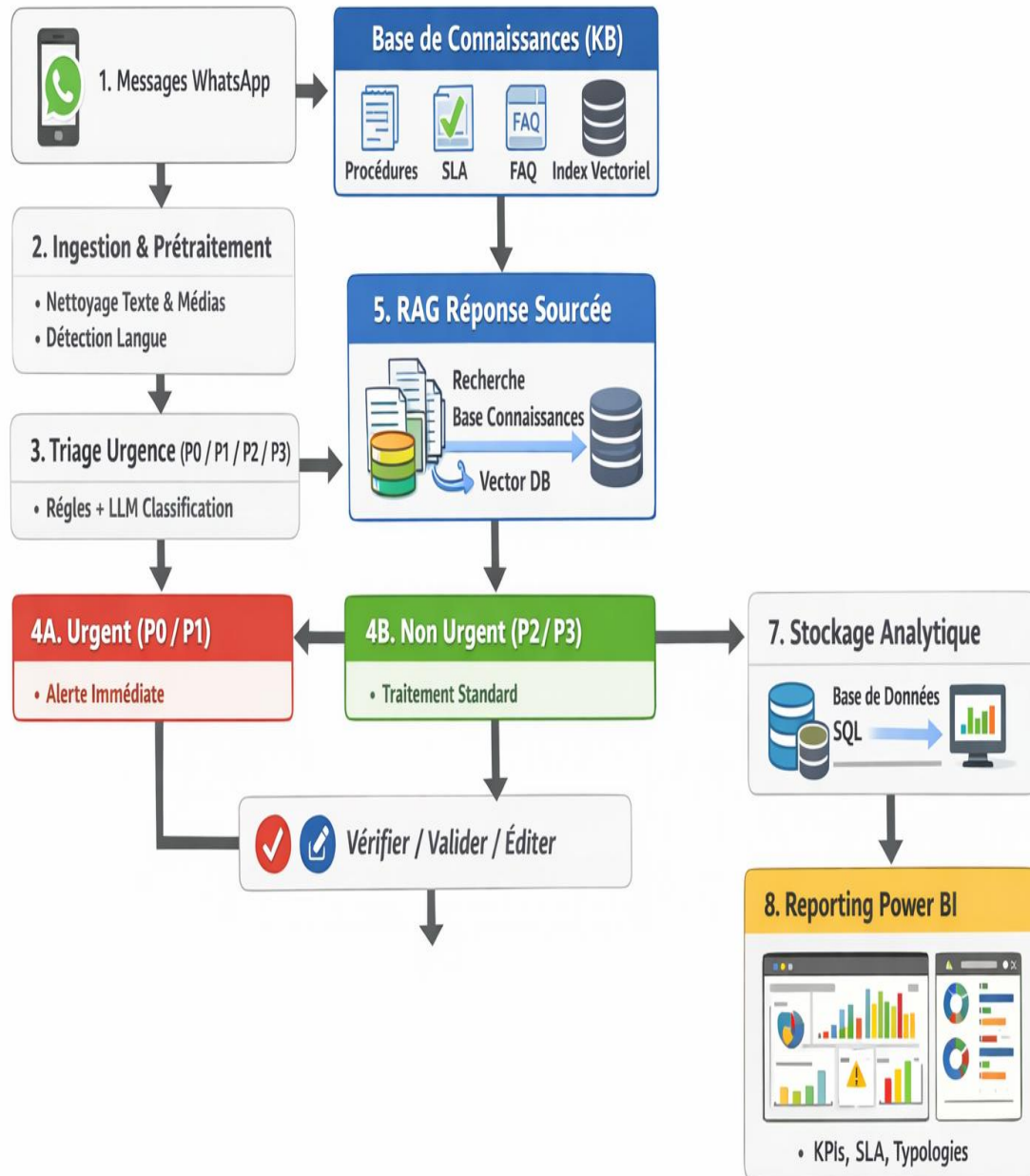
**Contexte :**

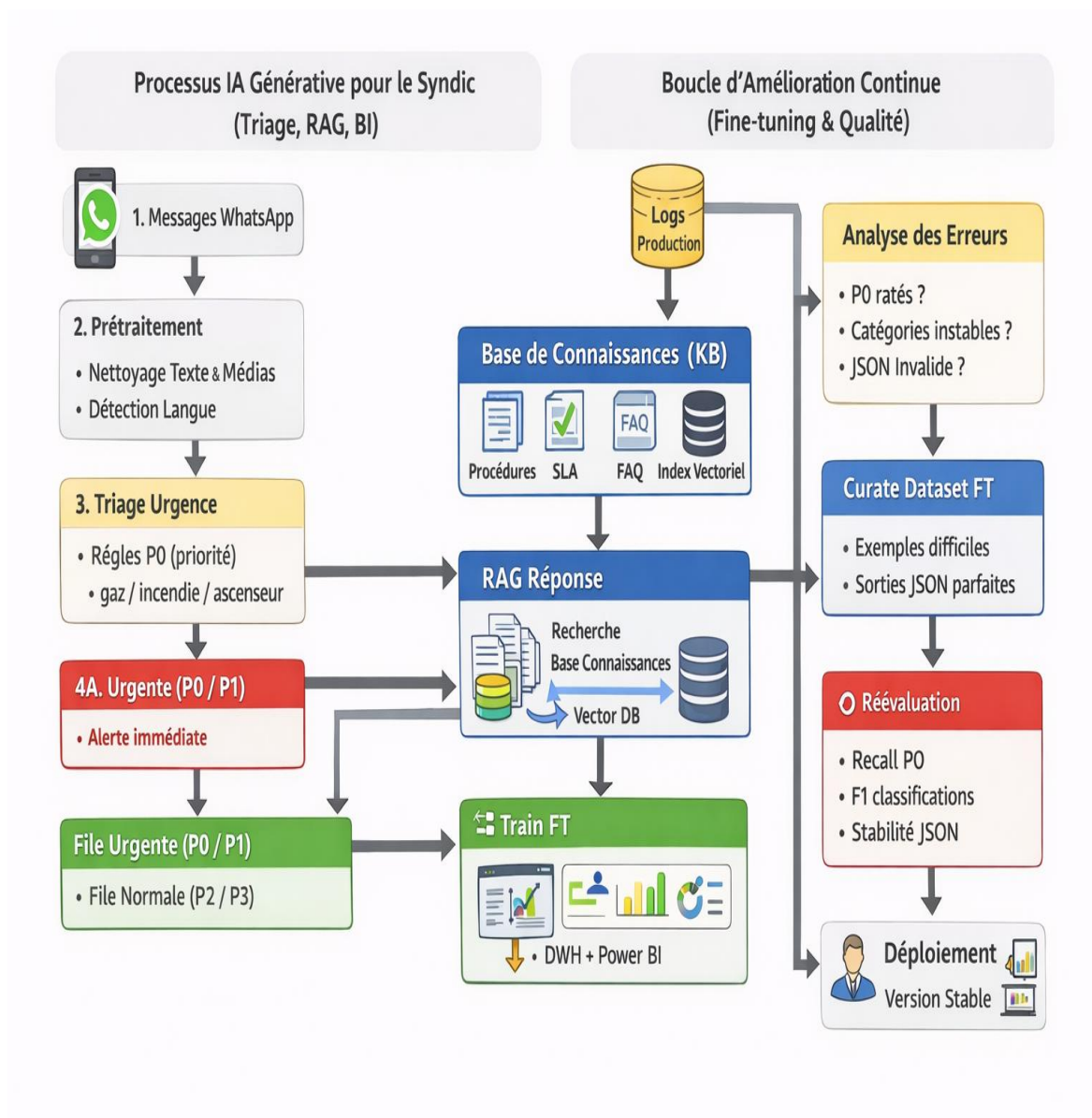
Une société de syndic reçoit chaque jour des messages WhatsApp de résidents (plaintes, incidents, demandes administratives). L'objectif est de construire un **système intelligent** qui :

1. **lit** les messages,
2. **classe l'urgence** en **P0 / P1 / P2 / P3**,
3. **génère une réponse** prête à être **validée** par le syndic,
4. **produit des reporting et tableaux de bord Power BI** (volumes, typologies, délais, satisfaction).

**Objectifs :**

1. **Collecte WhatsApp** (prototype : export .txt/.json ; production : WhatsApp Business API / fournisseur)
2. **Pré-traitement** (nettoyage, anonymisation, langue, déduplication, pièces jointes)
3. **Classification & routage** (urgent ? catégorie ? résidence ? priorité ? sentiment ?)
4. **RAG** (recherche dans les procédures/contrats/règlement/SLA + génération de réponse)
5. **Human-in-the-loop** (le syndic valide/édite/renvoie)
6. **Stockage analytique** (SQL/Data Warehouse)
7. **Power BI** (dashboards : volumes, urgences, SLA, typologies, satisfaction...)





Étapes :

Étape 1.1 — Définir les “types d’urgences” (règles métier)

Créer une politique simple (que l’IA doit suivre) :

- **Urgence critique (P0) :** incendie, fuite de gaz, inondation majeure, ascenseur bloqué avec personne, agression...
- **Urgence (P1) :** panne d’eau générale, panne électrique parties communes, porte garage bloquée, fuite importante...
- **Non urgent (P2/P3) :** bruit, réclamation propreté, demande attestation, question charges...

## Étape 1.2 — Construire un dataset WhatsApp (réaliste)

Créer un CSV (ou Parquet) `messages_raw` :

- `message_id`, `datetime`, `resident_id`, `residence_id`, `channel` (whatsapp)
- `text`, `has_media`, `media_type`, `language`

## Étape 1.3 — Schéma de sortie IA (JSON) = “contrat”

Définir un JSON strict

### Entrées :

- `messages.csv` (déjà collecté)

1) Définir la policy d’urgence

Créer/valider une grille simple :

- **P0** : incendie, gaz, inondation majeure, ascenseur bloqué avec personne, agression
- **P1** : panne d’eau générale, panne électrique parties communes, porte garage bloquée, fuite importante
- **P2/P3** : bruit, propreté, administratif (attestation, charges, quittance...)

2) Prétraitement rapide du CSV

Dans Python/Power Query (au choix) :

- supprimer lignes vides
- normaliser `text` (trim, espaces)
- créer `text_clean` (optionnel)
- ajouter `has_media` (si présent) sinon 0
- vérifier encodage UTF-8

**Sortie** : dataset propre prêt pour classification.

3) Baseline “règles garde-fou” (P0 d’abord)

Implémenter des règles simples:

- Si texte contient **gaz / odeur de gaz** → P0
- Si **incendie / fumée / flammes** → P0
- Si **ascenseur** + (**bloqué/coincé**) + (**personne/à l’intérieur**) → P0
- Si **agression/bagarre** → P0
- Si **inondation** + (majeure / “partout” / “monte”) → P0

**But :** ne pas rater P0.

#### 4) Classification via LLM (JSON strict)

- appeler le LLM sur chaque message (ou batch)
- valider JSON (sinon retry 1 fois)
- cohérence : P0/P1  $\Rightarrow$  is\_urgent=1
- ajouter sla\_target\_minutes via table (P0=5, P1=30, P2=240, P3=1440)

#### Mini reporting “pré-Power BI”

- % urgents
- top catégories
- messages par résidence
- messages par jour

#### Indexation RAG

Étapes :

1. extraire texte docs
2. chunking (300–800 tokens + overlap)
3. embeddings
4. index vectoriel (FAISS/Chroma)
5. retriever top-k (k=5)

#### Génération de réponse “validable” (RAG)

Pour chaque message (avec urgency\_level, category déjà calculés) :

1. construire la requête de recherche (query rewrite simple)
2. récupérer top-k chunks (preuves)
3. générer réponse avec un prompt qui renvoie JSON

**Règles :**

- si preuves insuffisantes  $\rightarrow$  demander info (required\_info) et rester prudent
- si P0/P1  $\rightarrow$  inclure consigne d’escalade + collecte d’infos minimales
- réponse courte, pro, actionnable (temp=0.1 top\_p=0.1)

Ajouter une colonne :

- status = TO\_VALIDATE par défaut
- (option) assigned\_to = Syndic / Prestataire / Gardien

Sortie Power BI

Dans Power BI :

1. Importer `messages_final.csv`
2. Créer mesures rapides :
  - Nb messages, % urgents, nb P0, nb P1
  - Top catégories
  - Tendence par jour/sem
  - SLA breach (si vous avez délais, sinon uniquement SLA cible)