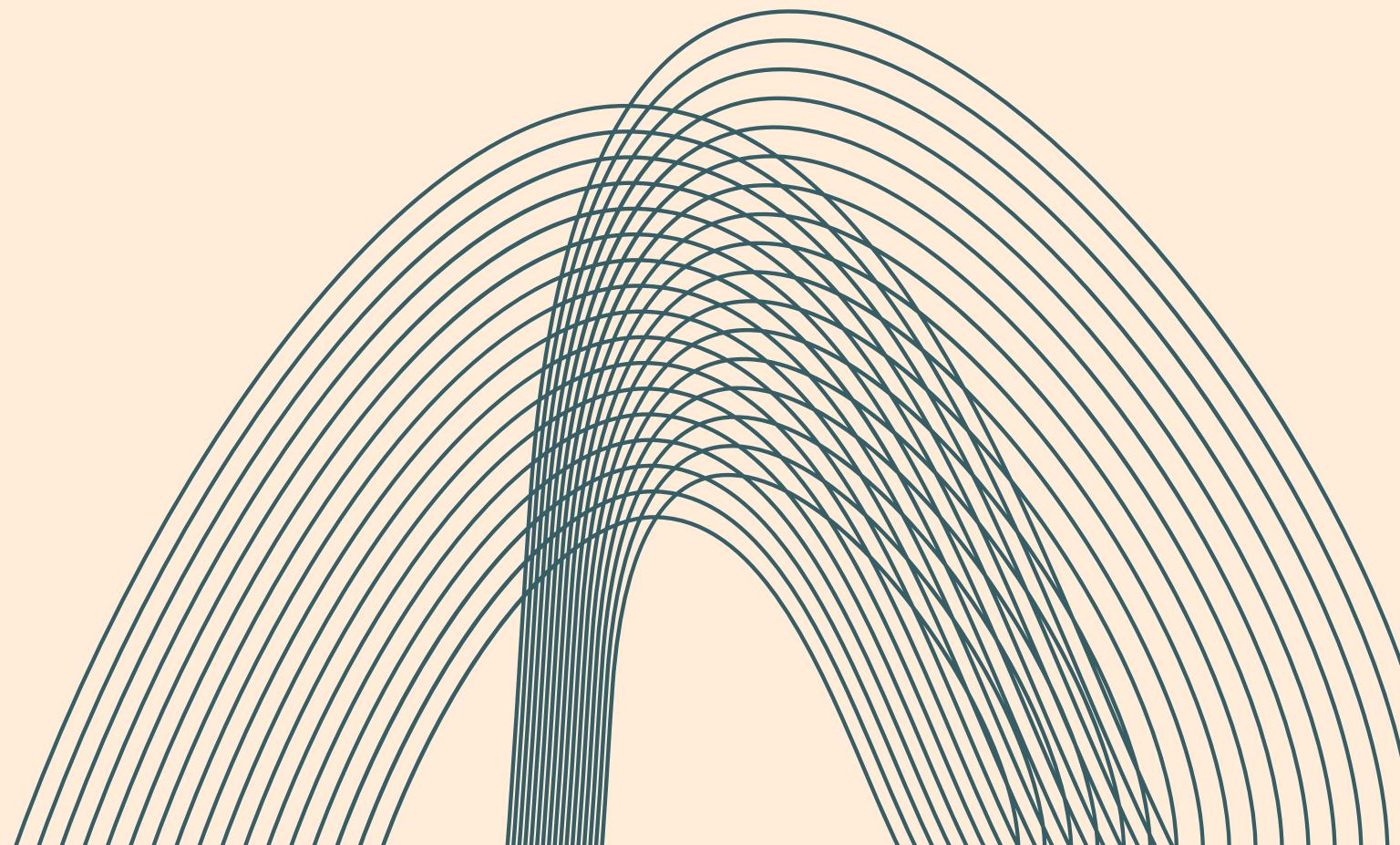
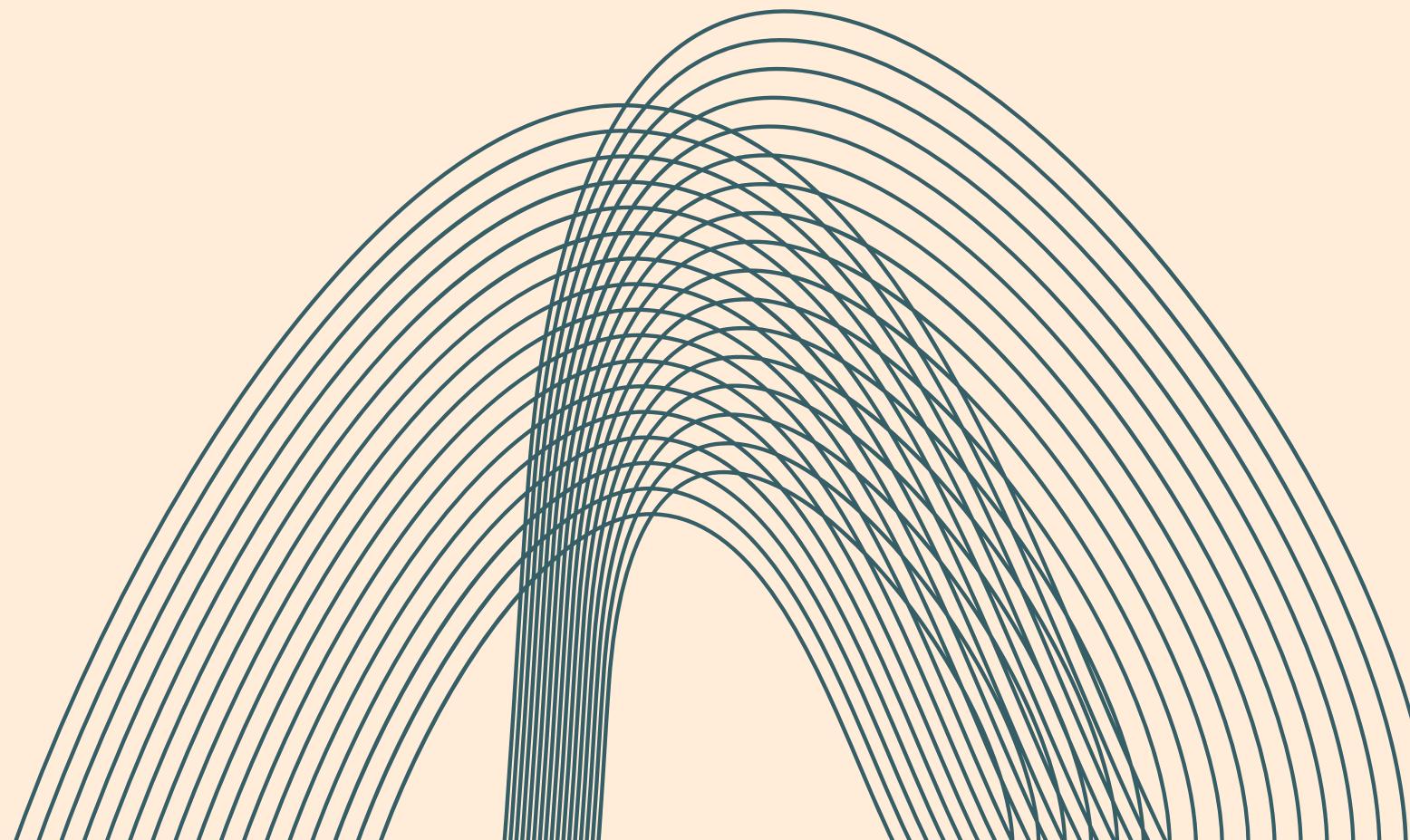


Digital Twin : De la donnée simulée au rapport RAG

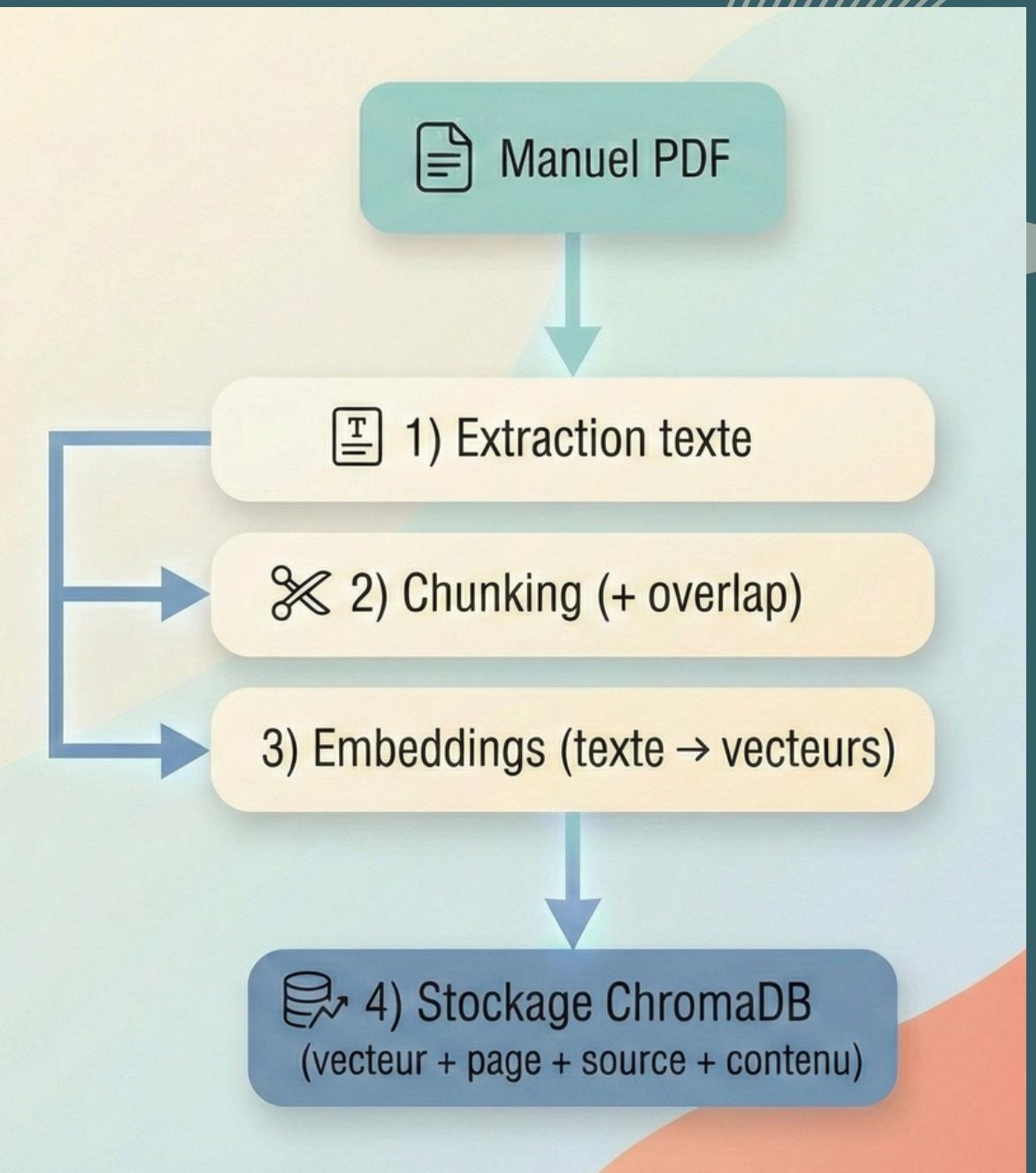


Concepts clés (définitions)



Ingestion

Processus automatisé qui transforme des données non structurées (PDF) en format exploitable. Il se compose de quatre étapes clés : extraction du contenu brut, segmentation en chunks, vectorisation (embedding) et indexation dans la base de données vectorielle (ChromaDB).





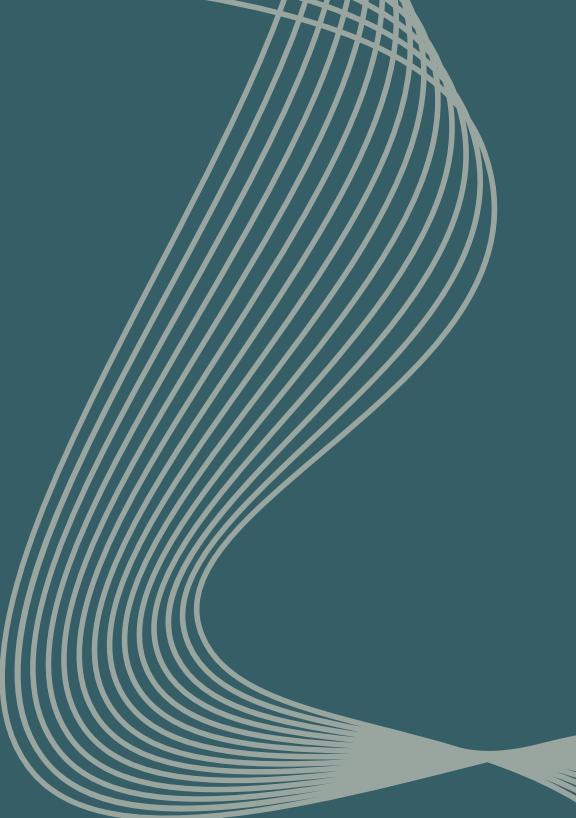
Chunk (Segment)

Unité logique de texte résultant du découpage du document original. Sa taille est calibrée pour maximiser la cohérence sémantique et respecter la fenêtre de contexte du LLM. Chaque chunk est associé à des métadonnées (source, numéro de page) pour assurer la traçabilité.



Embedding (Plongement vectoriel)

Représentation d'un texte sous forme d'un vecteur numérique de haute dimension. Cette transformation capture la signification sémantique du texte plutôt que ses mots-clés exacts, permettant de calculer mathématiquement la proximité de sens entre une requête et un document.



Vector DB (ChromaDB)

Base de données spécialisée dans le stockage et l'indexation de vecteurs. Elle utilise des algorithmes de recherche de plus proches voisins (comme ANN) pour identifier quasi-instantanément les segments les plus sémantiquement proches d'une requête utilisateur.

RAG

Retrieval (Récupération) : L'utilisateur pose une question (ou le système détecte une panne). Le système effectue une recherche sémantique dans votre base de données vectorielle (ChromaDB) pour trouver les passages du manuel technique les plus pertinents.

Augmentation (Augmentation) : Le système prend la question de l'utilisateur et y "colle" les passages trouvés dans le manuel. On enrichit (augmente) le contexte.
Prompt envoyé au LLM : "Voici le contexte : [Extraits du manuel]. En utilisant ce contexte, réponds à la question : [Question utilisateur]".

Generation (Génération) : Le LLM reçoit ce prompt enrichi. Il génère une réponse en langage naturel en utilisant les faits précis qu'on vient de lui fournir, sans avoir besoin de les connaître par cœur.





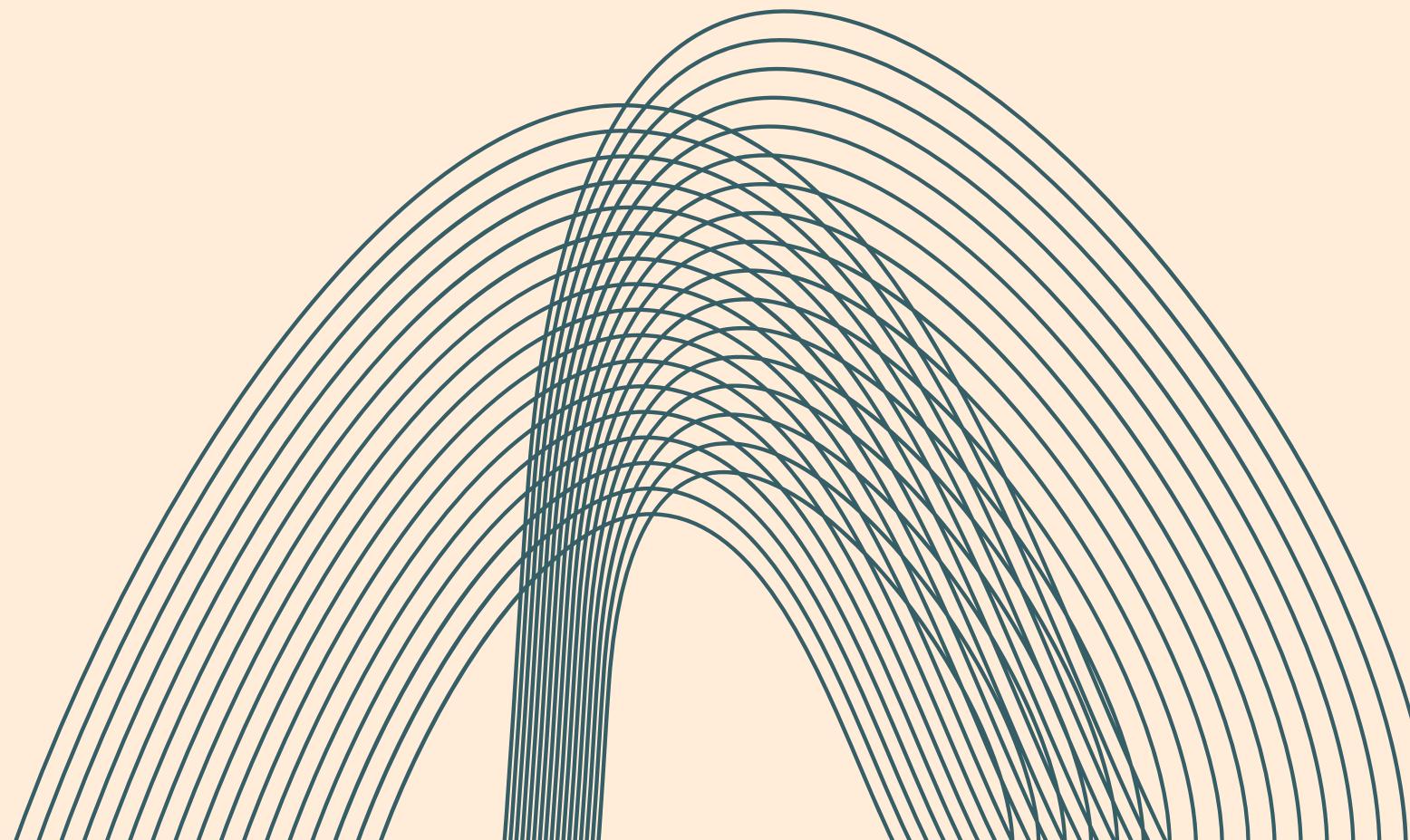
Autre Définitions :

Prompt : entrée structurée du LLM (rôle + données + contexte + tâche).

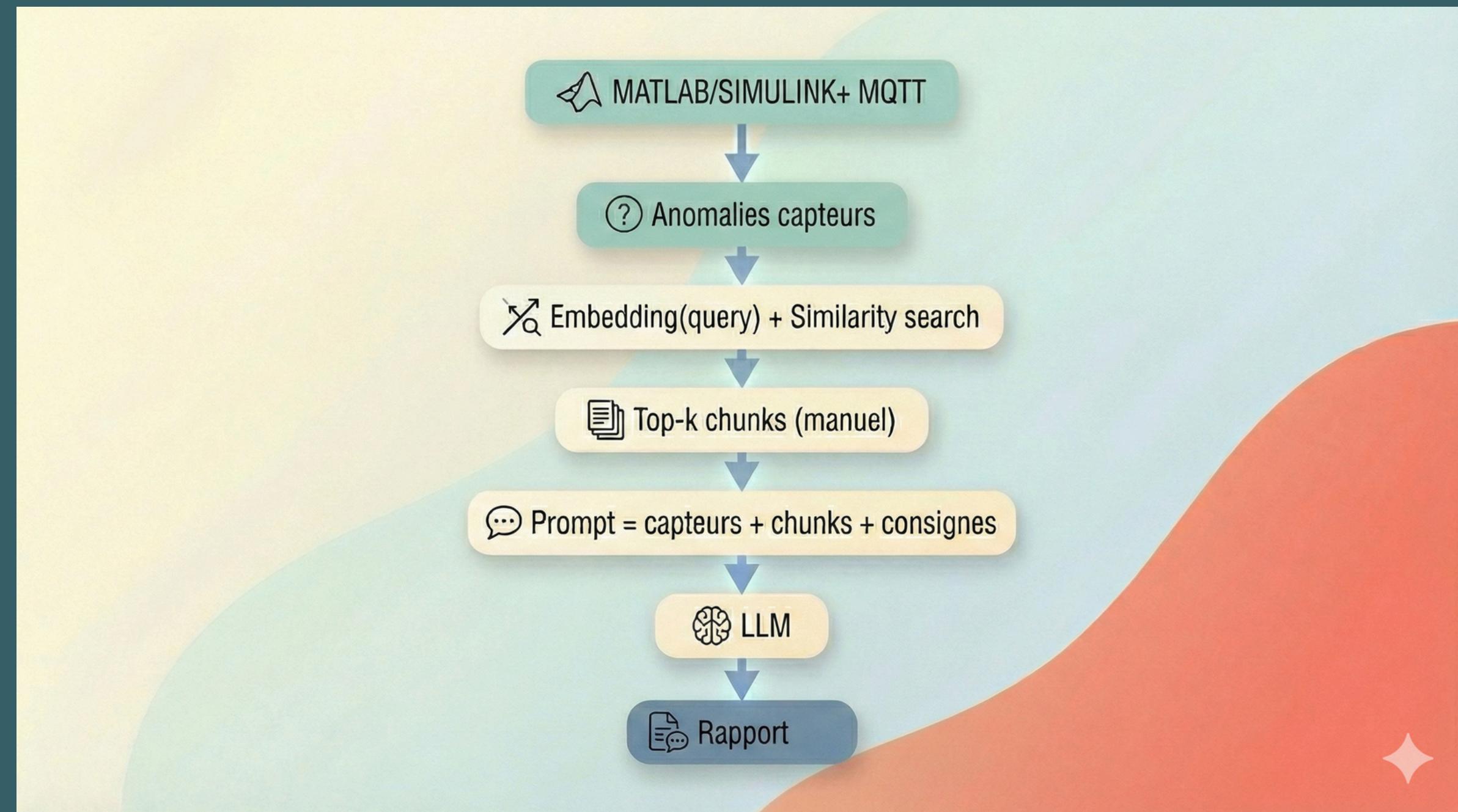
Guardrails: filtres qui limitent le chatbot aux sujets maintenance.

Semantic memory: notes long-terme (préférences/faits) persistées et réinjectées au chat.

Generation de rapport



Pipeline Rapport



Entrée du pipeline : données capteurs

- Source : MQTT (MATLAB/Simulink publie la télémétrie, le backend lit le dernier snapshot)
- Format : courant 3 phases + déséquilibre, tension, vibration, pression, température, fault_state + durée
- Ces données deviennent le « contexte temps réel » de l'IA

Transformation : capteurs → RAG query

Objectif : convertir des valeurs brutes en requête “texte” proche du vocabulaire du manuel.

- Exemple de règles (anomalies → mots-clés) :
déséquilibre courant > 5% → `motor winding defect phase imbalance`

Résultat : une requête courte, informative, “search-friendly”.

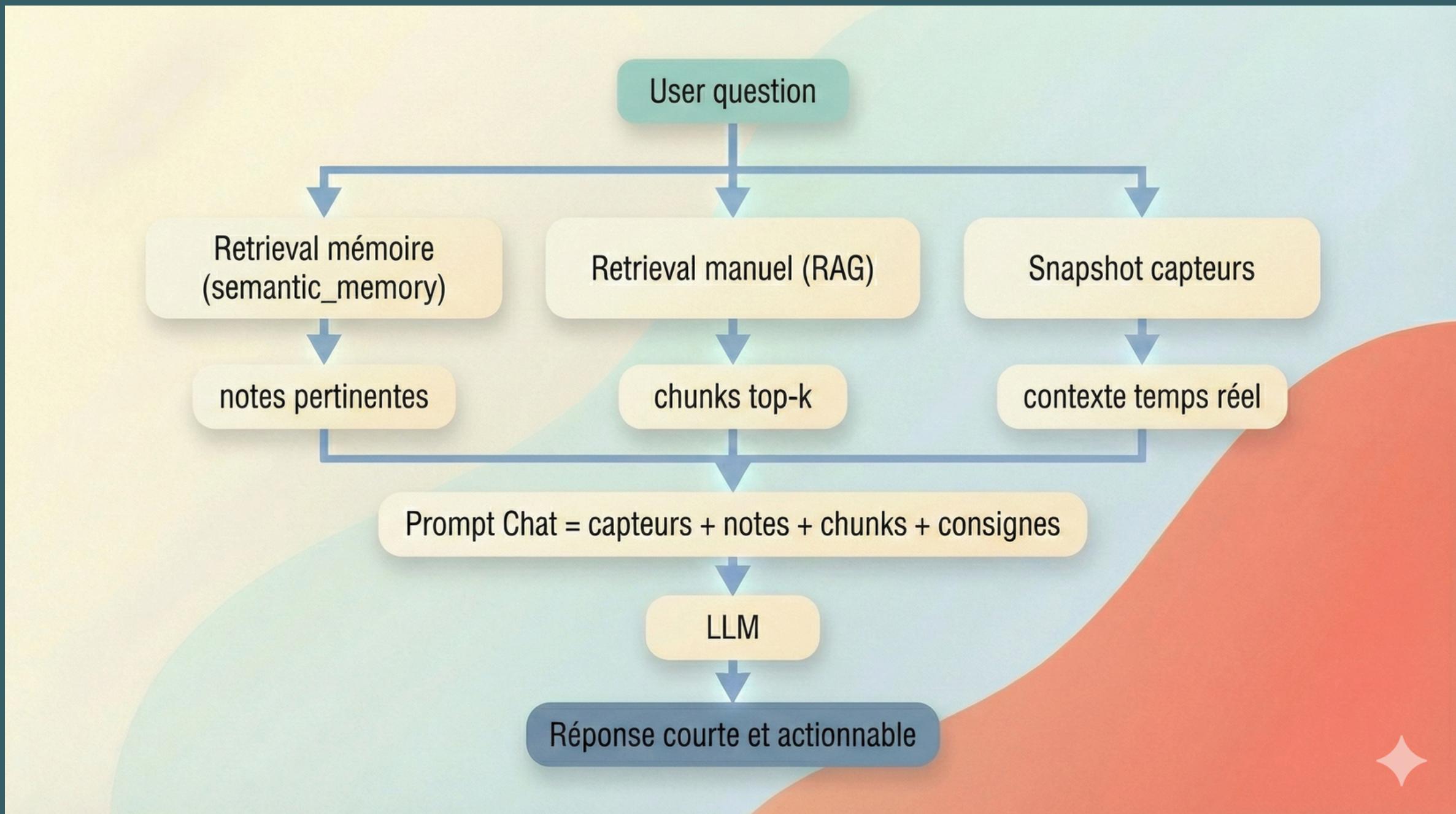
Recherche dans ChromaDB (Vector Search)

On calcule l'**embedding** de la query (Google Embeddings)

- ChromaDB fait **une similarity search** dans la base du manuel
- On récupère top_k extraits (chunks) :
 - content (texte)
 - page (référence)
 - score (pertinence)

But : fournir au LLM des preuves / procédures issues du manuel.

Pipeline ChatBot



Chatbot : contexte + mémoire

Le chatbot combine :

- Statut courant (snapshot capteurs)
- Contexte manuel (chunks RAG)
- Contexte événementiel (début du défaut)
- Mémoire long-terme (semantic memory : préférences/faits)

Chatbot : contexte + mémoire

Guardrails : rester “maintenance”

- Filtre 1 (règles) : refuser si aucun signal maintenance (pompe, moteur, vibration, cavitation, défaut, test...)
- Filtre 2 (sémantique) : refuser si similarité trop faible avec des intentions “maintenance”
- Fallback : réponse de cadrage

Chatbot : contexte + mémoire

User question

|
└ (1) Keywords maintenance ? — non → Refus + recadrage

|
└ oui

|
└ (2) Similarité sémantique ok ? — non → Refus

|
└ oui → RAG + LLM

