Recommandation Technique – Architecture RAG pour l'Assistant IA SKILLIA

1 Recommandation de l'architecture RAG

Nous recommandons de mettre en œuvre une **architecture RAG hybride**, combinant :

- **LangChain** : pour l'orchestration intelligente du pipeline de traitement, incluant agents, retrieval multi-sources, mémoire conversationnelle et enchaînement logique (scraping, résumé, génération, formatage).
- **Azure AI Search** : pour l'indexation et la recherche vectorielle/sémantique de documents (veille, benchmarks, offres concurrentes), intégrée à l'écosystème Azure.
- **Azure OpenAI (GPT-4)** : pour la génération de contenus stratégiques adaptés aux contextes métier, avec une qualité rédactionnelle élevée.

Cette combinaison permet de :

- Répondre aux besoins complexes du générateur d'offres sur mesure SKILLIA (veille, benchmark, génération, export).
- S'appuyer sur une solution **scalable, modulaire et sécurisée**, conforme aux exigences d'un environnement entreprise.
- Faciliter l'évolution du système vers des agents intelligents et une orchestration avancée.

2 Pourquoi cette architecture plutôt que les autres ?

1. Comparaison avec un RAG Classique

Limites du RAG Classique (Two-Step) :

- Une seule base documentaire souvent limitée.
- Absence de raisonnement ou de logique métier.
- Chaîne retrieve → generate trop rigide pour des tâches complexes.

Avantages du RAG Hybride :

- Capacité à combiner plusieurs retrievers (vectoriel + sémantique).
- Orchestration intelligente des étapes (veille → résumé → génération).
- Adaptation dynamique au profil et au secteur client.

2. Comparaison avec le RAG Fusionné (End-to-End Learning)

Limites du RAG Fusionné (ex. FiD) :

- Coût élevé d'entraînement.
- Manque de flexibilité pour ajouter des sources externes.
- Raisonnement opaque et difficile à adapter sans re-entraînement.

Pourquoi le RAG Hybride est préférable :

- Compatible avec des sources multiples et évolutives.
- Facile à déboguer et transparent (chaque étape est contrôlable).
- Moins coûteux et plus agile pour un POC ou MVP.

3. Comparaison avec Snowflake Cortex AI

Limites :

- Ciblé pour les cas analytiques, pas NLP avancé.
- Peu d'orchestration ou d'extension possible.

Pourquoi Azure + LangChain est meilleur :

- Azure AI Search + OpenAI = combinaison native vectorielle + LLM.
- LangChain permet un vrai raisonnement et des agents intelligents.

4. Comparaison avec Databricks + MLflow

Limites :

- Nécessite de la configuration complexe (DevOps).
- Plus adapté aux pipelines de ML/data science qu'au NLP orienté métier.

5. Comparaison avec LangChain seul (open-source local)

Limites :

- Sécurité, scalabilité, persistance manuelles à implémenter.
- Besoin d'héberger sa propre base vectorielle (FAISS, Weaviate, etc.).

Pourquoi Azure AI Search est préférable :

- Solution clé en main, sécurisée, cloud-native.
- Facilité d'intégration avec LangChain et Azure OpenAI.