**But** : tu donnes un sujet → l’agent explore news + publications + contenus “signal”, déduplique, évalue la qualité, et rend un **rapport structuré** (résumé, tendances, bibliographie annotée, gaps, next steps).

**Scope & sorties**

* **Rapport auto-généré** (Markdown/PDF) avec :
  + TL;DR (10–15 lignes)
  + Panorama rapide (3–5 points clés, acteurs, dates)
  + **Publications** (arXiv / OpenAlex / Semantic Scholar / PubMed) : 10–20 refs annotées (contribution, limites, liens)
  + **News** (RSS/NewsAPI/Guardian API) : 10 items notables (source, biais potentiel, date)
  + “**Signals**” (brevets / blogs tech / GitHub trending) si dispo
  + **Gaps & pistes** (questions de recherche + idées POC)
  + Bibliographie formatée (BibTeX/CSL)
* **Annexes** : graphe de citations/co-occurrences, timeline, mind-map auto.

**Architecture (modulaire, robuste)**

1. **Planner** (agent chef)
   * Décompose le sujet en sous-thèmes, génère des requêtes ciblées, priorise les sources.
   * Stratégies : ReAct + Reflexion (boucles courtes) + “toolformer” pour choisir les outils.
2. **Collectors (multi-agents)**
   * **Papers** : arXiv API, Crossref/OpenAlex, Semantic Scholar.
   * **News** : NewsAPI/RSS (sources whitelisées), date filtering, dedup (hash de titre+url+date).
   * **Brevets/Code** : Google Patents (scrape propre si autorisé) / GitHub search API.
3. **RAG & Mémoire longue**
   * Ingestion → chunking → embeddings (par ex. bge-m3 / E5) → **pgvector** (Postgres).
   * Enrichissement : extraction d’entités (NER), thèmes, scores de crédibilité (voir plus bas).
4. **Ranking & Qualité**
   * Score = f(pertinence sémantique, fraîcheur, **qualité source** (journal/revue factor), signaux sociaux (citations/étoiles), **détection d’infox** basique (liste noire, cross-source)).
   * Anti-doublons, clustering par thème (HDBSCAN).
5. **Writer**
   * Génère le rapport **contrôlé par gabarit** (templates Jinja), citations numérotées, **chaînes vérifiées** (retrieval-augmented, with inline refs).
   * Post-édition : un **Critic agent** checke factualité vs sources (vérifications pointées).
6. **Governance/Observability**
   * Logs structurés, traces (OpenTelemetry), coût/latence, “hallucination rate” proxy (réponses sans sources → rejet/relance).

**Stack conseillée**

* **Orchestration** : LangGraph (ou CrewAI), FastAPI, Celery/Arq pour jobs.
* **Data** : Postgres + **pgvector**, Redis (queues/cache).
* **IR/RAG** : llama-index / Haystack (peu importe, mais stable), FAISS/pgvector.
* **LLM** : un modèle instruct cost-aware (mix open/closed selon budget), + un plus petit pour critiques.
* **Front** : Next.js (export PDF via md → pandoc).
* **Obs** : Prometheus + Grafana, OpenTelemetry.

**Évaluation (académique)**

* **Pertinence** (NDCG@k / MRR vs un “oracle” annoté sur quelques sujets).
* **Couverture** (ratio sources uniques + diversité éditeurs).
* **Factualité** (pourcentage d’affirmations sourcées, taux d’invérifiables).
* **Utilité perçue** (Likert par utilisateurs métiers).
* **Coût/latence** (€/rapport, s/rapport).
* **Ablations** : sans critic, sans ranking, sans RAG, sans anti-doublons.

**Risques & garde-fous**

* **Hallucinations** → “no-source = no-claim”.
* **Biais sources** → diversité + pondération.
* **Copyright** → stocker **extraits** + métadonnées, pas les PDFs complets si non permis.
* **Confidentialité** (si sources internes) → silo séparé, chiffrement.

**Livrables**

* Repo GitHub (infra as code + docker-compose), README avec schéma, **dataset d’éval** (requêtes + jugements), **rapport demo** (ex : “LLM agents for supply chain”).
* **Notebook** d’ablation & métriques, **Dashboard** observabilité.

**Roadmap (MVP → V1)**

* S1 : collectors + RAG + template rapport.
* S2 : ranking qualité + critic + obs.
* S3 : éval & ablations, UX, packaging.