# Choix structurants

# Choix structurants

> *Dernière mise à jour : 2026-05-10*

Liste rapide des décisions d'architecture qui définissent le projet. Chaque entrée pointe vers son ADR détaillé dans le chapitre **Adrs**.

| Choix | Pourquoi | ADR |
|---|---|---|
| **Hono** plutôt qu'Express/Fastify | SSE natif, perfs, types meilleurs, moins de middleware tiers | ADR-001 |
| **SQLite + Drizzle** plutôt que Postgres | Self-host minimal, zéro service supplémentaire, ACID suffisant pour l'usage actuel | ADR-002 |
| **Claude via SSH** plutôt que API Anthropic directe | Quota Pro/Team du compte personnel (vs API à crédits), accès à l'outil `Read` de Claude Code pour la vision PDF, no-coût marginal sur appels | ADR-003 |
| **Vision inline** au moment de la question, pas pendant l'ingestion | Coût zéro à l'ingestion, lecture PNG ciblée 1 page max au moment opportun, latence acceptable (~30s) | ADR-004 |
| **RAG Fusion v2** (pondération question×2 + blending position-aware) | Préserve les exact-matches de la question brute contre la dilution HyDE, et garde le signal RRF quand le reranker est peu confiant sur du contenu technique abstrait | ADR-005 |
| **Repository pattern** (Phase 2) | Centralisation des requêtes Drizzle dans `repositories/`, jamais d'accès direct depuis routes/services/cron | ADR-006 |
| **Handlers MVC** (Phase 4) | Séparation routes (HTTP) → handlers (logique pure) → services → repos. Result discriminés pour les erreurs attendues | ADR-007 |

# Autres décisions notables (sans ADR formel)

- **TEI bge-m3 plutôt qu'Ollama mxbai-embed-large** : initialement Ollama, switch vers TEI pour les perfs (GPU dédié + serving optimisé HuggingFace) et le support BM25 sparse natif Qdrant.
- **Contextual Retrieval B** plutôt qu'A : full-LLM par chunk avec document complet + position hiérarchique. Plus cher mais qualité max sur les règles de jeux où le contexte est critique.
- **Pas de Helmet Hono** : tous les headers sécurité (HSTS, CSP, X-Frame-Options) sont gérés par NPM en amont. Évite la duplication.
- **CORS whitelist + CIDR LAN** : support des origines exactes ET des plages IPv4 ( `192.168.10.0/24` ) pour autoriser le subnet local sans exposer publiquement.
- **`hasCardDatabase`** : colonne nullable sur `games` qui pointe vers la collection Qdrant correspondante ( `magic-cards` , etc.). Permet d'activer dynamiquement l'autocomplete `@card` , le deck import, le mode deckbuilding par jeu.
- **Sticky mentions** plutôt que long context window : on borne explicitement (cap 20 ou 80 si deck) au lieu de balancer tout l'historique. Maîtrise des tokens, prévisibilité du coût.
- **Logger central + env vars centralisées** ( `config.ts` ) : aucun `console.*` ni `process.env.*` ailleurs. Garantit que les logs sont filtrables et que les vars sont validées au boot.

---

Revision #1
Created 2026-05-10 15:20:28 UTC by thymon
Updated 2026-05-10 15:20:28 UTC by thymon