IntelliSense AI – Hybrid Storage Agentic RAG Architecture

Core Principle

Store knowledge cheaply, store vectors selectively, retrieve hierarchically.

High-Level Layers

1. Vector Knowledge Index (Small, Fast)

2. Document Storage (Large, Cheap)

3. Metadata Knowledge Index (Document Mapping Layer)

Layer 1 — Vector Knowledge Index

Purpose: Fast semantic search to locate concept pointers rather than full knowledge. Stores: Important chunks, summaries, core concepts, high-frequency exam topics, topic-to-document pointers. Does NOT store: Full books or raw documents. Optimization: 384-dim embeddings, deduplication, compressed chunks, selective embedding.

Layer 2 — Document Storage

Purpose: Store complete academic library cheaply. Stores: PDFs, books, notes, PPT text, question papers, full extracted text. No embeddings stored here. Used for deep retrieval when needed.

Layer 3 — Metadata Knowledge Index

Purpose: Connect vector hits to exact document location. Fields include: doc_id, subject, topic, subtopic, page/section, importance_score, vector_chunk_id, storage_pointer.

Hierarchical Retrieval Pipeline

Step 1: Coarse semantic retrieval from Vector DB to identify relevant concept or document. Step 2: Fine retrieval using metadata to load exact paragraph/page from document storage. Step 3: Re-ranking based on semantic similarity, keyword overlap, and claim alignment.

Agentic MCP Access Layer

Agent tools include: - fetch_document_section(doc_id, page) - search_metadata(topic) - retrieve_vector_hits(query) Flow: Query → Vector pointer → Agent fetches document section → Verification → Explanation with evidence.

Knowledge Compression Pipeline

Raw Documents → Text Extraction → Deduplication → Noise Removal → Chunking → Importance Scoring → Selective Embedding (Vector DB) → Full Text Storage → Metadata Entry.

Storage Optimization Policies

Embed only high-value knowledge. Remove rarely used vectors. Use 384-dim embeddings to significantly reduce storage footprint.

Caching Layer

Cache frequent retrieval results, popular evidence chunks, and verification outputs using Redis to improve response speed during exam-time usage.

Confidence Calibration

Confidence derived from retrieval similarity, evidence agreement, claim coverage, and source reliability to ensure trustworthy verification.

Knowledge Update Strategy

New documents are stored in Layer-2, metadata generated, compression applied, and only important chunks embedded. Supports incremental indexing without full reprocessing.

Final Data Flow

```
User Query
   ↓
Vector Search (Layer-1)
   ↓
Metadata Lookup (Layer-3)
   ↓
Fetch Document Section (Layer-2)
   ↓
Re-rank Evidence
   ↓
Verification Agent
   ↓
Explanation Agent
   ↓
User (with traceable evidence)
```