Signal Memory Engine — Query Endpoint Flow

This document describes the flow of the application when a user sends a request to the `/query` endpoint. It highlights each stage of processing, from request intake to response generation and logging.

Stage	Description
1. User Request	The user (via Streamlit UI or HTTP client) sends a POST request to `/query`.
2. API Routing	FastAPI receives the request and routes it to `api/routes/query.py`.
3. Biometric Context	Synthetic biometric signals are sampled from `sensors/biometric.py`.
4. Retrieval & LLM	The query is passed to LangChain's RetrievalQA chain: • Query embedding generated via OpenAI/HF embeddings. • Pinecone vector store searched for top-k relevant chunks. • OpenAI Chat model generates an answer using retrieved context.
5. Coherence Mapping	Raw vector hits are normalized into structured memory events via `commons.py`.
6. Scoring & Flagging	Top similarity scores determine a stability flag: • stable (≤0.5) • drifting (>0.5) • concern (>0.8) Suggestions are mapped accordingly (e.g., escalate, check-in).
7. Logging & Storage	• Results logged in MLflow (query, embeddings, scores, latency). • Signal persisted in SQLite (`storage/sqlite_store.py`). • Trace log entry appended (`utils/tracing.py`). • Dashboard stub notified (`utils/dashboard.py`).
8. Response	The API responds with a JSON object containing: • Answer • Retrieved chunks • Stability flag • Suggestion • Trust score