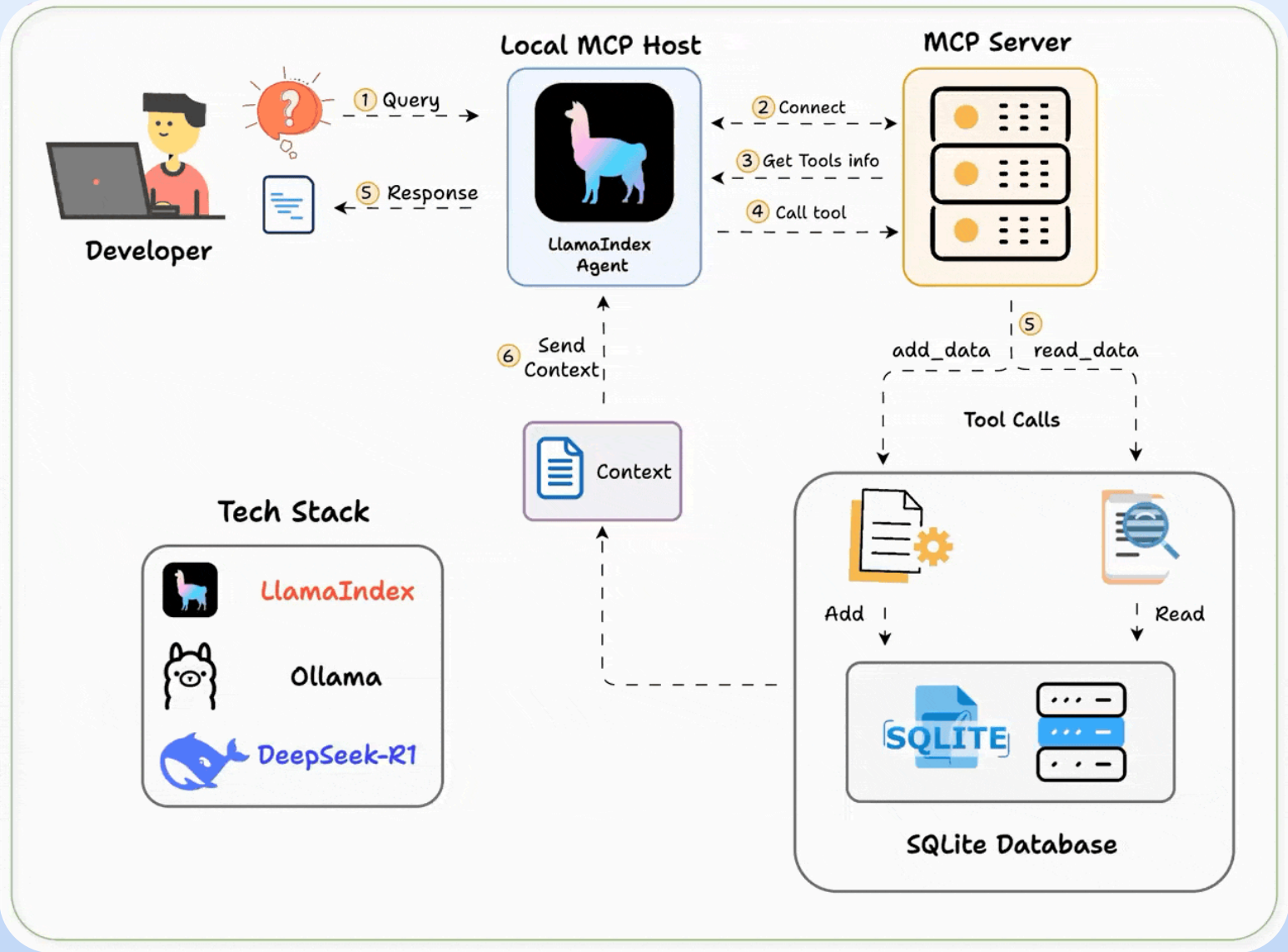
15 MCP PROJECTS for AI AGENT DEVELOPERS

SWIPE →

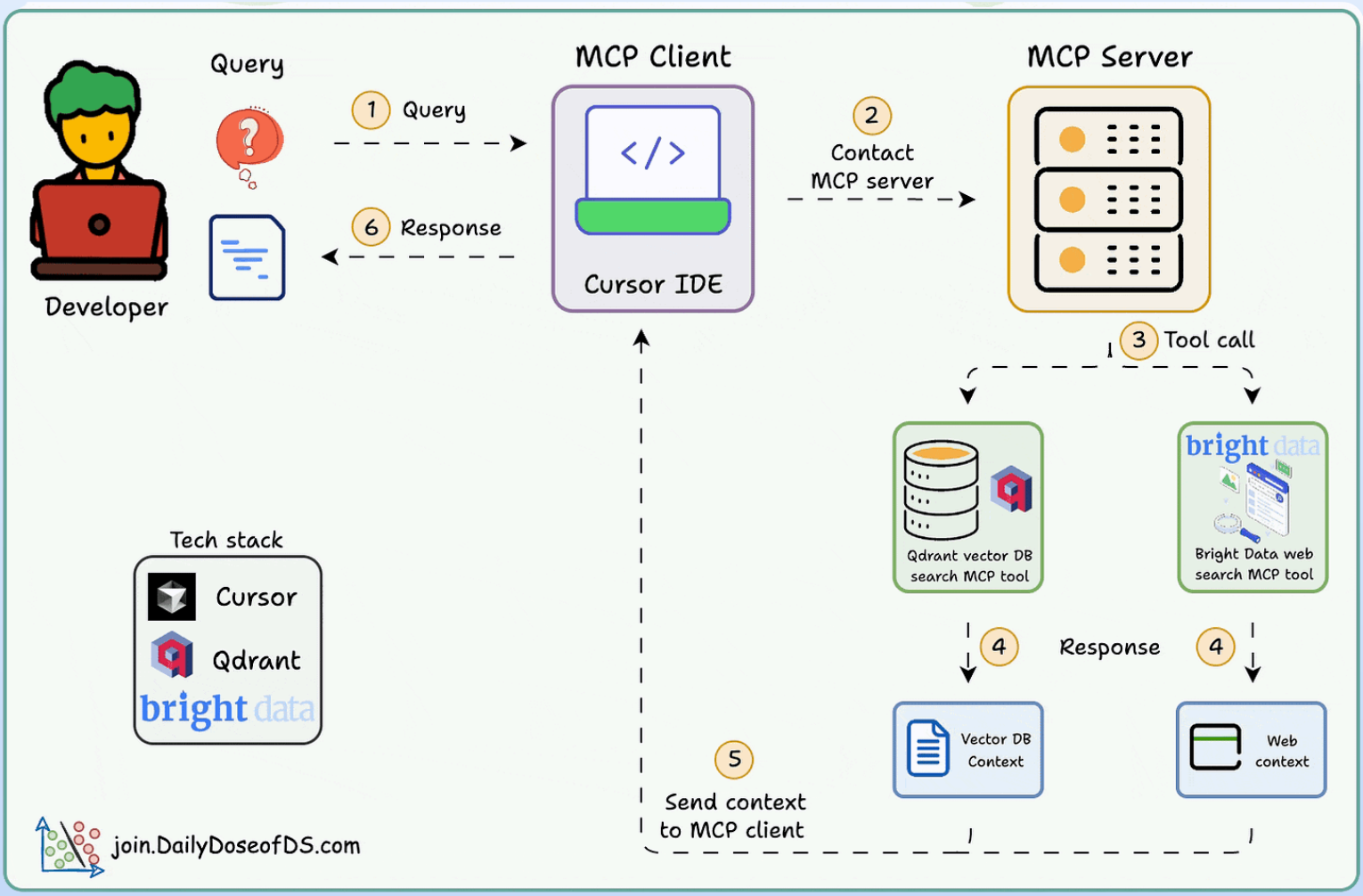
Build a fully local MCP client



Here's our workflow:

- **The user submits a query.**
- **Agent connects to the MCP server to discover tools.**
- **Based on the query, the agent invokes the right tool and gets context**
- **Agent returns a context-aware response.**

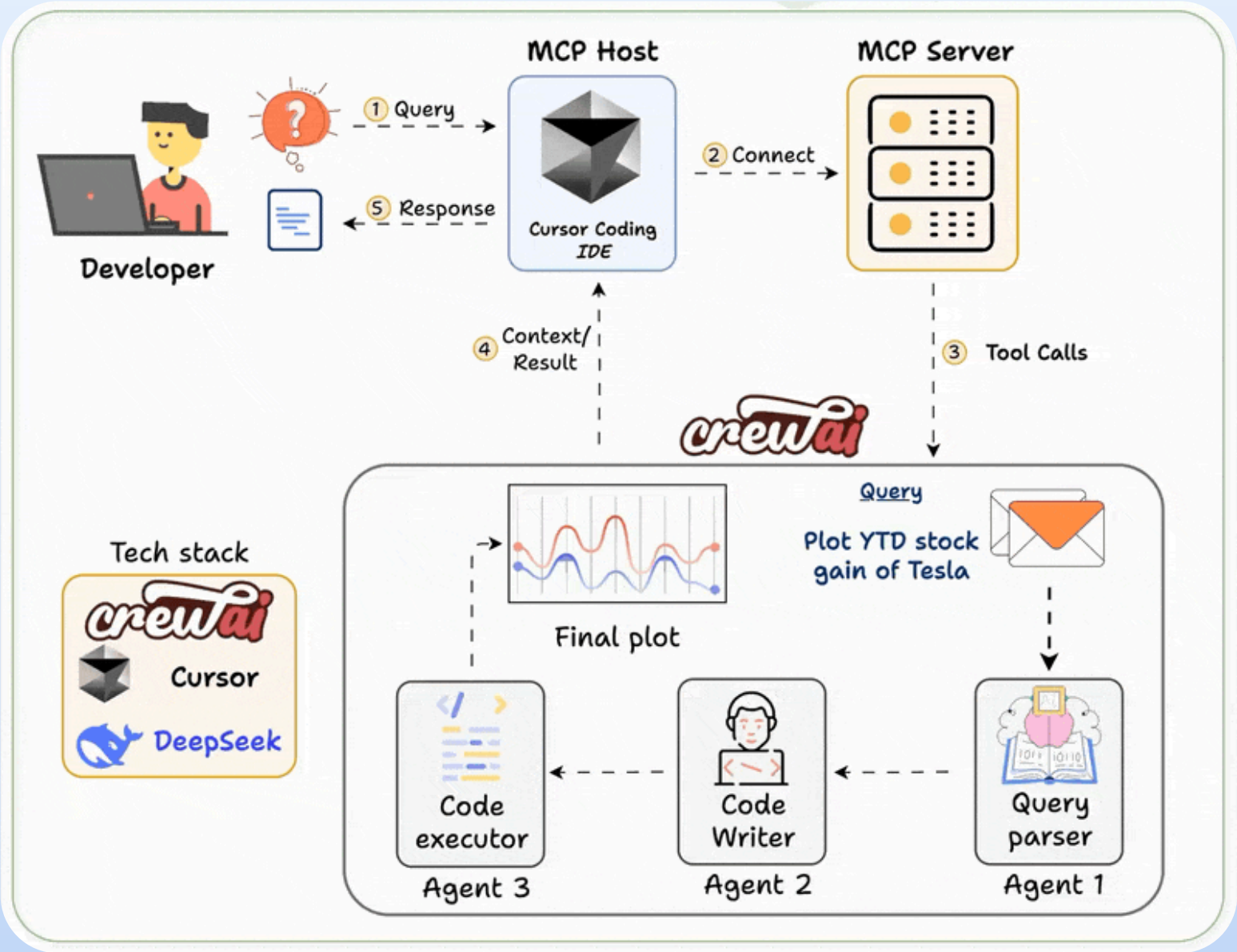
RAG with intelligent fallback



Here's our workflow:

- **(1) The user inputs a query through the MCP client (Cursor).**
- **(2-3) The client contacts the MCP server to select a relevant tool.**
- **(4-6) The tool output is returned to the client to generate a response.**

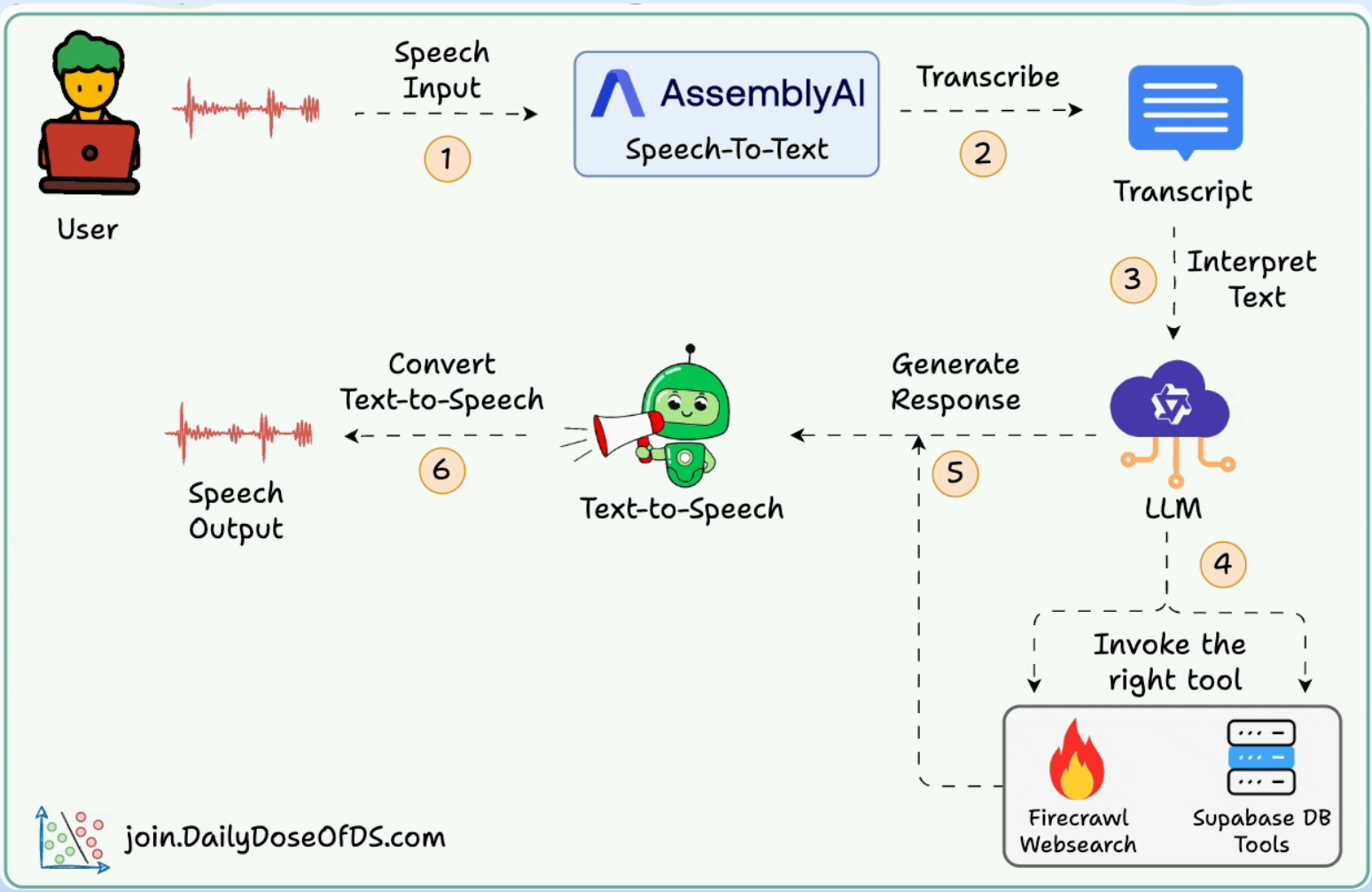
MCP-powered financial analyst



Here's our workflow:

- **The user submits a query.**
- **The MCP agent kicks off the financial analyst crew.**
- **The Crew conducts research and creates an executable script.**
- **The agent runs the script to generate an analysis plot.**

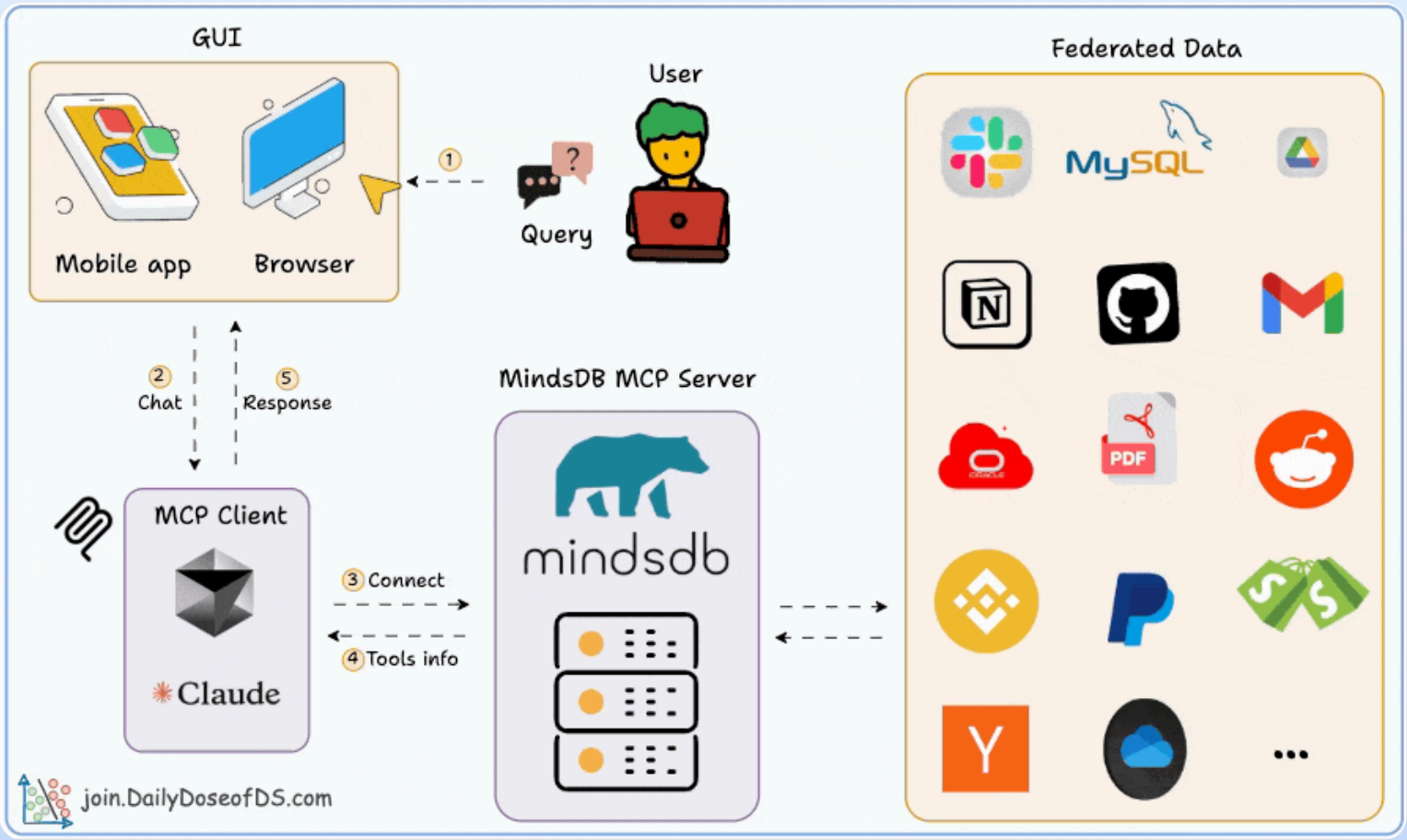
Voice-based assistance with tool use



Here's our workflow:

- **The user's speech query is transcribed to text with AssemblyAI.**
- **Agent discovers DB & web tools.**
- **LLM invokes the right tool, fetches data & generates a response.**
- **The app delivers the response via text-to-speech.**

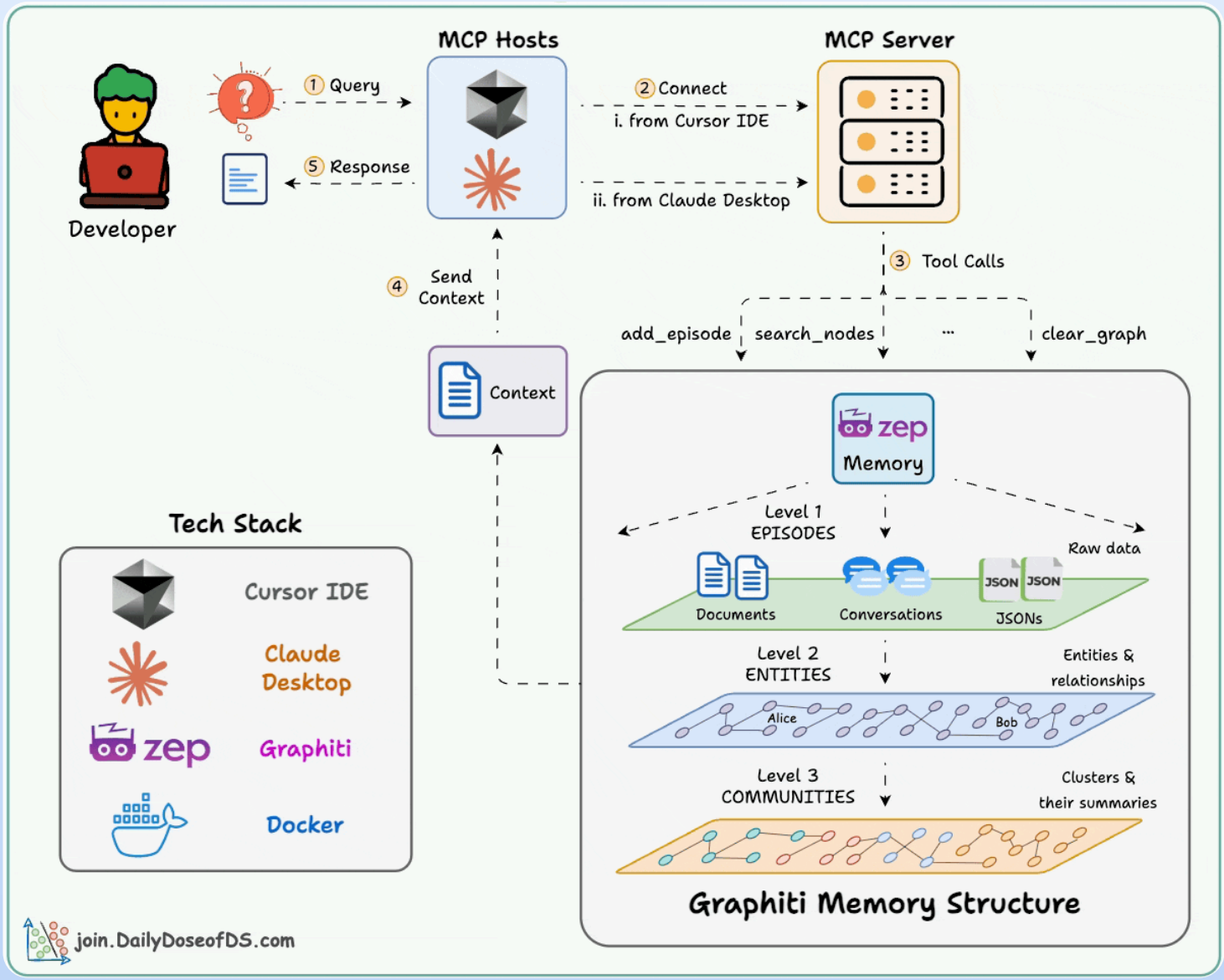
Unified MCP server for 200+ data sources



Here's our workflow:

- **User submits a query**
- **Agent connects to the MindsDB MCP server to find tools**
- **Agent selects an appropriate tool based on the user query and invokes it**
- **Finally, it returns a contextually relevant response**

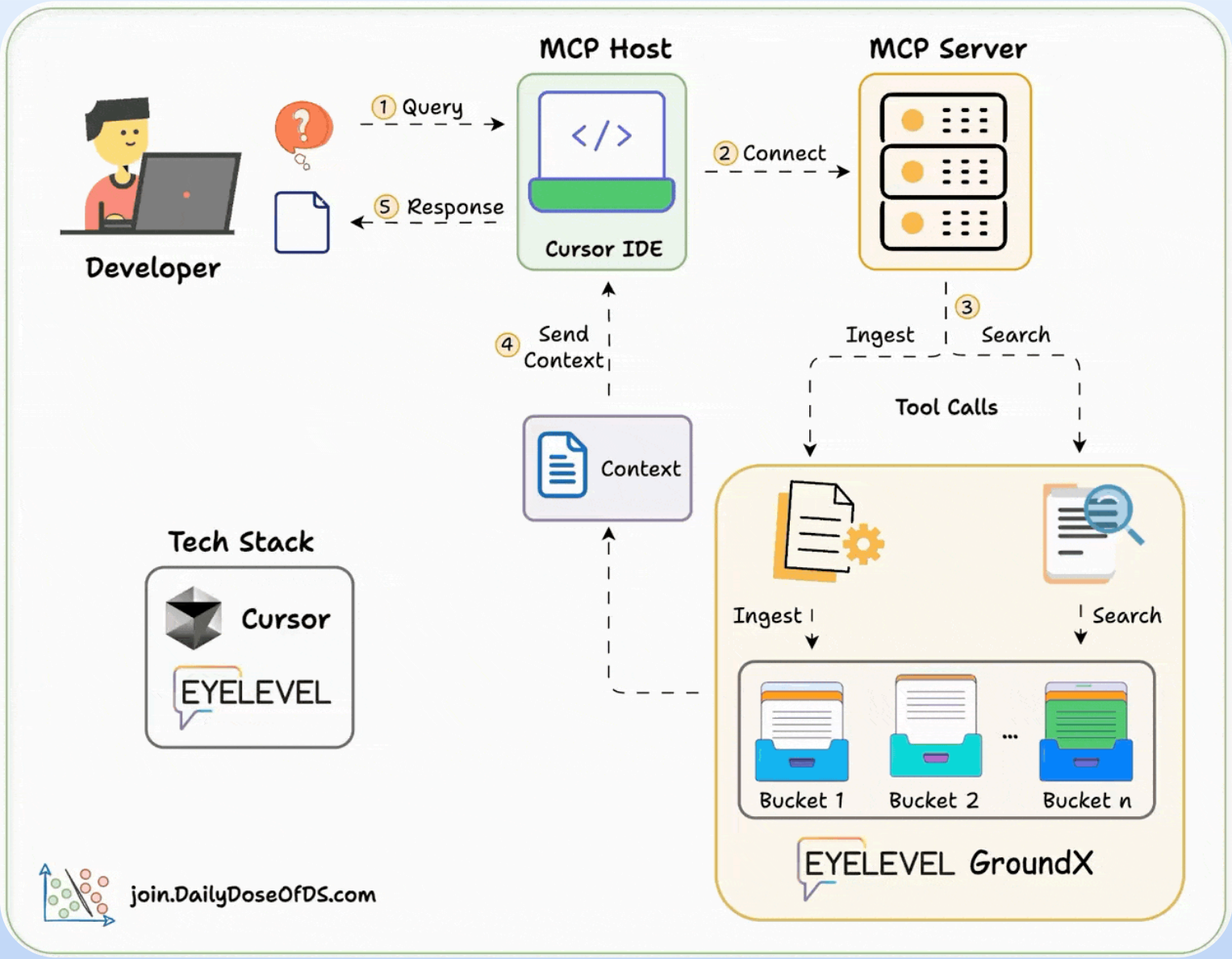
Shared memory between Cursor and Claude Desktop



Here's our workflow:

- **User submits query to Cursor & Claude.**
- **Facts/Info are stored in a common memory layer using Graphiti MCP.**
- **Memory is queried if context is required in any interaction.**
- **Graphiti shares memory across multiple hosts.**

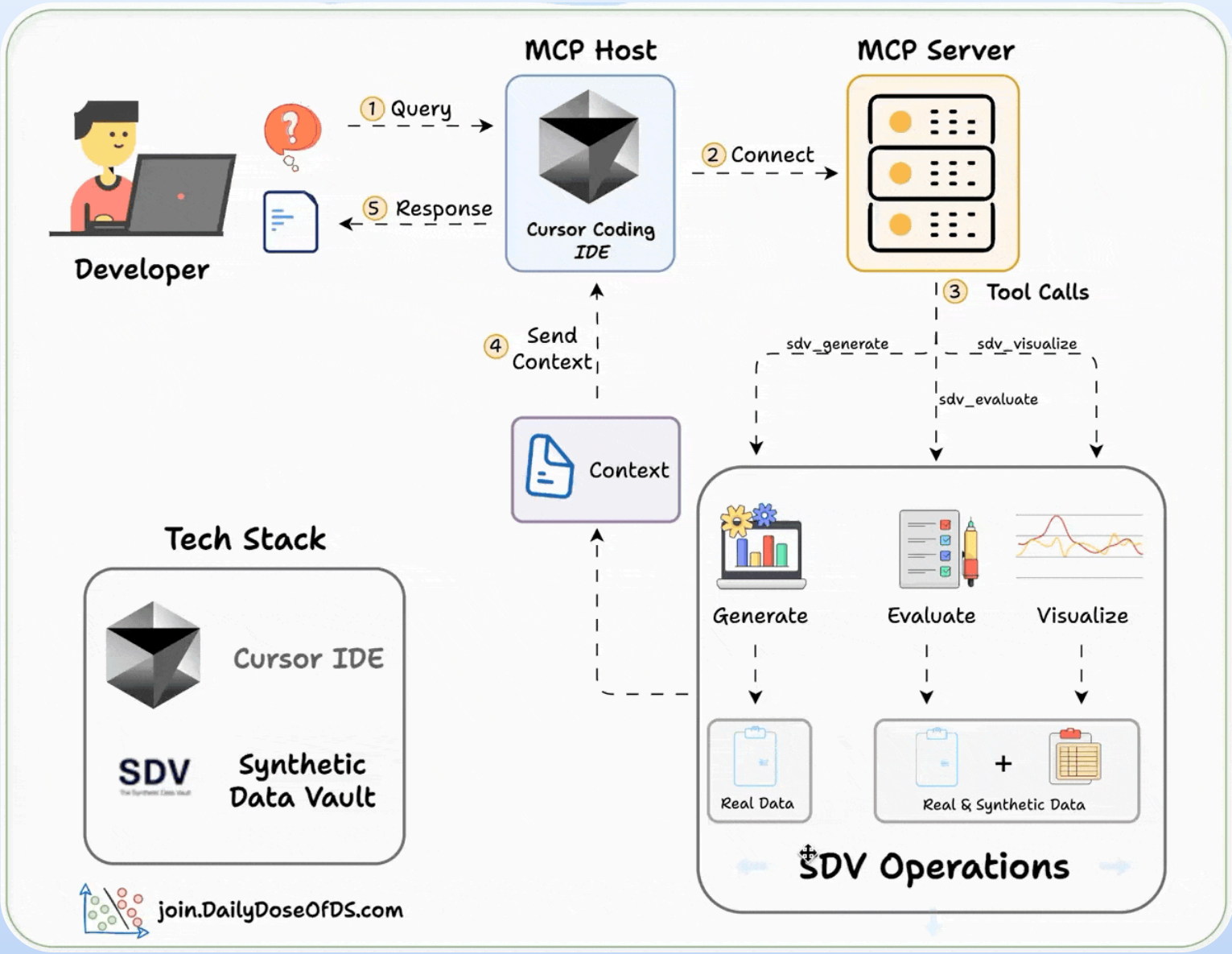
RAG over complex documents



Here's our workflow:

- **User interacts with the MCP client (Cursor IDE)**
- **Client connects to the MCP server and selects a tool.**
- **Tools leverage GroundX to do an advanced search over docs.**
- **Search results are used by the client to generate responses.**

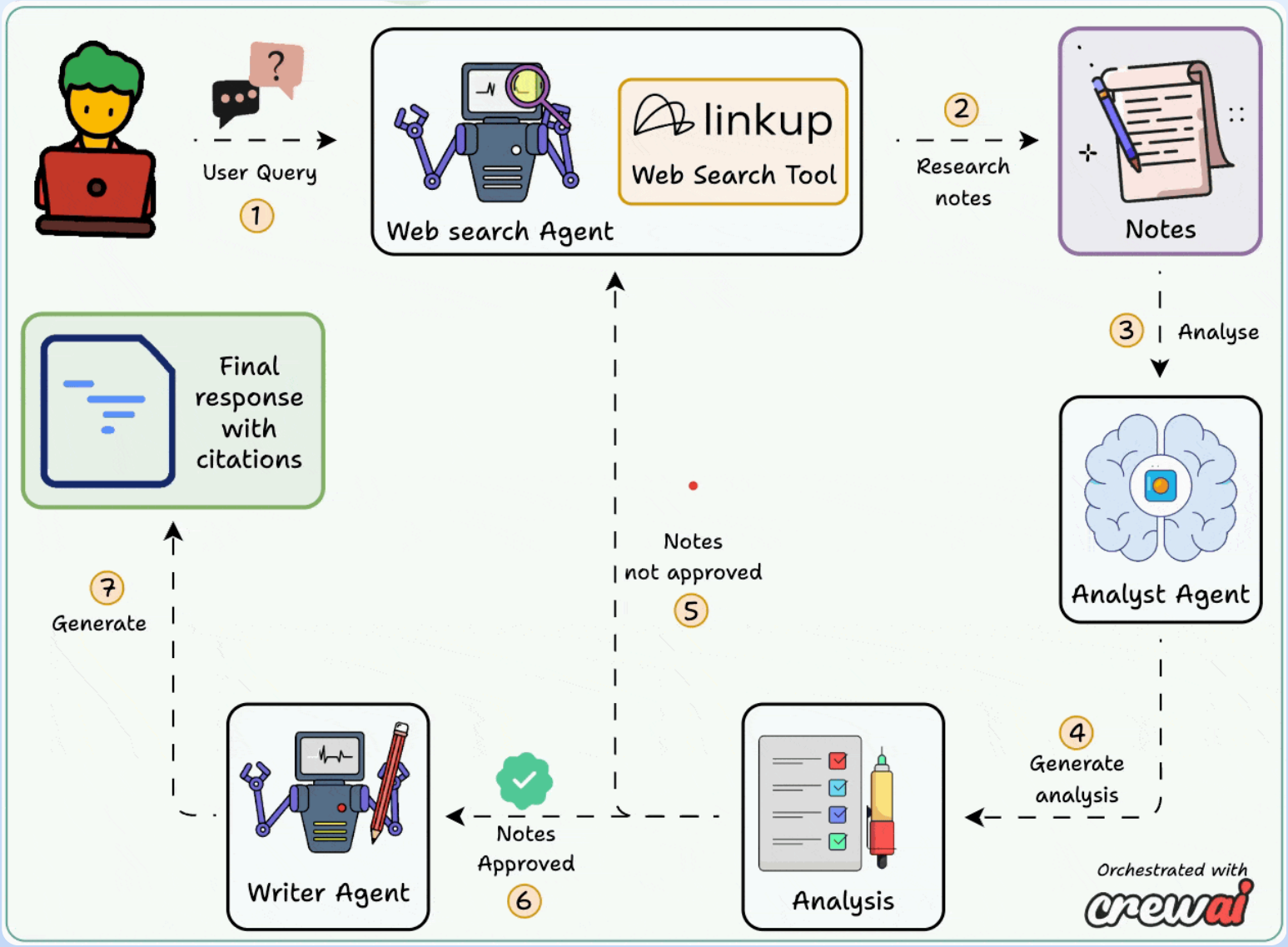
Synthetic data generation agent



Here's our workflow:

- **User submits a query**
- **Agent connects to the MCP server to find tools**
- **Agent uses the appropriate tool based on the query**
- **Returns response on synthetic data creation, eval, or visualization**

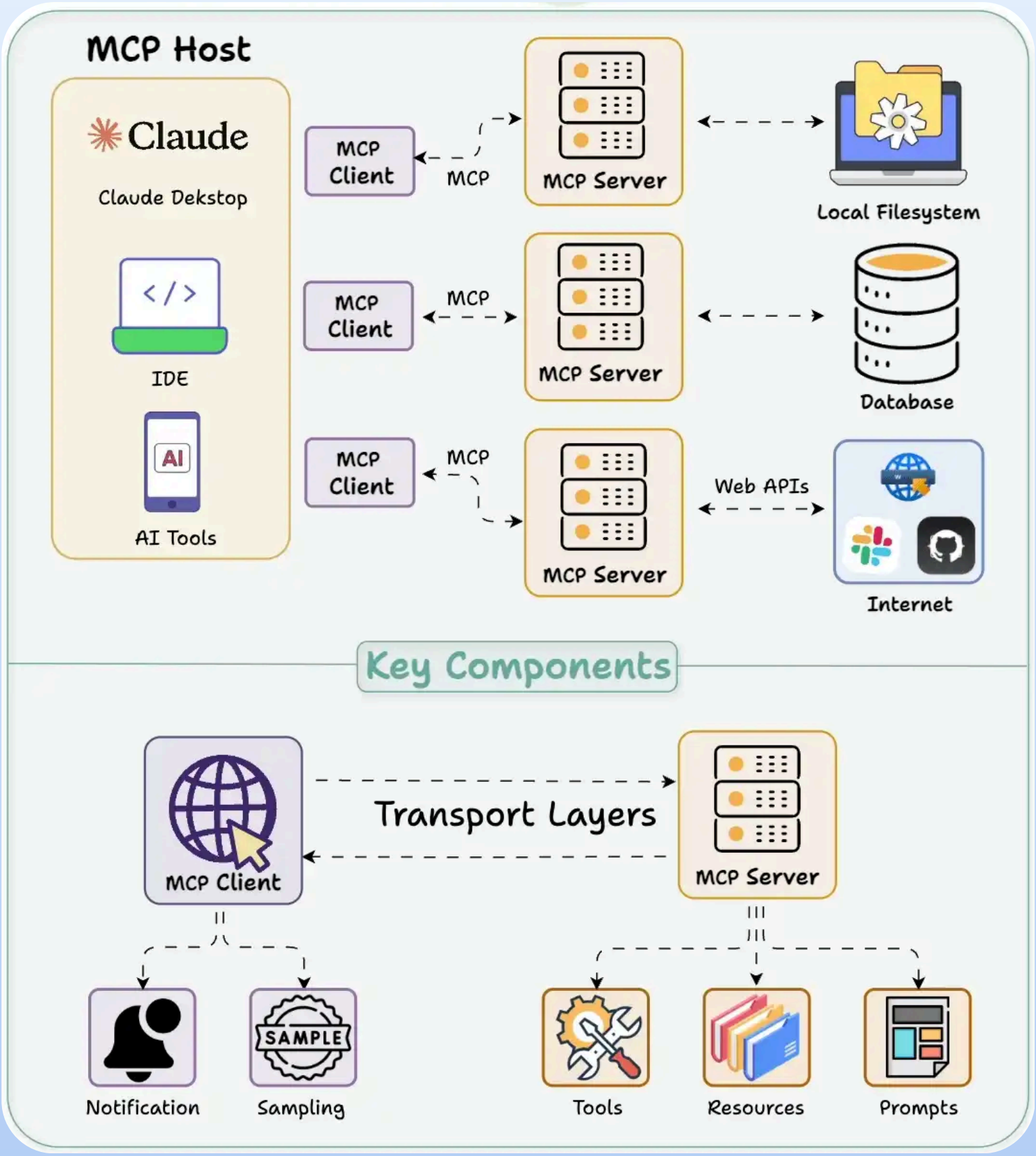
Local deep research assistant



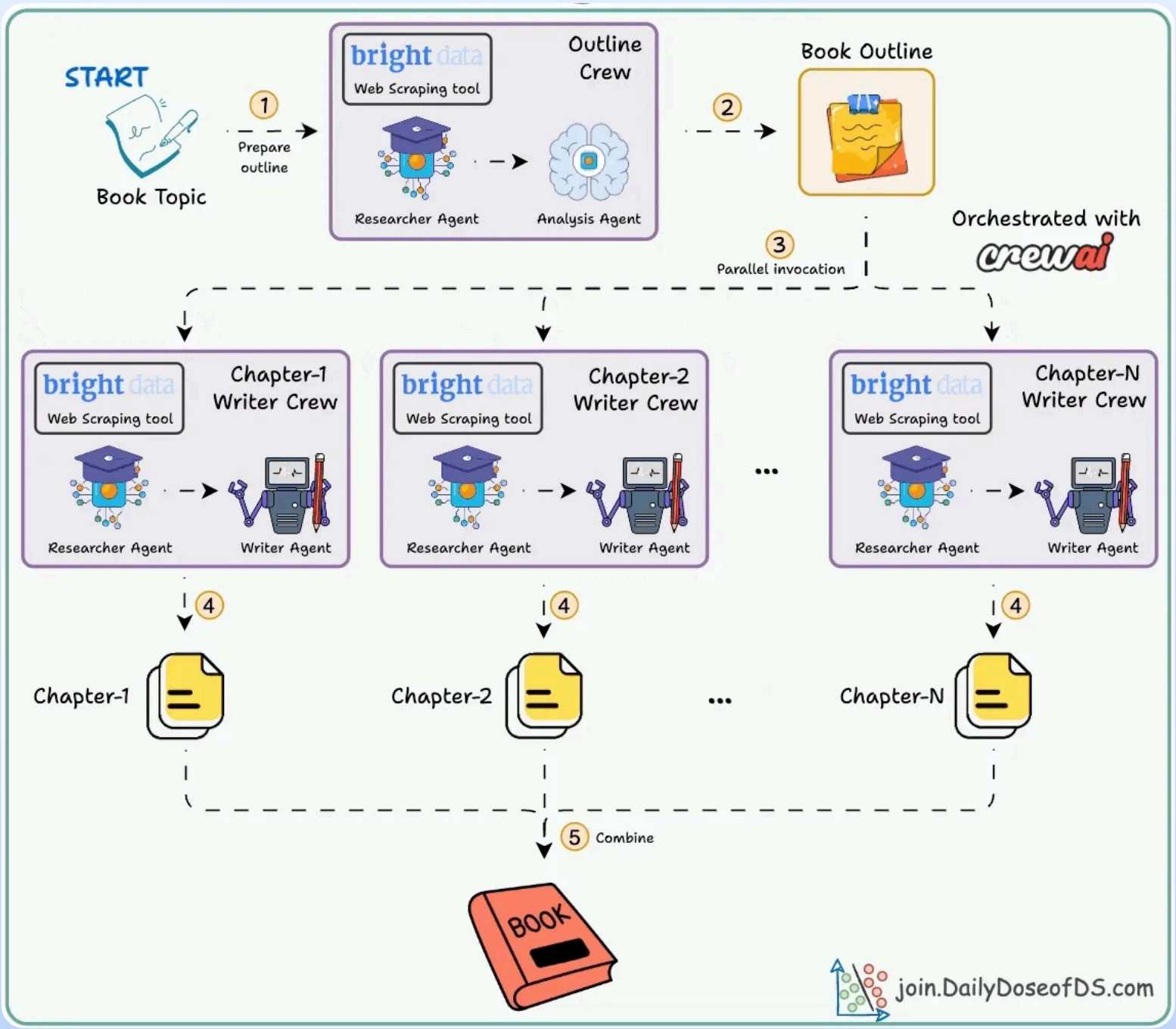
Here's our workflow:

- **User submits a query.**
- **Web search agent runs deep web search via Linkup.**
- **Research analyst verifies and deduplicates results.**
- **Technical writer crafts a coherent response with citations.**

MCP Explained Clearly



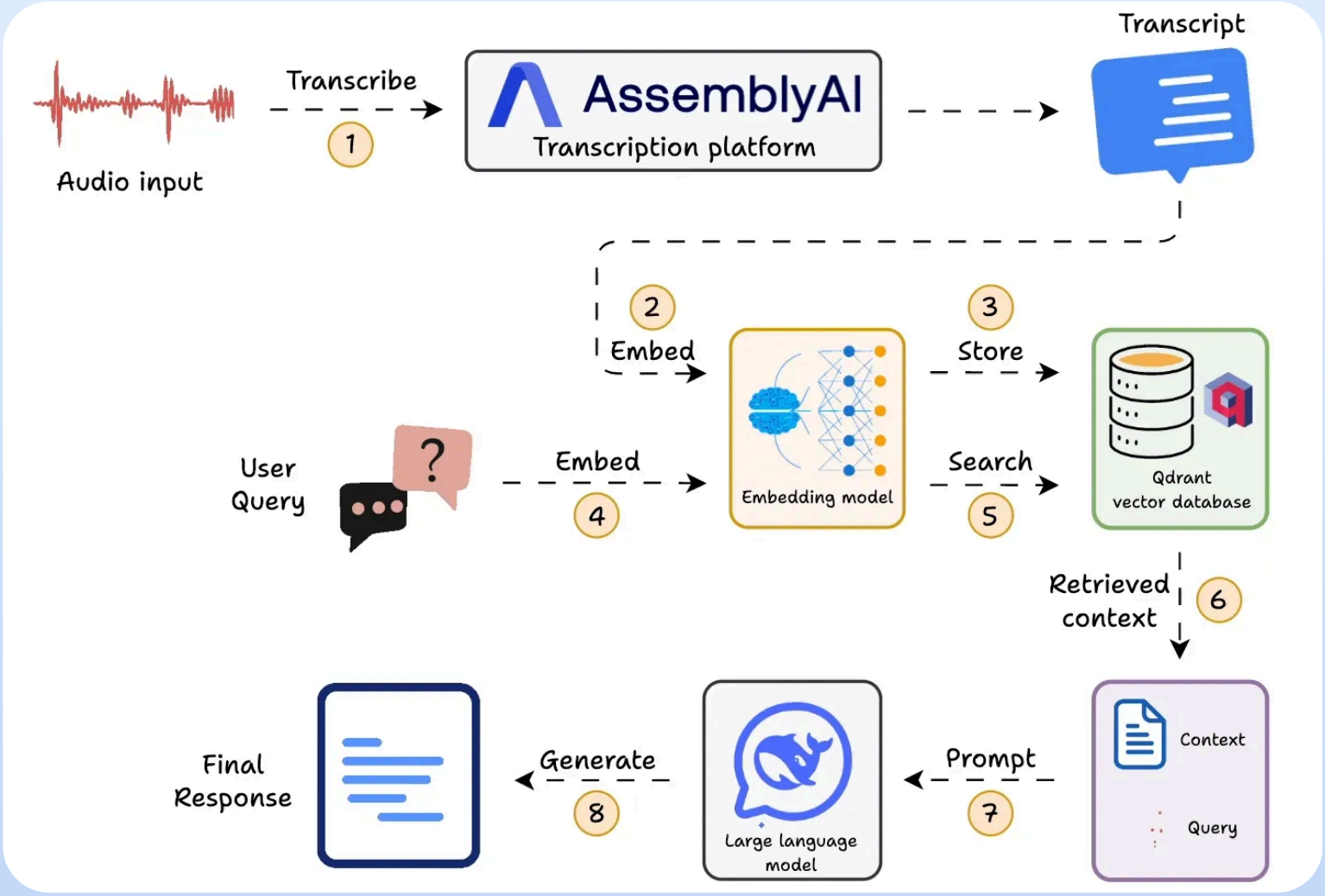
Book Writer Agentic Workflow



Here's our workflow:

- **Using Bright Data, Outline Crew scrapes data related to the book title and decides the chapter count and titles.**
- **Many Writer Crews are invoked in parallel to write one chapter each.**
- **Combine all chapters to get the book.**

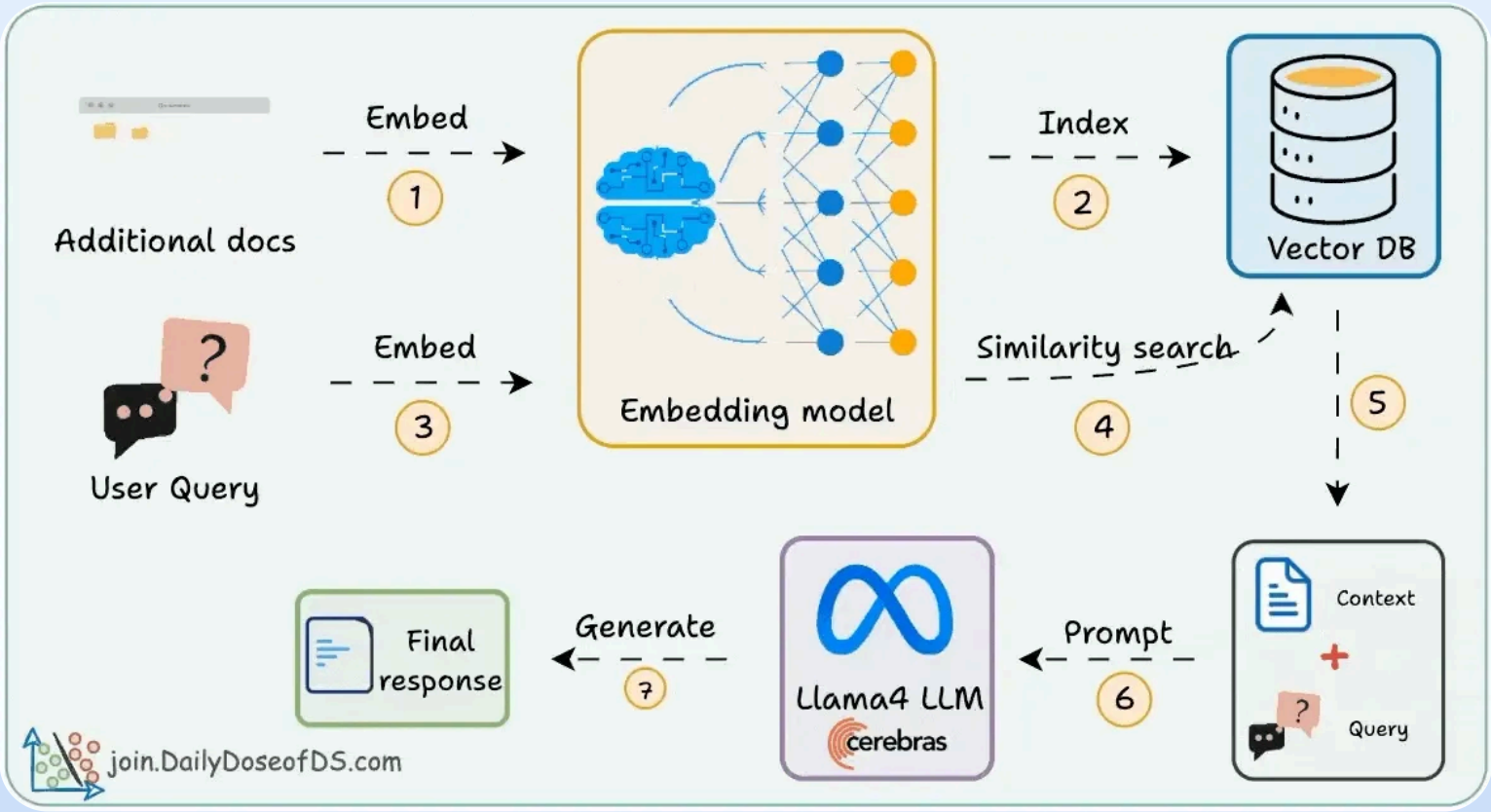
RAG over Audio using AssemblyAI and DeepSeek



Here's our workflow:

- **Step 1) Takes an audio file and transcribes it using AssemblyAI.**
- **Steps 2-3) Stores it in a Qdrant vector database.**
- **Steps 4-6) Queries the database to get context.**
- **Steps 7-8) Uses DeepSeek-R1 as the LLM to generate a response.**

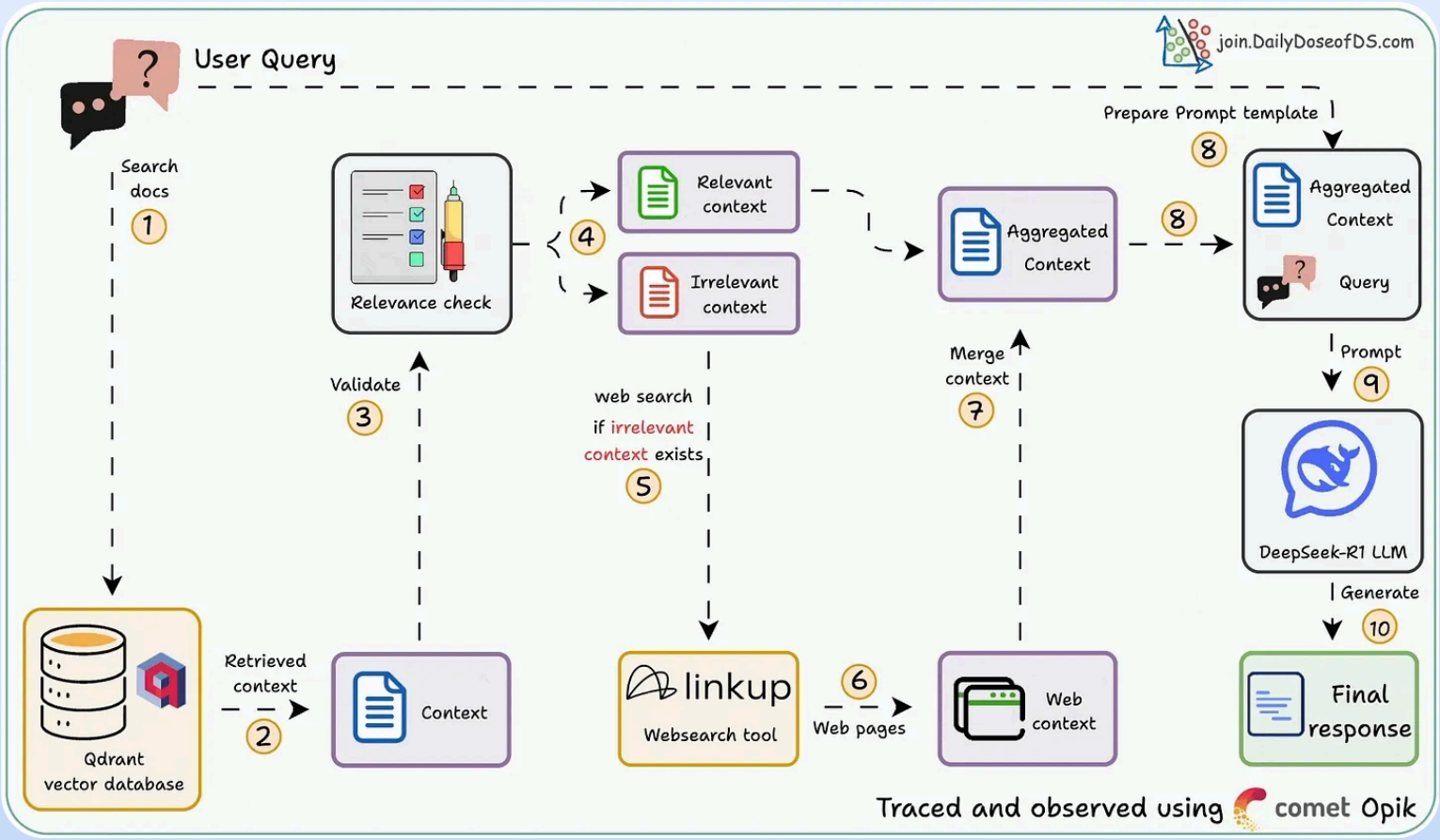
RAG powered by Llama 4



Here's our workflow:

- **Sets up a language model and embedding model.**
- **Ingests documents and builds an index.**
- **Retrieves the most relevant information for a query.**
- **Synthesizes a refined answer from the results.**

Corrective RAG Agentic Workflow



Here's our workflow:

- **First search the docs with user query.**
- **Evaluate if the retrieved context is relevant using LLM.**
- **Only keep the relevant context.**
- **Do web search if needed.**
- **Aggregate the context & generate a response.**

Interested in more content like this?

Follow me :

OM NALINDE