Building a complete AI assistant

In this final project, you will build a single AI assistant incorporating all the techniques from the course – and you will connect it to Claude Desktop via MCP so that other AI tools can utilize your n8n workflow.

Seven lessons on nodes, agents, memory, RAG, and production models. Now, it's all going to be combined. In this culminating project, you'll build a single AI assistant incorporating all the techniques from the course—and you'll connect it to Claude Desktop via MCP so other AI tools can utilize your n8n workflow.

Project summary

You will build **an AI Operations Assistant** - a chatbot that can:

1. Answer the questions based on your company's knowledge base (RAG from lesson 6).
2. Searching for current information on the web (Agent tool from lesson 4)
3. Remember the context of the conversation between sessions (Memory from Lesson 5)
4. Run calculations and format data (Code tools from Lesson 4)
5. Handle errors skillfully without crashing (Production model from lesson 7)
6. Called from Claude Desktop or other AI tools via MCP

? **Quick Check** : Your assistant needs to answer both questions about internal policy and questions about real-time market data. What two tools make this possible?

Answer: The Vector Store tool is for internal policies – it searches your company's internal documents. And SerpAPI is for real-time market data – it searches the web. The system prompt will tell employees which tool to use for which type of query.

This type of workflow replaces the role of a lower-level operations employee—not because it replaces that person, but because it handles the repetitive lookup/response/forwarding tasks that would otherwise take up their working time.

Overview of architecture

This is the complete workflow structure:

```
Chat Trigger ? AI Agent (GPT-4o) ??? Công c?: Supabase Vector Store (tài li
?u công ty) ??? Công c?: SerpAPI (tìm ki?m web) ??? Công c?
: Code Tool (tính toán) ??? Công c?: HTTP Request (API n?i b?) ??? B? nh?
```

```
: PostgreSQL Chat Memory ??? Thông báo l?i ? C?nh báo Slack + Ph?n h?i d?
 phòng
```

A Chat Trigger. An AI agent at the center. 4 tools for different capabilities. Persistent memory. Error handling on any external node.

Develop a summary project.

Step 1: Set up the platform

1. Create a new workflow called "AI Operations Assistant"
2. Add a **Chat Trigger** node - this creates the chat interface.
3. **Add an AI Agent** node and connect it to **Chat Trigger.**
4. Attach the OpenAI conversational model ? gpt-4o (you need strong reasoning capabilities for multi-tool decisions)

Step 2: Connect your tools

Attach 4 tools to the AI ??agent:

Tool 1: Vector Archive (RAG)

Add a **Vector Store Tool** child node pointing to your Supabase vector store from lesson 6. This allows the agent to access your company's knowledge base.

Tool 2: Web search

Add a SerpAPI tool to obtain real-time web information. Agents use this tool when their knowledge base lacks an answer.

Tool 3: Code Execution

Add a **Code Tool** that supports JavaScript. The agent can write and run code to perform calculations, format data, or process it.

Tool 4: HTTP Request

Add an **HTTP Request Tool** . Configure it with your internal API endpoints (or public APIs for practice). The agent can call the API to check state, retrieve data directly, or trigger actions.

Step 3: Add storage

Connecting PostgreSQL chat memory to an AI agent:

1. Connection: your PostgreSQL instance
2. Session ID: `{{ $json.sessionId }}`from Chat Trigger

3. This loads the chat history so the agent remembers previous exchanges.

Step 4: Write the system prompt.

This is the most important part - it coordinates everything:

```
B?n là AI Operations Assistant cho nhóm. B?n có quy?n truy c?
p vào: 1. **Company Knowledge Base** (Vector Store): Tài li?u n?i b?
, chính sách, quy trình ? S? d?ng cho: câu h?i v? s?n ph?m, tra c?
u chính sách, quy trình n?i b? ? Luôn ki?m tra ? ?ây TR??C tiên ??i v?
i các câu h?i n?i b?
 2. **Web Search** (SerpAPI): Thông tin Internet theo th?i gian th?c ? S? d
?ng cho: tin t?c ngành, d? li?u ??i th? c?nh tranh, thông tin bên ngoài ?
 Ch? s? d?ng khi c? s? ki?n ??th?c không có câu tr? l?
i 3. **Code Tool**: Môi tr??ng th?c thi JavaScript ? S? d?
ng cho: tính toán, ??nh d?ng d? li?u, chuy?n ??i ? Trình bày các b??
c tính toán 4. **HTTP Request**: Truy c?p API n?i b? ? S? d?ng cho: ki?
m tra tr?ng thái h? th?ng tr?c ti?p, truy xu?t d? li?u theo th?i gian th?
c Quy t?c: - ??i v?i l?i chào h?i ??n gi?n và cu?c trò chuy?
n xã giao, hãy tr? l?i tr?c ti?p mà không s? d?ng công c? - Luôn trích d?
n ngu?n: "[Knowledge Base]" ho?c "[Web: URL]" - N?u b?n không tìm th?
y thông tin ?áng tin c?y, hãy nói rõ ?i?u ?ó - ??ng bao gi? ?oán mò - Gi?
 câu tr? l?i ng?n g?n (d??i 200 t?) tr? khi ???c yêu c?u chi ti?t h?
n - Hãy nh? b?i c?nh t? ph?n tr??c c?a cu?c trò chuy?n
```

Step 5: Add error handling functionality

For each external tool node:

1. Enable **Retry on Fail** (3 retries, rescheduling exponentially).
2. Configure **Error Output** to redirect to the Set node with the fallback message: "I'm having trouble accessing that resource right now. Let me try to help you with what I have, or please try again in a moment."

Set up a global Error Workflow (from lesson 7) to send Slack alerts when any execution fails.

Step 6: Check the entire system.

Run these test scripts through the chat:

Test	What are you verifying?
"What is our refund policy?"	Retrieving RAG data from a vector repository.
"What is the current price of Bitcoin?"	Web search engine
"Calculate compound interest on a $10,000 loan at a 5% interest rate over 3 years."	Code Tool
"What were we talking about just now?"	Memory continuity

Test	What are you verifying?
Ask a question, close the conversation, reopen it, and ask another question.	Memory between sessions
Please disconnect from the internet and ask your question on the web.	Error handling + backup plan

Additional tip: Connect via MCP

Model Context Protocol (MCP) allows you to expose your n8n workflows as tools that other AI assistants can call. This means, for example, Claude Desktop can use the Operations Assistant's knowledge base without you having to copy and paste anything.

n8n as an **MCP Server** (showing your workflows):

1. Add **MCP Server Trigger** to a new workflow.
2. Identify the name and description of the tool (e.g., "company_knowledge_base")
3. Connect it to the Q&A Chain with your vector repository.
4. Configure Claude Desktop's MCP settings to point to your n8n version.

MCP is evolving rapidly. As of March 2026, n8n supports the HTTP Streamable transport protocol. Please check the n8n documentation for the latest MCP setup instructions, as details change frequently.

Next step

You have completed the lessons. Here's how to continue your progress:

1. **Automate your actual workflows** – Start with the task you perform most frequently. Email categorization, report generation, data lookup – choose one and build upon it.
2. **Explore the template library** - n8n has over 8,000 community templates, including more than 5,800 AI-specific workflows. Filter by "AI" at n8n.io/workflows.
3. **Join the community** - Over 200,000 members on the n8n community forum. Post your workflows, get feedback, and learn from others.
4. **Scalability** - When a single instance isn't enough, explore multi-worker queueing, external secret management, and CI/CD for workflow deployment.

The gap between "I use AI chatbots" and "I build AI automation" is smaller than you think. You've just crossed it!

Key points to remember

1. Node AI Agents are central to complex workflows – they coordinate tools, memory, and reasoning capabilities within a single node.
2. A good system prompt will tell the agent when to use each tool, how to handle conflicts, and which output format to use.
3. Error handling is essential – retry logic, error output, and warnings are the difference between a prototype and a product.

4. MCP allows you to expose n8n workflows as tools for other AI assistants – and use external MCP tools within your agents.
5. Start simple, test thoroughly, and only add complexity when the simple method is insufficient.

1. Question 1:

   Your final workflow is complete. Before enabling it for real users, what is the most important thing to verify?

   1. A. The workflow diagram looks neat and organized.
   2. B. Error handling has been established - retry logic on API nodes, error output with fallback response, and global error workflow have been configured.
   3. C. You are using the most expensive LLM model available.

   EXPLAIN:

   A workflow without error handling will silently fail the first time an API crashes. Before going live: Enable Retry on Fail for all external API calls, configure error output on AI nodes with user-friendly fallback messages, and establish a global error handling process for alerts. A production workflow is incomplete until it handles errors skillfully.

2. Question 2:

   Your AI assistant has 5 connected tools. The user sends a simple greeting ('Hi, how are you?'). What should the agent do?

   1. A. Call all five tools to gather comprehensive context before responding.
   2. B. Direct response without calling any tools - the system prompt should instruct the agent to process simple messages without calling any tools.
   3. C. Search the vector repository for the presence of a greeting protocol in the document.

   EXPLAIN:

   An agent searching the web to respond with 'Hi' is a waste of time and tokens. A good system prompt includes instructions such as: 'For simple greetings and small talk, respond directly without using any tools.' This makes the response quick, efficient, and natural. Tool calls should only occur when the user asks something that requires external data.

3. Question 3:

   You are designing an AI assistant. Which n8n node should be at the center of the workflow?

   1. A. Basic LLM Chain - simpler and faster
   2. B. AI Agent - capable of using tools, accessing memory, and making multi-step decisions.
   3. C. Q&A Chain - handles document retrieval in a natural way.

   EXPLAIN:

AI Agents are the right choice because AI assistants need to combine multiple capabilities: tool usage (web search, code execution), memory (PostgreSQL), document retrieval (storing vectors as a tool), and decision-making. Basic LLM Chains cannot utilize tools. Q&A Chains only handle retrieval. Only AI Agents can connect everything together using their ReAct inference loop.

Submit your work

Training results

You have completed **0** questions.

-- / --

Review the lesson