Why Our AI Agent Couldn't Scale Without MCP – and We Build It

Mohit Jichkar
Senior Data Scientist

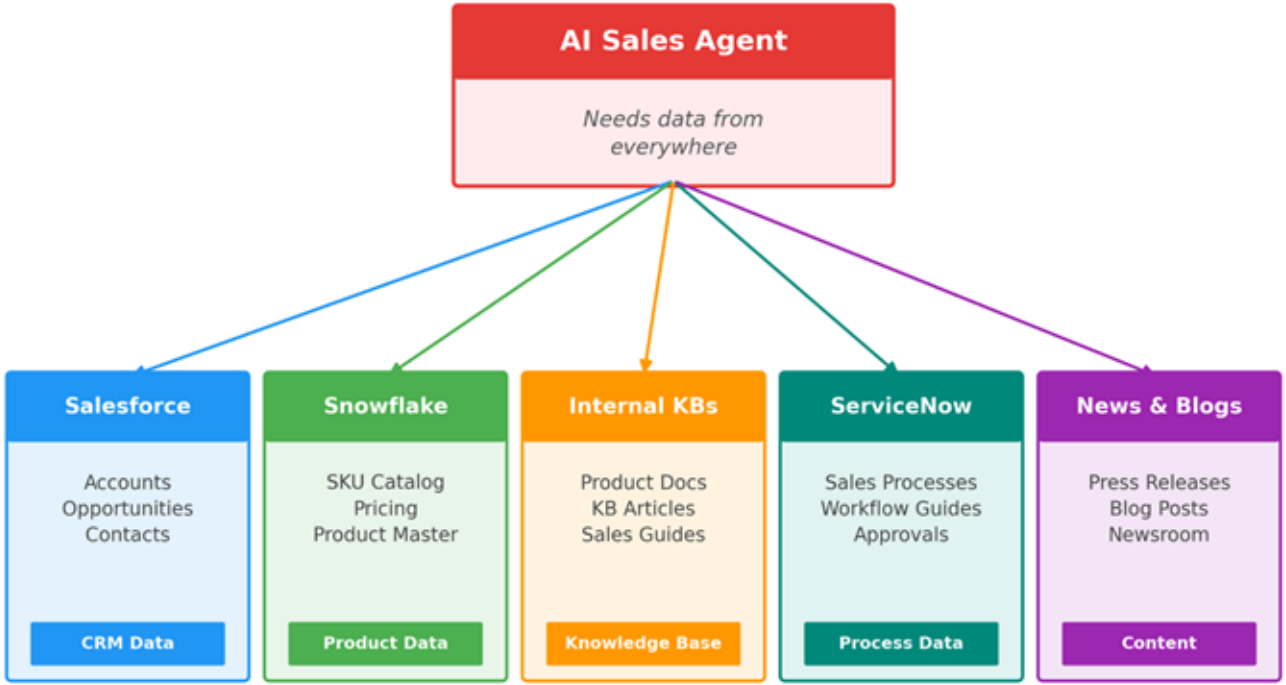
Para Hitesh
Senior Data Scientist

Red Hat

What We Build

- An **AI-powered sales assistant** for Red Hat's sales team
- Answers questions about products, accounts, pricing, SKUs, opportunities — all in one conversation
- Built with **LangGraph + LLMs + RAG across multiple knowledge collections**
- ReAct agent pattern — reasons step-by-step, decides which tools to call, and iterates until it finds the answer
- **12+ tools with intelligent routing** — middleware classifies user intent and dynamically injects context for accurate tool selection
- Deployed on **Red Hat OpenShift** (production)

The Data Is Never in One Place



"The agent needs to search, correlate, and reason across ALL of them — in a single conversation."

Red Hat

Why Connecting to External systems is Challenging

Code Complexity

- Each source has its own SDK, query language, and data format
- Every new source = weeks of integration code inside the agent
- Domain logic grows fast (SQL, pagination, transforms)

Authentication Chaos

- Each system has its own auth (OAuth, API keys, SSO)
- Token refresh logic duplicated per data source
- Agent becomes a credential store — security risk

Tight Coupling

- Can't update data layer without redeploying entire agent
- Domain teams blocked by agent team's release cycle
- One bug in data integration → redeploy the whole agent

Framework Lock-in

- Tools built for one framework only work in that framework
- Switch agent framework? Rebuild every integration
- No reusability across teams or projects

"We hit all four walls. We needed a different approach."

What Is MCP?

- **Model Context Protocol** — an open standard for connecting AI agents to tools and data
- Created by **Anthropic** (Nov 2024), wire format: **JSON-RPC 2.0**
- Now governed by the **Linux Foundation** under the **Agentic AI Foundation** (AAIF, Dec 2025)
- Co-founded by **Anthropic, Block, and OpenAI**
- Platinum members: **AWS, Google, Microsoft, Cloudflare, Bloomberg**
- **170+ member organizations** — more than 2× the pace of CNCF at the same stage
- Think of it as **"USB-C for AI"** — one protocol, any agent, any tool

- **97M+** monthly SDK downloads (4,750% growth in 18 months)
- **10,000+** public MCP servers (official registry: **9,652 latest records, ~16K GitHub repos**)
- Natively supported by: **Claude, ChatGPT, Gemini, Copilot, Cursor, VS Code, Windsurf, Zed**

The N×M → N+M idea

Without MCP: **5 agents × 5 data sources = 25 custom integrations** → With MCP: **5 agents + 5 servers = 10 implementations**

Every new server benefits ALL existing agents. Every new agent benefits from ALL existing servers.

Red Hat

What Is MCP? (continued...)

MCP vs Function Calling

	Function Calling	MCP
Scope	Single app, single model	**Cross-system, model-agnostic**
Discovery	Hardcoded in prompt	**Runtime discovery via tools/list**
Portability	Rebuild per framework	**One server, any client**
Credential Isolation	Limited	**Strong (transport-layer auth)**

They're complementary: function calling is the execution primitive; MCP is the infrastructure that makes tools portable and discoverable.

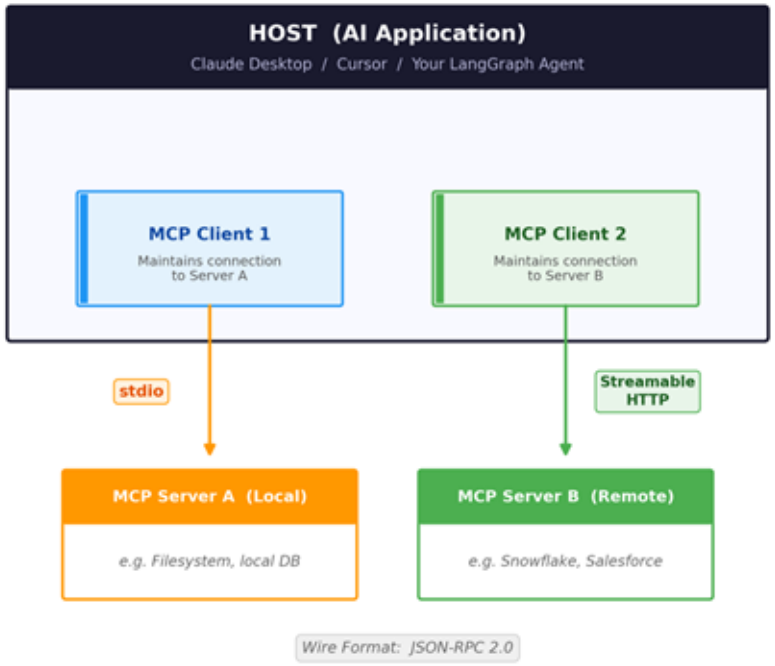
MCP Specification Evolution

Version	Date	Key Additions
2024-11-05	Nov 2024	Launch — Tools, Resources, Prompts, stdio transport
2025-03-26	Mar 2025	Streamable HTTP, OAuth 2.1 + PKCE, Tool Annotations
2025-06-18	Jun 2025	Elicitation, RFC 8707 Resource Indicators, security hardening
2025-11-25	Nov 2025	Tasks (long-running ops), server-side sampling
2026-07-28 RC	Jul 2026	Stateless core, MCP Apps, Tasks extension (in progress)

4 releases in 13 months · Next RC (Jul 2026) moves to fully stateless core

Red Hat

How MCP Works?



MCP Primitives & Transports

Three Primitives

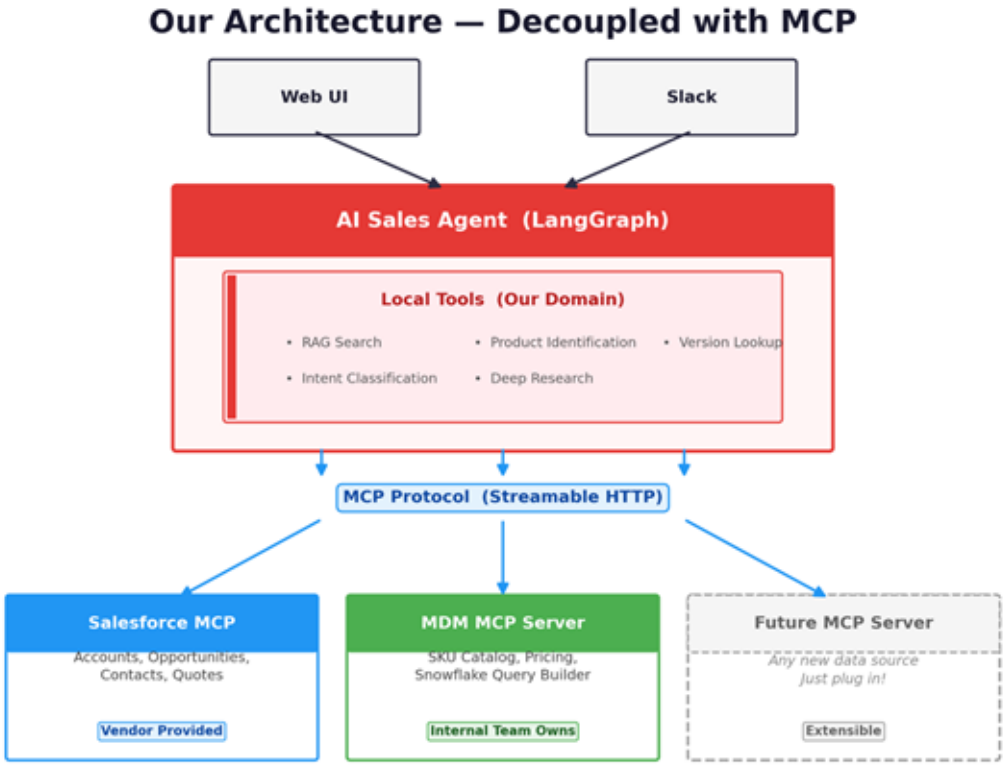
Primitive	What It Does	Controlled By
Tools	Functions the LLM can call (e.g. query database, search CRM)	The AI Model
Resources	Read-only data for context (e.g. file contents, DB schemas)	The Application
Prompts	Reusable prompt templates (e.g. workflow guides, instructions)	The User

Two Transports

Transport	Use Case	Example
stdio	Local — runs as subprocess on same machine	*Filesystem, local DB*
Streamable HTTP	Remote — deployed as service over network	*Snowflake, Salesforce, CRM*

Red Hat

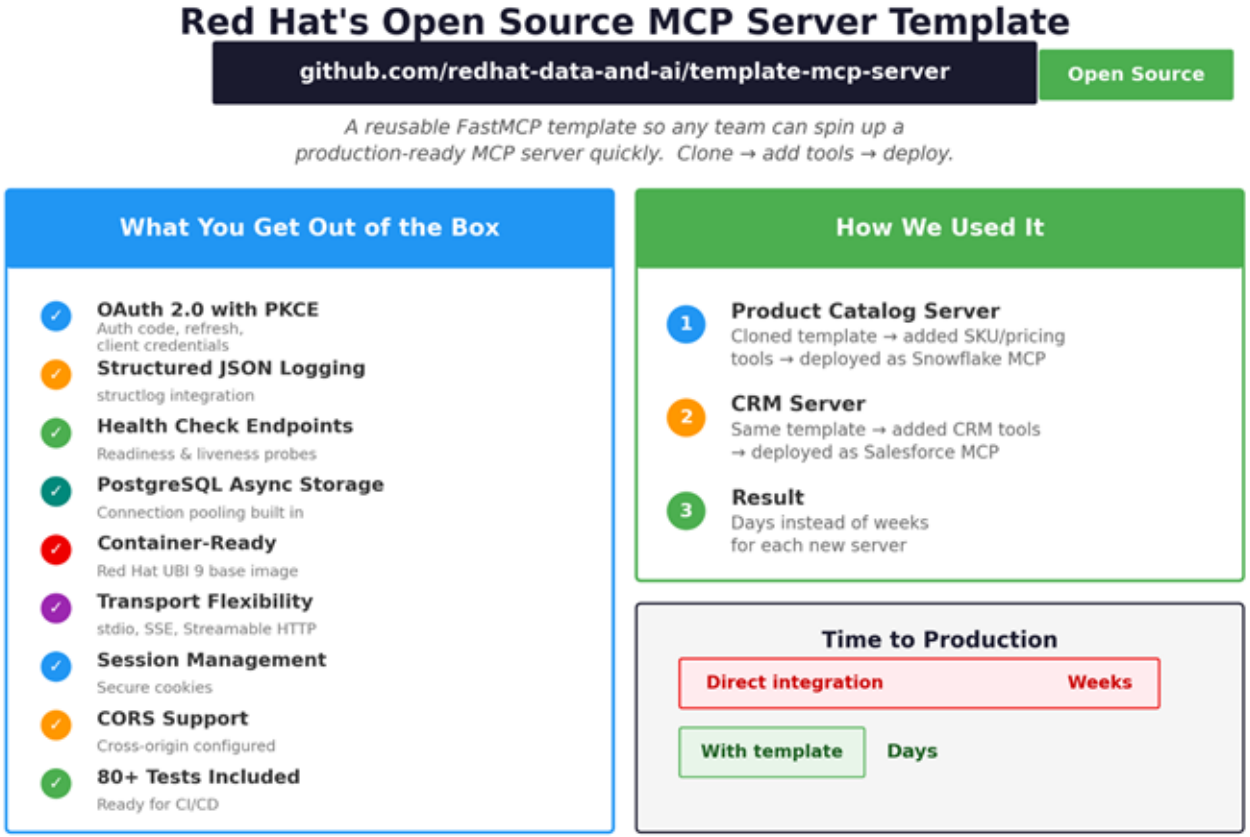
Our Architecture



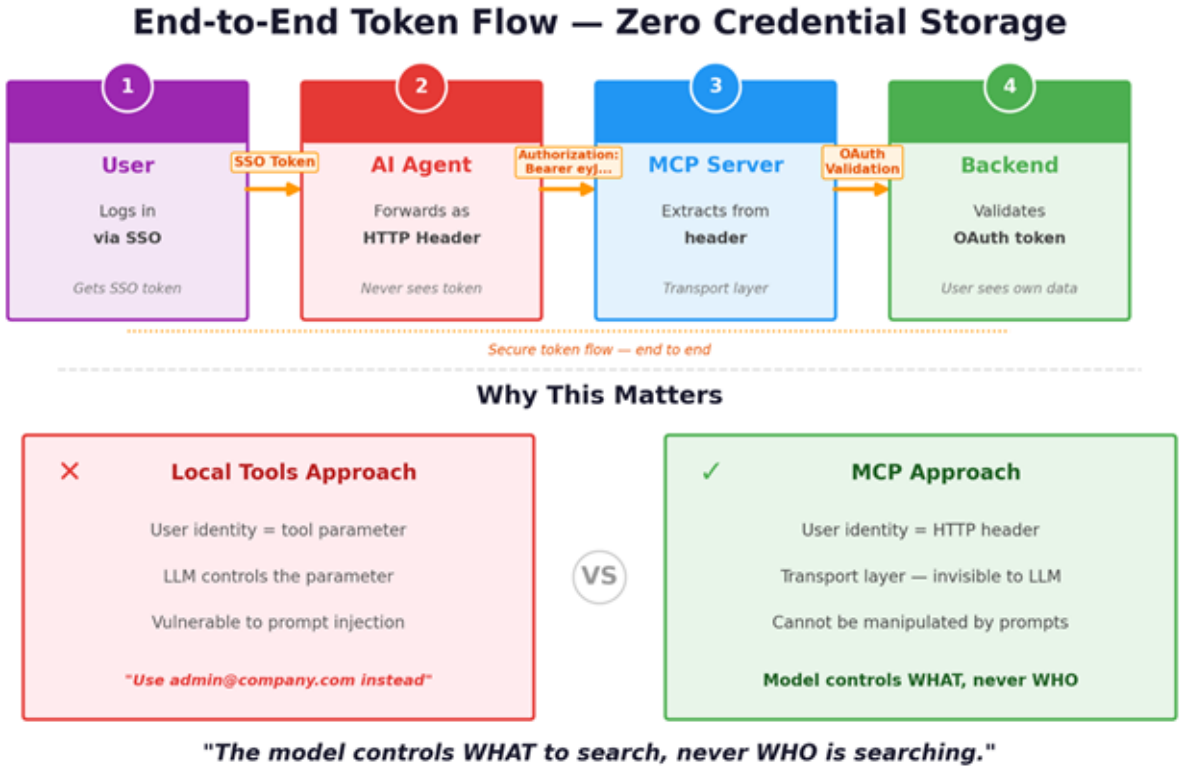
"Local tools for what WE own. MCP servers for what other teams or vendors own."

Red Hat

Template Pattern



Auth End-to-End



Red Hat

Tool Descriptions & Prompts

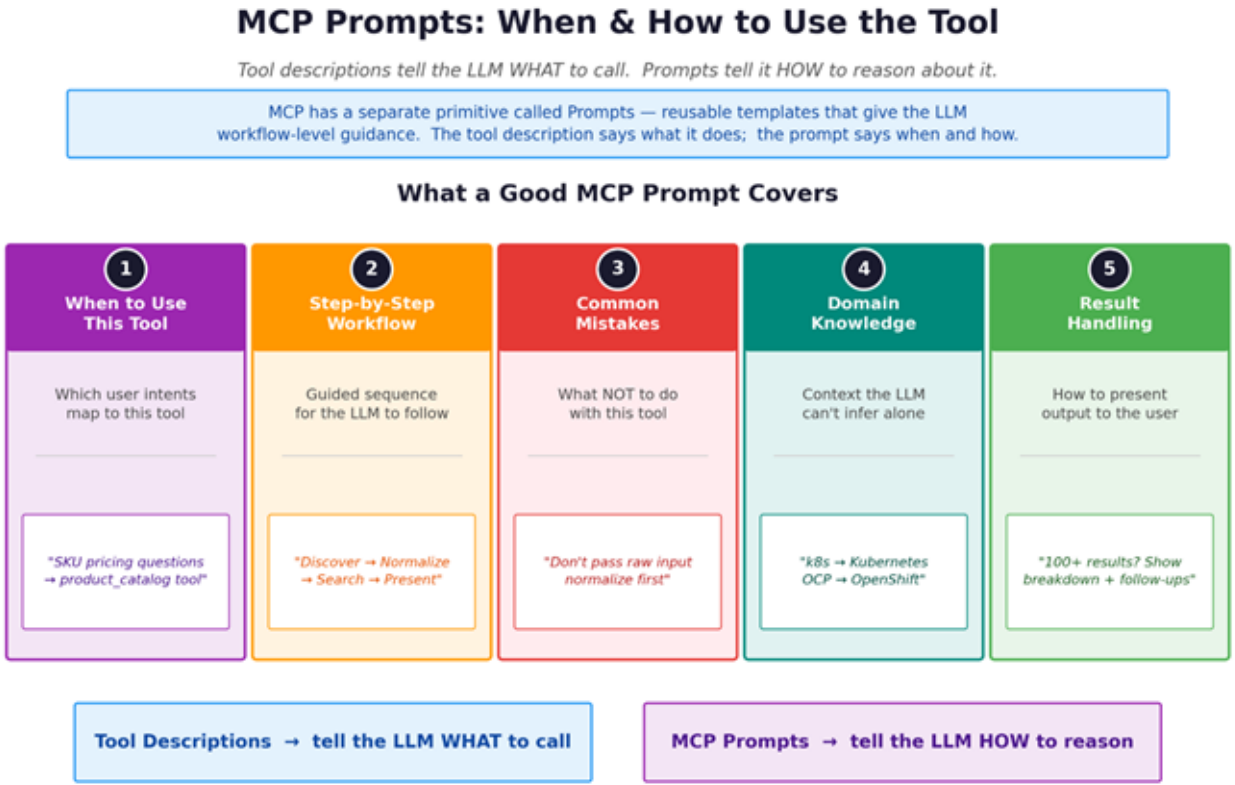
Tool Descriptions: What to Include

The tool description (docstring) is the ONLY thing the LLM reads to decide whether and how to call your tool. A vague description = wrong tool calls, missing parameters, failed queries.

Element	What It Tells the LLM	Example
What the tool does	Purpose and scope — when to use it vs. other tools	*"Search accounts by name. For opportunity lookups, use search_opportunity instead."*
Input args with types	Parameter names, types, constraints, valid values	*offering_group (str) Valid: "OpenShift", "RHEL", "Ansible" ...*
Output structure	What the response looks like, key fields returned	*Returns dict with: status, accounts[], total_count*
Examples	Concrete call patterns the LLM can mimic	*search_accounts(name="Acme", limit=10, offset=0)*

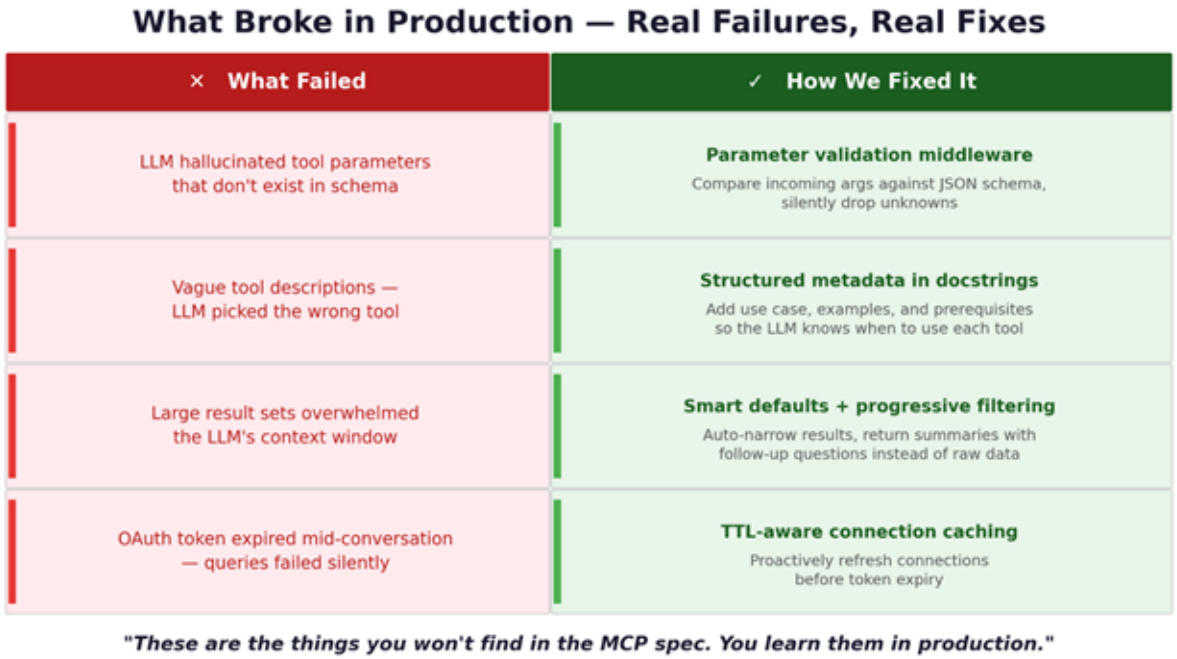
"The more specific your description, the fewer wrong calls you debug in production."

Tool Descriptions & Prompts



Red Hat

What Broke in Production



Red Hat

Key Takeaways

1. **MCP solves the N×M problem**

   Build one server, connect any agent. Build one client, use any server. The ecosystem compounds.

2. **Keep local tools for your domain, MCP for everything else**

   MCP isn't about replacing all tools. It's about creating ownership boundaries.

3. **Auth belongs at the transport layer**

   Token in HTTP headers, not tool parameters. The LLM should never control who is searching.

4. **Tool descriptions ARE your UX**

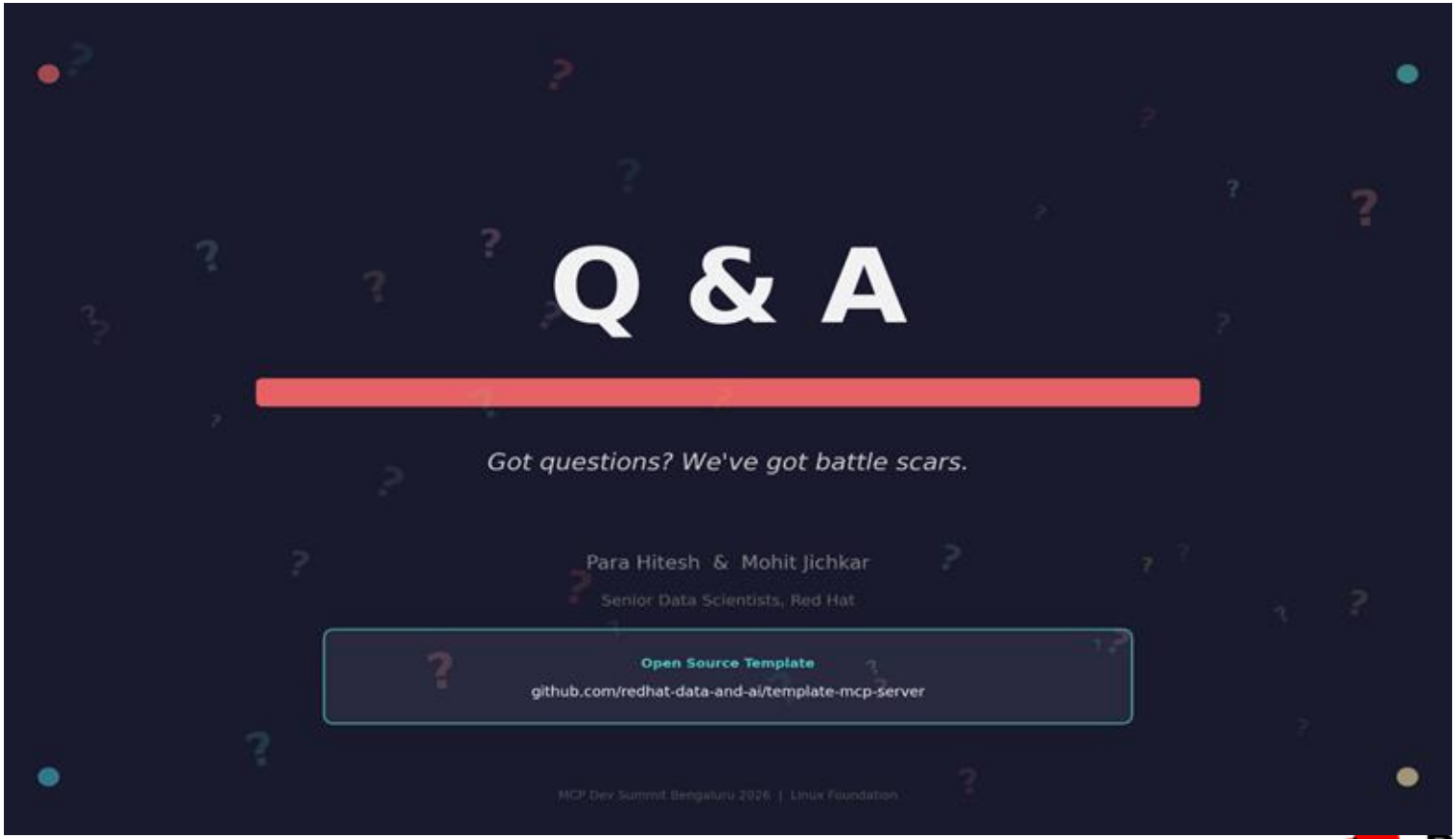
   Multi-operation tools, normalize-before-search, guard against hallucinated parameters.

5. **Use templates, build fast**

   Open source: github.com/redhat-data-and-ai/template-mcp-server

Red Hat

Q&A


Q & A
Got questions? We've got battle scars.
Para Hitesh & Mohit Jichkar
Senior Data Scientists, Red Hat
Open Source Template
github.com/redhat-data-and-ai/template-mcp-server
MCP Dev Summit Bengaluru 2026 | Linux Foundation

Red Hat