APPLIED AND GENERATIVE AI

Building Agentic AI with the Model Context Protocol (MCP)

Level: Practitioner • 2 days (expandable to 3) • Virtual, In-person

Overview

An AI agent is only as useful as the tools and data it can reach. The Model Context Protocol (MCP) is the emerging standard for making that connection: instead of writing bespoke glue for every tool and data source, you expose them through a common protocol that any MCP-aware agent can use. This course teaches you to build with it.

This is a hands-on, practitioner course. It follows the order that makes MCP click. We start with the problem MCP exists to solve and a clear model of how it works, its architecture and its handful of core primitives. Then we deliberately learn to consume existing MCP servers before building any, because using the protocol from the outside is the fastest way to understand it. Only then do we build our own server, add richer context, and finally wire an agent to it safely. Rather than tour the entire specification, we go deep enough on tools, resources, and prompts to build something real, and we treat security and permissions as first-class, not an afterthought. Every module ends with hands-on work and builds on the one before.

Who Should Attend

- Developers building AI agents that need to use tools and reach real data
- Engineers integrating LLMs with internal systems
- Technical leads evaluating MCP as a standard for their teams

Prerequisites

- Comfortable building applications in a language with an MCP SDK, such as Python or TypeScript
- Familiar with calling an LLM API and the basic idea of tool or function calling
- *Introduction to Agentic AI*, or equivalent experience with agents, is helpful

What You Will Learn

- Explain what MCP standardizes and where it fits alongside function calling and agent frameworks
- Describe MCP's architecture and its tool, resource, and prompt primitives
- Configure a client to consume one or more existing MCP servers
- Build and test your own MCP server that exposes a useful tool
- Add resources and prompts to give an agent richer context
- Wire an agent to complete a real task through MCP, with permissions and error handling

Course Outline

Day one: fundamentals and consuming MCP

- Why the Model Context Protocol
  - The integration problem: agents need tools, data, and context

  - What MCP standardizes, and why a standard beats bespoke glue
  - Where MCP fits alongside plain function calling and agent frameworks
  - Lab: connect an agent to an existing MCP server and watch it use a tool
- How MCP Works
  - Architecture: hosts, clients, and servers
  - The core primitives: tools, resources, and prompts
  - Transports and the request and response lifecycle
  - Lab: inspect the messages exchanged between a client and a server
- Consuming MCP Servers
  - Finding, running, and trusting existing servers
  - Configuring a client to use one or more servers
  - Understanding what a server exposes and how to call it
  - Lab: compose two existing servers into a single agent's toolset

Day two: building servers and agents

- Building Your First MCP Server
  - Project setup with the SDK
  - Exposing a tool: inputs, outputs, and descriptions the model can actually use
  - Testing the server in isolation
  - Lab: build and test a server that exposes a useful tool
- Resources, Prompts, and Richer Context
  - Exposing data to the agent as resources
  - Reusable prompts as a server primitive
  - Giving the agent the context it needs without overloading it
  - Lab: extend your server with a resource and a prompt
- Wiring an Agent to MCP, Safely
  - Driving a multi-step task through MCP tools
  - Permissions and untrusted tools: what the agent should and should not be allowed to do
  - Authentication, error handling, and failure recovery
  - Lab: build an agent that completes a real task through your server, with guardrails

Extended Version

The three-day version keeps the same gradient and adds room to build for a team:

- Remote MCP servers and deploying a server others can use
- Stateful and longer-running tools
- Testing and observability for MCP-based agents
- A capstone that designs and builds a server plus agent for a realistic internal use case