MULE AI FORGE

CURRICULUM – 5 WEEKS / 25 DAYS / 75 MICRO-LESSONS

WHAT FORGE TEACHES

The agentic backend on MuleSoft: agents, the tool layer and orchestration. No front end. The interface of these solutions is not a chat window. It is an API, a business event and another agent over A2A. This is the difference from AI courses for application developers: there the agent ends in a chat window, here it plugs into a process that already runs in the company.

OUT OF SCOPE

RAG and vector databases · multimodality · voice interfaces · generative UI · private knowledge bases · agent evaluation · fine-tuning.

This is a deliberate choice, not an oversight. Some of it belongs to the front end, and some of it belongs to building AI applications rather than agentic integrations.

25	75	25	5
DAYS OF MATERIAL	MICRO-LESSONS OF 20-30 MIN	ARTIFACTS FOR YOUR TOOLBOX	LIVE SESSIONS

OVERVIEW

FIVE WEEKS, ONE SYSTEM

Every week adds a layer to the same project instead of opening a new topic. A day is three micro-lessons, one assignment and one artifact that stays in your toolbox.

WEEK 1	**HOW AN AGENT WORKS AND HOW TO DESIGN ONE** You run an agent on the platform on day two, you learn to define its scope and its instruction, and at the end of the week you open the black box and see what the platform does for you.
WEEK 2	**WHAT THE AGENT WORKS WITH AND WHAT IT SEES** The tool layer built on MCP, deliberate control over what reaches the model, and an agent that is started by an event instead of a person.
WEEK 3	**BEFORE IT SHIPS: VISIBILITY, CONTROL AND COST** The agent leaves your laptop. Every day still ends with something you built, only now it is published, secured and priced.
WEEK 4	**WHEN ONE AGENT IS NOT ENOUGH** Splitting work between agents and sub-agents, a protocol that carries state, orchestration as a graph, and context across the network.
WEEK 5	**FROM PROJECT TO CLIENT DELIVERY** The blueprint, the automation, the operations and the conversation that sells the solution.

WEEK 1

HOW AN AGENT WORKS AND HOW TO DESIGN ONE

You run an agent on the platform on day two, you learn to define its scope and its instruction, and at the end of the week you open the black box and see what the platform does for you.

DAY 1

The model as an API

Stop treating the LLM as a chat. Start treating it as a dependency in your architecture, with a contract, latency, cost and a failure mode

- **A.** The LLM as an endpoint
- **B.** Tokens, cost and latency
- **C.** Choosing a model

DAY 2

Your first agent and the Agent Network project

You create an Agent Network project and run one agent inside it, because even a single agent needs that project

- **A.** The Agent Network project and your first agent node
- **B.** Running and testing the agent
- **C.** MuleSoft Vibes as an assistant, not an author

DAY 3

Designing an agent: scope and instruction

The agent stops being a prompt and becomes a component with a defined responsibility and clear limits

- **A.** Scope of responsibility
- **B.** The instruction as a contract
- **C.** Dynamic instructions

DAY 4

Structured output: the data contract

The agent stops returning text and starts returning data your flow can rely on

- **A.** Why text is a poor interface
- **B.** Designing a schema for the model
- **C.** Defensive model calls

DAY 5

Under the hood: function calling and the agent loop

Open the black box. Understand the mechanism the platform hides from you, so that you can fix it

- **A.** How function calling works
- **B.** Anatomy of an agent harness
- **C.** Workflow or orchestration: your first decision

WEEK 2

WHAT THE AGENT WORKS WITH AND WHAT IT SEES

The tool layer built on MCP, deliberate control over what reaches the model, and an agent that is started by an event instead of a person.

DAY 1

MCP: why another standard

Understand the problem MCP solves, and know when it is not worth using

- **A.** The M×N problem
- **B.** Anatomy of the protocol
- **C.** Transports: why MuleSoft has only one

DAY 2

Build an MCP server from zero

Your own working server with tools and resources, tested without an agent

- **A.** A spec-driven start
- **B.** Implementation: tools and resources
- **C.** Testing with Inspector and Playground

DAY 3

Agent context and memory

Stop sending the model everything you have, and start designing what the model sees

- **A.** The context window as a resource
- **B.** Context management strategies
- **C.** Long-term memory

DAY 4

Agents in an event-driven world

The agent stops waiting for a person. It reacts to events, reports progress and asks for the data it does not have

- **A.** What a tool tells the agent between calls
- **B.** Agents started by events
- **C.** Human in the loop

DAY 5

The tool layer at scale: Bridge and tool design

The API portfolio of your client becomes the tool layer, and it still works when there are a hundred tools, not three

- **A.** MCP Bridge in practice
- **B.** API design is not tool design
- **C.** A hundred tools and five servers

WEEK 3

BEFORE IT SHIPS: VISIBILITY, CONTROL AND COST

The agent leaves your laptop. Every day still ends with something you built, only now it is published, secured and priced.

DAY 1

Discovery: what you already have

Understand that your client already has agents. Nobody knows about them yet

- **A.** The registry and publishing
- **B.** The agent card: a light start with A2A
- **C.** Scanners and the trusted catalog

DAY 2

Omni Gateway: access governance

Traffic to your agents and tools stops being uncontrolled

- **A.** A gateway for agents
- **B.** Trusted Agent Identity
- **C.** Policies

DAY 3

Security of an agentic solution

You learn how an agent gets attacked, because you attack your own

- **A.** Identity and permissions
- **B.** Prompt injection and tool poisoning
- **C.** Defence patterns

DAY 4

AI/LLM Gateway: content, cost and provider control

The model stops being a direct dependency of the agent, and the choice of provider stops being permanent

- **A.** Vendor lock-in and the architectural boundary
- **B.** AI policies
- **C.** Semantic routing: the right model for the task

DAY 5

The economics of an agentic solution

You can price an agent before the client asks about the bill

- **A.** Credits and token usage
- **B.** Cost reduction patterns
- **C.** Budgets, alerts and usage monitoring

WEEK 4

WHEN ONE AGENT IS NOT ENOUGH

Splitting work between agents and sub-agents, a protocol that carries state, orchestration as a graph, and context across the network.

DAY 1

Splitting the work: agents, sub-agents, broker

You know when to add a tool, when to add a sub-agent and when to add a broker, and you can explain the choice

- **A.** Three ways to extend an agent
- **B.** The broker and agent-network.yaml
- **C.** One agent calls another

DAY 2

A2A 1.0 in detail

Communication between agents as a protocol with state, not as a function call

- **A.** The task lifecycle
- **B.** contextId and taskId
- **C.** A mixed network

DAY 3

Agent Script: the graph as code

Orchestration stops being prose inside a prompt and becomes an artifact you can review

- **A.** Nodes, edges and triggers
- **B.** Code next to the canvas
- **C.** Vibes on the graph and the audit checklist

DAY 4

Guided determinism in practice

The decision from week 1 applied to a real broker

- **A.** Probabilistic nodes
- **B.** Deterministic nodes and workflow agents
- **C.** Hybrid patterns

DAY 5

Context and memory across the network

Context stops being a problem of one agent and becomes a problem of the architecture

- **A.** Who sees what
- **B.** Memory and cost across the network
- **C.** Agent Visualizer and monitoring

WEEK 5

FROM PROJECT TO CLIENT DELIVERY

The blueprint, the automation, the operations and the conversation that sells the solution.

DAY 1

A blueprint for agentic architecture on MuleSoft

You get a decision tree, not a layer diagram, and you know when to advise a client against using AI

- **A.** When to use what
- **B.** The default solution skeleton
- **C.** A one-page blueprint and the practitioner toolkit

DAY 2

Anypoint CLI: build, publish, deploy

No more clicking. The agent network becomes an artifact you deploy with a command

- **A.** The artifact lifecycle
- **B.** The Anypoint CLI plugin
- **C.** Per-environment configuration

DAY 3

CI/CD for agentic workloads

A repeatable deployment instead of manual work once per project

- **A.** The pipeline
- **B.** What you test before deployment
- **C.** Rollback and change

DAY 4

Resilience and operations in production

A solution that fails in a predictable way and can be handed over to the operations team of your client

- **A.** Failure modes of an agent network
- **B.** Privacy, compliance and the production checklist
- **C.** Runbook and alerts

DAY 5

Summary: what you built and what you do next

You stop seeing 25 assignments and start seeing one system and one skill, and you know what to do on Monday after the course

- **A.** The path you took
- **B.** Your toolbox
- **C.** What comes next