

2 DAYS / LIVE INSTRUCTOR-LED, VIRTUAL OR ONSITE

Applied AI Engineering

From Architecture to Implementation

Move from knowing the patterns to building with them.

Your team can call a model. The next challenge is engineering the application around it: connecting the right data, controlling actions, testing outcomes, and handling the moment a dependency or model response fails.

This two-day course turns architectural understanding into implementation practice. Work through retrieval, structured output, tools, bounded agent workflows, and evaluation. Compare execution paths and inspect the tradeoffs instead of treating a fluent response as proof of success.

The course emphasizes the engineering habits teams need to move a pilot forward: explicit contracts, observable behavior, repeatable evaluation, and controlled failure. Build a stronger basis for deciding what to improve, what to automate, and what still requires human oversight.

In this course, you will learn how to

- Implement a model-backed application path with explicit inputs, outputs, and validation.
- Connect enterprise knowledge through retrieval and check the evidence behind an answer.
- Integrate tools with permission checks, approval, and protection against duplicate actions.
- Build bounded workflows or agent loops with persistent state and stopping limits.
- Create evaluations that test output quality and consequential application behavior.
- Compare routing options and identify security, reliability, and cost gaps before expanding a pilot.

AUDIENCE + COURSE AGENDA

Make the time count.

Who should attend

Software engineers, technical leads, platform and DevOps engineers, solution architects, and technical consultants who will build, integrate, test, or operate AI-enabled applications.

Learning experience

Hands-on implementation, guided troubleshooting, and evidence-based review. Activities build on a shared application scenario. Activities are tailored during scoping to your team’s goals.

Course agenda

Topics covered
Application foundation and enterprise retrieval
Controlled tool use and bounded workflows
Evaluation and model routing
Security, failure testing, and readiness review

Topic order, pacing, and time spent on each subject vary with the group’s experience, questions, and learning needs. Topics are not assigned to fixed times or days. Plan to attend the full course.

What you take back to work

Implementation experience, evaluation approaches, and a practical list of the evidence and controls your team needs for its next pilot iteration.

Explore the three-day AI Engineering Boot Camp to combine architecture and implementation in one program.

Plan private training at www.attainable.tech/contact.html.

DETAILED COURSE OUTLINE

What you will explore.

Part 1: Build the application around the model

Begin with a bounded use case and implement a model-backed application path. Make inputs, outputs, and failure behavior explicit so the model can be tested as one component of a larger system.

- Task boundaries, request flow, and model-provider integration
- Prompt configuration and structured output schemas
- Baseline behavior, validation, and error handling

Part 2: Connect enterprise knowledge with RAG

Implement retrieval that supplies relevant evidence to a model, then examine how changes to context affect the response. Compare a document-based answer with a lookup from an authoritative application source.

- Embeddings, indexing, chunking, and retrieval
- Authorized context, citations, and source traceability
- Missing evidence, poor retrieval, and grounded-answer checks

Part 3: Turn recommendations into safe tool calls

Connect the model to a narrowly scoped application operation. Keep permission checks and business rules in code, and make approval apply to the action that will actually execute.

- Tool definitions, schema validation, and application APIs
- Read and write permissions; human approval boundaries
- Idempotency, retries, and partial-failure handling

Part 4: Build bounded workflows and agent loops

Implement the next-step logic for a controlled workflow or agent. Explore where MCP helps expose tools, and how persistent state and limits prevent a restart from repeating an action or resetting a budget.

- Fixed orchestration versus model-selected next steps
- Direct API and MCP integration boundaries
- Durable progress, context management, budgets, and stop conditions

DETAILED COURSE OUTLINE / CONTINUED

Connect learning to practice.

Part 5: Make quality measurable

Build an evaluation set and compare the application against explicit acceptance criteria. Combine exact checks with rubric-based review while keeping severe failures visible instead of averaging them away.

- Representative examples and independently checked expected results
- Deterministic assertions and AI-reviewer rubrics
- Regression evaluation, failure categories, and human review

Part 6: Route for quality, latency, and cost

Compare execution paths for the same task and implement a routing policy. Account for the whole workflow, including retrieval, tool use, retries, and stronger-model review.

- Code, smaller models, and frontier-model paths
- Observable escalation triggers and fallback behavior
- Measure per-task quality, latency, and total cost

Part 7: Exercise security and failure behavior

Challenge the implementation with hostile content and broken dependencies. Check whether the surrounding application preserves its boundaries when the model is persuaded, uncertain, interrupted, or unavailable.

- Prompt injection, unauthorized access, and unsafe tool requests
- Timeouts, repeated requests, partial completion, and recovery
- Audit evidence, graceful degradation, and operational limits

Part 8: Review readiness and plan the next iteration

Bring the application path, evaluation evidence, and operating controls together. Identify what the training implementation demonstrates and what still needs to be proven before use in your organization.

- Trace behavior across model, data, tools, and validation
- Version prompts and configuration; rerun tests after changes
- Readiness gaps, pilot acceptance criteria, and next engineering steps