Faysal Aziz

Hard Tech with Claude

Build production-grade AI agents, coding systems, API workflows, and Kafka-driven multi-agent automation with Claude in 2 days

FA-0239 · 2 days

Agentic & Generative AI · Software Development · Automation, RPA & Power Platform

---

Introduction

Claude is no longer just a chat interface; it is becoming an engineering substrate for coding, automation, tool execution, workflow orchestration, and agentic systems. This 2-day hardcore program is designed for technical teams that need to move beyond prompts and into real implementation: APIs, Claude Code, agent loops, MCP, workflow automation, secure tool use, and Kafka-style multi-agent coordination inspired by Anthropic's published coordination patterns.

The instructor has over 30 years of industry experience and will focus on real industry-demanded implementation patterns rather than academic theory. Current Claude capabilities covered include tool use, Agent SDK, Claude Code workflows, GitHub Actions, routines, hooks, MCP, prompt caching, batch processing, and Sonnet 5 migration considerations.

Learning outcomes

By the end of this course, participants will be able to:

- Design Claude-powered agentic systems for real engineering workflows
- Use Claude Code for codebase analysis, bug fixing, refactoring, test generation, and pull request automation
- Build Claude API integrations using Messages API, tools, structured outputs, and streaming concepts
- Implement safe tool-calling workflows with approval, execution, validation, and observability
- Design multi-agent systems using coordinator, specialist, evaluator, and worker-agent patterns
- Integrate Claude agents with Kafka-based event-driven architectures
- Use MCP to connect Claude to internal tools, databases, browsers, APIs, and enterprise systems

- Automate engineering workflows using Claude Code hooks, routines, GitHub Actions, and CI/CD pipelines
- Apply prompt caching, batching, rate-limit awareness, and cost-control strategies
- Establish production governance for security, auditability, human approval, and failure handling

Prerequisites

- Strong programming experience in Python, TypeScript, Java, or similar
- Working knowledge of REST APIs and JSON
- Familiarity with Git, GitHub, CI/CD, and command-line workflows
- Basic understanding of event-driven architecture
- Basic Kafka knowledge, including topics, producers, consumers, consumer groups, offsets, and schemas
- Familiarity with Docker or containerized development environments
- Basic understanding of cloud or enterprise application architecture

Training outline

1. Claude Platform for Engineering Teams

Claude as an engineering automation platform

1. Claude models and capability selection
2. Claude Sonnet 5 considerations
3. Claude Opus-class use cases
4. Haiku-class use cases
5. Coding and agentic workload suitability
6. Token budgeting and context planning
7. Model migration considerations
8. Adaptive thinking considerations
9. API behavior changes and compatibility concerns

Claude ecosystem overview

1. Claude web interface
2. Claude Code
3. Claude API

4. Claude Agent SDK
5. Model Context Protocol
6. Claude Code GitHub Actions
7. Claude Code routines
8. Claude Code hooks
9. Files, vision, search, and tool integrations

Enterprise adoption considerations

1. Developer productivity use cases
2. Engineering automation use cases
3. Platform engineering use cases
4. DevOps and SRE use cases
5. Security review use cases
6. Data and analytics use cases
7. Governance and approval boundaries
8. Human-in-the-loop operating models

2. Claude API Core Engineering

Messages API architecture

1. Request structure
2. System prompts
3. User messages
4. Assistant responses
5. Content blocks
6. Stop reasons
7. Token limits
8. Error handling
9. Retry strategies
10. Idempotency considerations

API client implementation

1. Python SDK usage
2. TypeScript SDK usage

3. Environment configuration
4. API key management
5. Secure secrets handling
6. Request tracing
7. Timeout handling
8. Backoff strategies
9. Production logging

Response control

1. Structured outputs
2. JSON response design
3. Schema-oriented prompting
4. Deterministic response contracts
5. Validation layers
6. Parsing failures
7. Recovery prompts
8. Output normalization

Streaming and interaction design

1. Streaming response lifecycle
2. Partial output handling
3. User interface streaming
4. Backend streaming patterns
5. Tool-use streaming considerations
6. Cancellation handling
7. Long-running request handling

Cost and throughput optimization

1. Token counting
2. Prompt caching
3. Cache breakpoints
4. Batch processing
5. Rate-limit planning
6. Throughput modeling

7. Cost estimation
8. Usage monitoring
9. Model selection economics

3. Tool Use and Function Calling with Claude

Tool-use fundamentals

1. Client tools
2. Server tools
3. Tool definitions
4. Input schemas
5. Tool descriptions
6. Tool selection behavior
7. Tool-use stop reasons
8. Tool result submission
9. Multi-step tool loops

Designing production tools

1. Tool naming conventions
2. Minimal input schemas
3. Strict validation
4. Safe defaults
5. Permission boundaries
6. Read-only versus write-capable tools
7. Tool result shaping
8. Error result design
9. Tool observability

Agent loop implementation

1. User request intake
2. Claude reasoning step
3. Tool-call extraction
4. Application-side execution
5. Tool result return

6. Iterative continuation
7. Completion detection
8. Loop guardrails
9. Max-iteration controls
10. Timeout controls

4. Claude Code for Hardcore Development Workflows

Claude Code operating model

1. Terminal workflow
2. IDE workflow
3. Browser workflow
4. Repository context loading
5. File reading
6. File editing
7. Command execution
8. Multi-file changes
9. Project conventions

Codebase understanding

1. Architecture discovery
2. Dependency mapping
3. Module analysis
4. Service boundary review
5. Data model discovery
6. API route discovery
7. Authentication flow review
8. Build system analysis
9. Legacy code assessment

Coding workflows

1. Feature implementation
2. Bug fixing
3. Refactoring

4. Test generation
5. Test repair
6. Documentation updates
7. Code review preparation
8. Pull request creation
9. Release note generation

Advanced Claude Code practices

1. CLAUDE.md project instructions
2. Context management
3. Plan-before-edit workflow
4. Parallel worktrees
5. Resuming previous sessions
6. Delegating research to subagents
7. Piping Claude into scripts
8. CI and batch usage
9. Monorepo considerations

5. Claude Agent SDK and Programmatic Agents

Agent SDK fundamentals

1. SDK architecture
2. Query interface
3. Client interface
4. Message streaming
5. Agent options
6. Tool permission controls
7. Subagent context tracking
8. Parent tool-use identifiers
9. Result handling

Building custom agents

1. Research agent
2. Coding agent

3. Test agent
4. Review agent
5. Documentation agent
6. Deployment assistant agent
7. Incident triage agent
8. Data extraction agent
9. Workflow coordinator agent

Agent configuration

1. System instructions
2. Tool selection
3. Allowed tools
4. Denied tools
5. Approval requirements
6. Working directory scope
7. Environment variables
8. MCP server connections
9. Execution sandboxing

Agent orchestration

1. Single-agent orchestration
2. Multi-agent orchestration
3. Coordinator-worker pattern
4. Planner-executor pattern
5. Reviewer-approver pattern
6. Critic-refiner pattern
7. Router-specialist pattern
8. Consensus pattern
9. Escalation pattern

6. Workflow Automation with Claude

Automation design principles

1. Repeatable workflows

2. Deterministic boundaries
3. Human approval insertion
4. Event-driven triggers
5. Scheduled execution
6. API-triggered execution
7. GitHub-triggered execution
8. Auditability requirements
9. Failure recovery paths

Claude Code hooks

1. Hook lifecycle
2. Pre-tool hooks
3. Post-tool hooks
4. Notification hooks
5. Edit and write hooks
6. Formatting automation
7. Validation automation
8. Security checks
9. Policy enforcement
10. Async hook considerations
11. MCP tool hooks

Claude Code routines

1. Routine definition
2. Scheduled routines
3. API-triggered routines
4. GitHub event routines
5. Repository binding
6. Connector configuration
7. Branch permissions
8. Run management
9. Usage limits
10. Research preview considerations

7. Model Context Protocol Integration

MCP fundamentals

1. MCP purpose
2. MCP server architecture
3. MCP client architecture
4. Tool exposure model
5. Resource exposure model
6. Prompt exposure model
7. Local MCP servers
8. Remote MCP servers
9. Enterprise MCP gateways

Claude with MCP

1. Claude Code MCP integration
2. Agent SDK MCP integration
3. Browser automation MCP
4. Database MCP
5. API gateway MCP
6. Monitoring system MCP
7. Issue tracker MCP
8. Documentation MCP
9. Internal platform MCP

MCP security and operations

1. Authentication
2. Authorization
3. Tool-level permissions
4. Data boundary controls
5. Network restrictions
6. Tenant isolation
7. Audit logging
8. Connector lifecycle management
9. MCP server hardening

8. Multi-Agent Architecture with Claude

Multi-agent design foundations

1. Agent roles
2. Agent responsibilities
3. Shared state
4. Task decomposition
5. Message passing
6. Coordination strategy
7. Conflict resolution
8. Result aggregation
9. Completion criteria

Coordination patterns

1. Central coordinator pattern
2. Decentralized collaboration pattern
3. Supervisor-worker pattern
4. Specialist swarm pattern
5. Pipeline pattern
6. Debate and critique pattern
7. Evaluator pattern
8. Human approval pattern
9. Failover agent pattern

Agent communication contracts

1. Task envelopes
2. Message schemas
3. Correlation identifiers
4. Conversation identifiers
5. Parent-child task identifiers
6. Status events
7. Result payloads
8. Error payloads
9. Retry metadata

Agent memory and state

1. Short-term context
2. Long-term task state
3. External state stores
4. Vector memory considerations
5. Relational state stores
6. Object storage
7. Event logs
8. Replayable state
9. State compaction

9. Claude with Kafka for Event-Driven Multi-Agent Systems

Kafka-based agent architecture

1. Event-driven agent coordination
2. Claude as reasoning worker
3. Kafka as coordination backbone
4. Topic-per-agent pattern
5. Topic-per-task pattern
6. Shared command topic
7. Result topic
8. Dead-letter topic
9. Audit topic

Kafka topic design

1. Task request topics
2. Agent command topics
3. Tool execution topics
4. Tool result topics
5. Agent status topics
6. Human approval topics
7. Evaluation topics
8. Final response topics
9. Error and retry topics

Producer and consumer design

1. Task producer services
2. Agent consumer services
3. Consumer groups
4. Partitioning strategy
5. Ordering guarantees
6. Key selection
7. Offset management
8. Commit strategy
9. Backpressure handling

Multi-agent coordination over Kafka

1. Coordinator agent service
2. Planner agent service
3. Coding agent service
4. Test agent service
5. Reviewer agent service
6. Security agent service
7. Documentation agent service
8. Deployment agent service
9. Evaluator agent service

Kafka message schemas

1. Task schema
2. Agent instruction schema
3. Tool-call schema
4. Tool-result schema
5. Approval-request schema
6. Approval-response schema
7. Agent-status schema
8. Error schema
9. Completion schema

Observability for Kafka agents

1. Correlation IDs
2. Trace propagation
3. Agent execution logs
4. Tool-call logs
5. Token usage metrics
6. Latency metrics
7. Failure metrics
8. Kafka lag monitoring
9. Cost dashboards

10. Production-Grade Claude Agent Systems

Reference architecture

1. API gateway
2. Orchestrator service
3. Agent worker services
4. Tool execution services
5. Kafka backbone
6. State store
7. Vector store
8. Audit store
9. Human approval interface
10. Observability stack

Disclaimer

This training outline is provided as a professional guideline for planning and delivery purposes. The trainer reserves the right to amend, restructure, expand, reduce, substitute, or otherwise modify any portion of the outline, sequence, emphasis, tools, demonstrations, or coverage areas as deemed appropriate based on participant readiness, organizational priorities, technology updates, time constraints, and classroom conditions, without prior notice.

© 2026 Faysal Aziz · Kuala Lumpur

https://faysalaziz.com/courses/hard-tech-with-claude/