GENERATIVE AI · TAUGHT WITH CLAUDE

Generative AI Engineering

The craft under every AI product: streaming, tool calls, structured output, caching, and cost. You learn where the token bill and the latency actually come from, and ship something people can rely on.

Advanced | 4 phases | 25 sessions | 38 contact hours

Syllabus

4 phases, 25 sessions. Each phase ends in something you have built.

1 The Wire

Drive the Messages API directly and account for every field in the request and the response.

6 sessions
$108 alone

1.1	**Messages, Not Chat**	One stateless endpoint, the whole history resent every turn, and where the system prompt lives. FREE PREVIEW
1.2	**Streaming, Event by Event**	message_start to message_stop, deltas accumulated by hand before you trust the SDK helper.
1.3	**Every Stop Reason**	end_turn, max_tokens, tool_use, pause_turn, refusal, and the branch most code forgets.
1.4	**Thinking and Effort**	Adaptive thinking, the effort ladder from low to max, and why budget_tokens now returns 400.
1.5	**Counting Tokens Honestly**	The count_tokens endpoint, why tiktoken undercounts Claude, and reading usage on every call.
1.6	**Errors That Retry**	429 and 529 back off, 400 never does, and retry-after beats your own exponential guess.

By the end: **A streaming CLI client with correct stop-reason branching, bounded retries, and per-call token accounting.**

2 Data, Not Prose

Make the model return typed data your code consumes without a regex or a hope.

6 sessions
$108 alone

2.1	**Structured Outputs**	output_config.format with a JSON schema, and why output_format and assistant prefill are gone.
2.2	**Schemas That Compile**	The supported JSON Schema subset, additionalProperties false, and the first-request compile cost.
2.3	**Tool Definitions**	Name, description, input_schema, and descriptions that say when to call, not just what.
2.4	**Strict Mode**	strict true on the tool, parsing input as JSON, and never matching a serialized argument.
2.5	**Parallel Tool Calls**	Several tool_use blocks in one turn, and all results returned in a single user message.
2.6	**When The Shape Breaks**	Truncation, refusals, and the validation error you count instead of silently swallow.

By the end: **An extraction service that validates against a schema and reports its own failure rate per field.**

3 Context and Cost

Cut spend and latency on a real workload without moving output quality.

7 sessions
$108 alone

3.1	**The Prefix Is The Key**	Render order of tools, system, messages, and how one timestamp invalidates everything after.
3.2	**Breakpoints and TTLs**	Four cache_control markers, 5 minute against 1 hour, and the per-model minimum that fails quietly.
3.3	**Auditing a Cache Miss**	cache_read_input_tokens at zero, unsorted JSON keys, and a tool list that varies by user.
3.4	**Budgeting the Window**	A million tokens is a budget, not a bucket, plus context editing and server-side compaction.
3.5	**The Batch Lane**	Half price, a 24 hour ceiling, and results keyed by custom_id because order is not promised.
3.6	**Picking a Model**	Opus 5 against Sonnet 5 and Haiku 4.5, measured on your own task, not on a leaderboard.
3.7	**The One Second Budget**	Time to first token, prefill against generation, and the 200ms you are never getting back.

By the end: **A before-and-after benchmark reporting cache hit rate, p95 time to first token, and cost per request.**

4 Shipping It

Treat the prompt as a versioned artifact and know what every request cost, took, and returned.

6 sessions
$108 alone

4.1 Prompts as Artifacts

Prompts in git with immutable version ids, pinned per environment, rolled back by pointer.

4.2 Regression Before Deploy

A fixture set, a diff against the last version, and the gate that blocks a bad prompt.

4.3 Tracing a Request

OpenTelemetry spans carrying prompt version, model, tokens, cache hits, and dollars.

4.4 Cost Per Success

Per-call dollars, retries that quietly double spend, and the number that actually matters.

4.5 Streaming to a Browser

SSE over HTTP/2, proxy buffering, reconnects, and a half-written answer left on screen.

4.6 The Failure Runbook

Rate limits, overload, schema drift, and what your feature does when the API is down.

By the end: **A deployed feature with pinned prompt versions, traced requests, and a rollback you have actually performed.**

Tools you will use

Claude Opus 5 | Claude Sonnet 5 | Claude Haiku 4.5 | anthropic Python SDK 0.120 | Claude Code | Pydantic 2 | FastAPI | OpenTelemetry GenAI semantic conventions | Langfuse | promptfoo

What you will build

1. **Streaming Extraction Service**
   Typed JSON from free text, streamed, with a measured schema failure rate
2. **The Cost Teardown**
   One workload, caching and batch applied, spend and p95 latency cut on record
3. **A Feature People Depend On**
   Pinned prompt versions, traced requests, and a rollback performed under load

Registration

Phase 1: The Wire	$108
Phase 2: Data, Not Prose	$108
Phase 3: Context and Cost	$108
Phase 4: Shipping It	$108
Whole track, all 4 phases	**$360**

Buying every phase separately costs $432, so the whole track saves $72.

No payment is taken online. Registering creates an order and payment instructions are sent with it. Access opens once payment is confirmed.