ARTIFICIAL INTELLIGENCE | INTERMEDIATE | 5 DAYS

AI Engineering Bootcamp: Building Production LLM Systems

A five-day intensive covering the full AI engineering stack: foundation models, prompt and context engineering, evaluation, retrieval and RAG, tool calling, fine-tuning decisions, inference optimisation, and production deployment with observability. Built around the principle that anyone can generate a demo, and the engineering is everything that happens after. Assumes working Python and API experience.

Why AI Engineering Bootcamp?

Evaluation from day two

Evals are taught before retrieval and agents, because everything downstream depends on them.

Provider-neutral

Works across Anthropic, OpenAI and Gemini rather than locking into one vendor's SDK.

Non-deterministic systems engineering

Treats AI work as distributed systems with a probabilistic component, not prompt tweaking.

Ships something real

Five days ends with a deployed, traced, evaluated service, not a notebook.

What you'll be able to do

- ✓ Explain how foundation models work at the depth needed to make architecture decisions
- ✓ Build against LLM APIs across multiple providers with proper error and cost handling
- ✓ Apply prompt and context engineering to produce reliable structured output
- ✓ Design evaluation suites using golden datasets, rubrics and calibrated LLM judges
- ✓ Diagnose failures in non-deterministic systems using tracing and error taxonomies
- ✓ Build production retrieval systems with hybrid search, reranking and grounded citations
- ✓ Measure retrieval quality separately from generation quality
- ✓ Design tool calling and multi-step workflows, and know when an agent is unnecessary
- ✓ Decide between prompting, retrieval and fine-tuning on evidence rather than fashion
- ✓ Optimise latency and cost through caching, batching, routing and serving choices
- ✓ Deploy with observability, guardrails and cost monitoring in place
- ✓ Apply security and compliance practice appropriate to regulated environments

Who this is for

- Software engineers moving into AI engineering roles
- Backend and platform engineers building LLM features
- Data engineers and ML engineers extending into GenAI
- Technical leads and architects accountable for AI systems in production

Curriculum

01 Foundation Models and What They Actually Do

- Transformers at working depth: attention, tokenisation and generation
- Context windows, the KV cache and why long context is expensive
- Why the same input can produce different output, and what that means for testing
- Model families across providers and how capability, latency and cost trade off
- Choosing a model deliberately rather than defaulting to the largest available

02 Working with LLM APIs Across Providers

- Message-based APIs, parameters and the shape of a request
- Streaming, async patterns and concurrency in Python
- Rate limits, retries, timeouts and graceful degradation
- Abstracting across providers without building a lowest-common-denominator wrapper
- **Hands-on:** build a resilient multi-provider service layer

03 Prompt and Context Engineering

- Structuring prompts: roles, delimiters, examples and explicit directives
- Structured outputs and enforcing schemas the downstream code can trust
- Reasoning and extended thinking modes, and when the tokens are worth it
- Context budgeting: what to include, what to summarise, what to leave out
- **Hands-on:** take an unreliable prompt to consistent structured output

04 Evaluation: Golden Datasets and Judges

- Why evaluation is the discipline that separates prototypes from products
- Building golden datasets from real usage rather than invented cases
- Grading approaches: exact match, rubric-based and model-graded
- Calibrating an LLM judge and validating it against human labels
- **Hands-on:** build an evaluation harness with a golden dataset

05 Failure Analysis and Debugging Probabilistic Systems

- Tracing every model call, retrieval step and tool invocation
- Building an error taxonomy for your own system
- Why an eval starts lying the moment you optimise against it
- Recognising sycophancy and other measurable training side effects
- Regression testing so improvements do not quietly break something else

06 Retrieval Foundations: Embeddings and Vector Stores

- Embedding models and how to choose between them
- Vector database options and the managed against self-hosted decision
- Postgres-native retrieval and when it is entirely sufficient
- Graph stores and where relationship structure beats similarity
- **Hands-on:** index a real corpus and measure retrieval quality

07 Building Production RAG

- Chunking strategies and why naive chunking silently degrades answers
- Hybrid search: combining keyword and dense retrieval
- Reranking and contextual retrieval to lift precision
- Grounded generation with citations users can verify
- Recognising when RAG is the wrong answer entirely
- **Hands-on:** build a RAG pipeline over an internal document set

08 Evaluating Retrieval Quality

- Faithfulness, answer relevancy and context recall as distinct measures
- Separating retrieval failures from generation failures
- Automated RAG scoring and regression suites
- Diagnosing the difference between a bad chunk and a bad prompt
- **Hands-on:** instrument and score the pipeline you built

09 Tool Calling and Workflow Design

- Tool schema design the model can use reliably
- The tool loop, parallel calls and error surfacing
- Workflow patterns: chaining, routing and parallelisation
- Sandboxed code execution and containment
- Deciding when a deterministic workflow beats an agent

10 Fine-Tuning, Dataset Engineering and When to Avoid Both

- Supervised fine-tuning, preference optimisation and reinforcement approaches
- Dataset engineering: collection, cleaning, labelling and splits
- Honest cost-benefit against better prompting or better retrieval
- Evaluating a fine-tune properly before it goes anywhere near production
- Small models, distillation and the cases where they win

11 Inference Optimisation, Cost and Latency

- Prompt caching strategies and designing prompts to be cache-friendly
- Batch processing for non-interactive workloads
- Model routing and tiering work by complexity
- Streaming, perceived latency and output length control
- Self-hosted serving basics and quantisation trade-offs
- **Hands-on:** cut cost and latency on a working service without losing quality

12 Production Architecture

- Service design, queues and handling long-running generation
- Multi-provider resilience and fallback strategy
- Caching layers, idempotency and retry semantics
- Secrets, isolation and safe execution boundaries
- Containerisation and deployment to a cloud target

13 Observability and LLMOps

- Distributed tracing for model calls using open telemetry standards
- Tracing and observability platforms and what to instrument
- Cost attribution per feature, per tenant and per request
- Monitoring quality drift as models and data change underneath you
- Feedback capture and turning production signal into eval cases

14 Security, Guardrails and Compliance

- Prompt injection and indirect injection through retrieved content
- Data leakage, PII handling and output filtering
- Input and output guardrails and where to place them
- Regulatory context: AI act obligations, management-system standards and risk frameworks
- Documenting a system so an auditor or client security review can follow it

15 Capstone Build

- Design and build a grounded, tool-using service end to end
- Attach a golden dataset and a passing evaluation suite
- Instrument tracing, cost monitoring and guardrails
- Present architecture decisions and defend the trade-offs
- Produce a handover document your team could actually operate from