GenAI Engineering Programme

Build GenAI systems that hold up in production.

From transformer internals to RAG, agents, MCP, evaluation and security. 59 focused topics, built for engineers who need to ship it, and defend it.



59 topics

9 tracks

115 hours

53 high-priority topics

Most courses teach you to call an API. This one teaches you what happens when it fails.

Anyone can get a demo working. Teams hire for the engineer who knows why retrieval missed, what the cache is costing, where the agent can be tricked and how to prove a change made things better.

Understand

Attention, KV cache, decoding, tokenization and quantization, taught as the reasons behind behaviour, cost and latency you see in production.

LLM Fundamentals · Model Adaptation

Build

Full RAG pipelines, tool-using agents and MCP integrations, with the design choices and failure modes of each made explicit.

RAG · Agents · MCP

Ship

Evaluation, tracing, caching, routing, rollout and defence against injection, so what you build keeps working after launch.

LLMOps · Safety · System Design

Who it is for

- Software and ML engineers moving into GenAI roles
- Engineers who have built a prototype and now need it production-grade
- Candidates preparing for GenAI engineering and system design interviews
- Tech leads who must make model, cost and architecture calls

You will leave able to

- Design a production RAG system end to end
- Choose between prompting, RAG and fine-tuning with reasons
- Build and evaluate agents with safe, validated tools
- Set up golden sets, regression tests and release gates
- Explain a system you built and defend every trade-off

Nine tracks. One path from tokens to the interview room.

Each track builds on the one before. Bar length shows study hours.



115 hours of focused learning

How we teach: one idea at a time, until you can explain it cold.

Every one of the 59 topics follows the same loop, so you always know what to learn, how well you know it, and when you are ready.

1 **Focus**

One topic per session, about two hours, with a clear brief on what to know.

2 **Explain**

Frame each idea as an interview answer: what it is, when to use it, what breaks.

3 **Rate**

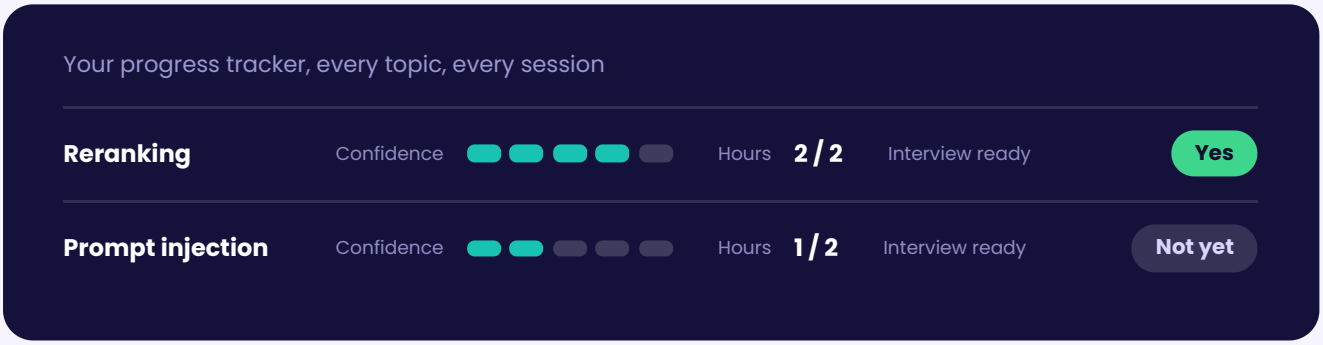
Score your own confidence from 1 to 5 against the outcome, honestly.

4 **Revisit**

Keep notes and a last-reviewed date, so weak spots resurface instead of fading.

5 **Clear the gate**

A topic is marked interview-ready only when you can defend it unprompted.



Priority first

53 topics are high priority and 6 medium. You always know what matters most when time is short.

Trade-offs over trivia

Topics are framed as decisions: dense or sparse, fine-tune or retrieve, one agent or many.

Failure is a topic

RAG failure modes, agent recovery, red teaming and incident handling each get their own session.

Concepts before maths

You learn how things work and why they matter, without drowning in algorithm proofs.

LLM Fundamentals

Open the black box. Know why models behave the way they do.

16 hours
8 topics

- **Transformer architecture** — 2h
  Encoders vs decoders, attention blocks, residuals, FFNs and layer norm
- **Self-attention & Q/K/V** — 2h
  Scaled attention, causal masking, and why long context gets expensive
- **Multi-head attention, GQA & MQA** — 2h
  Why heads exist and how GQA/MQA shrink KV-cache memory and cost
- **RoPE & positional information** — 2h
  Why position matters, how RoPE works, and where long context breaks
- **Tokenization & context windows** — 2h
  BPE and subwords, special tokens, token budgets and window trade-offs
- **Autoregressive decoding** — 2h
  Temperature, top-p, top-k, greedy vs sampling and stopping criteria
- **KV cache, prefill & decode** — 2h
  What the cache stores and why memory bandwidth becomes the bottleneck
- **Hallucinations & grounding** — 2h
  Why models make things up and the practical ways to curb it

Model Adaptation

Pick the right lever: prompt it, retrieve for it, tune it or shrink it.

7.5 hours
4 topics

- **Fine-tuning vs prompting vs RAG** — 2h
  Choose by data, latency, control, freshness and cost
- **LoRA, QLoRA & PEFT** — 2h
  Why adapters cut trainable parameters, and when to use them
- **Quantization** — 2h
  FP16/BF16 vs INT8/INT4: quality, memory, latency and deployment trade-offs
- **RLHF vs DPO** — 1.5h
  The preference-alignment pipeline, and why DPO is simpler to operate

● High priority ○ Medium priority

RAG

The workhorse of enterprise GenAI, from ingestion to evaluation.

25.5 hours
13 topics

- **RAG architecture end-to-end** — 2h
  Ingestion, parsing, chunking, embedding, retrieval, reranking, prompt, generation
- **Chunking strategies** — 2h
  Fixed, recursive and semantic chunking; size and overlap trade-offs
- **Dense vs sparse retrieval** — 2h
  Vector search vs BM25, and when lexical search wins
- **Vector databases & indexes** — 2h
  Collections, filtering, metadata, freshness and ANN concepts
- ○ **Query transformation** — 1.5h
  Rewriting, multi-query and HyDE; when it helps and when it hurts
- **Grounding, citations & hallucination control** — 2h
  Evidence selection, attribution, and refusing when evidence is thin
- **RAG evaluation** — 2h
  Recall@k, MRR/NDCG, faithfulness, context precision and golden datasets
- **Document parsing & metadata** — 2h
  PDF, HTML, Markdown and tables; why ingestion quality decides everything
- **Embeddings & similarity search** — 2h
  What embeddings represent, cosine vs dot product, choosing a model
- **Hybrid search** — 2h
  Combining dense and sparse signals for robust retrieval
- **Reranking** — 2h
  Cross-encoder and LLM rerankers: higher precision, higher latency
- **Context compression & prompt packing** — 2h
  Cut the noise, keep the evidence, fit the window
- **RAG failure modes** — 2h
  Bad parsing, poor chunks, retrieval misses, stale data, injection, synthesis errors

● High priority ○ Medium priority

Agents

Models that act. Make them reliable, bounded and testable.

15.5 hours
8 topics

- **ReAct & the reasoning-action loop** — 2h
  How agents alternate between thinking and tools, and how that loop fails
- **Tool & function calling** — 2h
  Schemas, structured arguments, validation, retries and safe execution
- **Tool design & reliability** — 2h
  Small deterministic tools, idempotency, timeouts, permissions, failure handling
- **Planning & task decomposition** — 2h
  When to plan, how to break work down, how to recover from a bad step
- **Agent state & memory** — 2h
  Short-term vs durable memory, persistence and retrieval
- **Human-in-the-loop** — 2h
  Approval gates for risky actions; automation without losing control
- ○ **Multi-agent systems** — 1.5h
  Routing and delegation, coordination cost, and when one agent is better
- **Agent evaluation & simulation** — 2h
  Task success, tool-call correctness, trajectories and regression tests

MCP

The Model Context Protocol, from architecture to the trust boundary.

7.5 hours
4 topics

- **MCP architecture & lifecycle** — 2h
  Client/server model, sessions, transports and capability negotiation
- **MCP tools, resources & prompts** — 2h
  The three primitives and what each one is for
- **MCP security & authorization** — 2h
  Trust boundaries, untrusted content, auth and least privilege
- ○ **MCP vs APIs and SDKs** — 1.5h
  When MCP adds value over direct function-calling integration

● High priority ○ Medium priority

LLMOps & Evaluation

Measure it, trace it, speed it up, cut its cost, ship it safely.

19.5 hours
10 topics

- **Prompt engineering & structured outputs** — 2h
  System prompts, few-shot, output schemas, constraints and versioning
- **LLM evaluation strategy** — 2h
  Golden sets, deterministic checks, LLM-as-judge and human review
- **Regression testing for LLM apps** — 2h
  Test suites, thresholds and release gates for prompt, model and retriever changes
- **Observability & tracing** — 2h
  Trace model calls, retrieval, tools, latency, tokens, errors and outcomes
- **Latency optimization** — 2h
  Streaming, parallel calls, batching, caching and smaller models
- **Caching** — 2h
  Exact, semantic and prefix caching; correctness and invalidation
- **Model routing & fallbacks** — 2h
  Route by task, cost and latency; fail over across providers
- **Cost & token economics** — 2h
  Token accounting, caching savings, model choice and unit economics
- **Model serving basics** — 2h
  vLLM/TGI-style serving, continuous batching, GPU memory limits
- ○ **Deployment & rollout patterns** — 1.5h
  Canary, shadow and A/B rollouts, versioning and rollback

Safety & Security

Assume an adversary. Design for it.

9.5 hours
5 topics

- **Prompt injection & indirect injection** — 2h
  Direct and indirect attacks, why they work, and layered defences
- **Jailbreaks & model misuse** — 2h
  Common attack classes, mitigations and an evaluation mindset
- **Data leakage & output sanitization** — 2h
  PII and secrets, unsafe tool output, schema validation, redaction
- **Tool permissions & excessive agency** — 2h
  Least privilege, allowlists, approval gates and sandboxing
- ○ **Red teaming & adversarial evaluation** — 1.5h
  Build attack suites, track regressions, test your controls

● High priority ○ Medium priority

System Design

Put every piece together and defend the architecture.

10 hours
5 topics

- **GenAI system design fundamentals** — 2h
  Requirements, models, retrieval, tools, storage, APIs, security, cost, failure
- **Design a conversational or agentic system** — 2h
  State, memory, tools, orchestration, safety, latency and reliability
- **Failure handling & incident management** — 2h
  Timeouts, retries, partial failure, stale indexes, outages and rollback
- **Design a production RAG system** — 2h
  Ingestion, retrieval, evaluation, scaling, freshness and fallbacks
- **Model selection & routing** — 2h
  Hosted vs open models on quality, latency, cost, context and privacy

Interview Readiness

Turn knowledge into answers that land.

4 hours
2 topics

- **Explain a system you built end-to-end** — 2h
  Problem, architecture, trade-offs, metrics, failures and what you would change
- **Trade-offs & technical judgment** — 2h
  Defend choices on RAG vs fine-tuning, model size, latency, cost, accuracy, safety

● High priority ○ Medium priority

Walk in able to design it, build it, break it and defend it.

- Explain how attention, caching and decoding drive cost and latency
- Ship RAG that retrieves the right evidence and cites it
- Build agents with tools that are validated, permissioned and testable
- Prove every change with golden sets and regression gates
- Stand up defences against injection, leakage and excessive agency
- Lead a GenAI system design round with confidence

Reserve your seat

Cohort dates, fees and enrolment: add your details here.

