



Prompt Engineering 2026

From prompts that work once to prompts that work in production

THE OPPORTUNITY

Executive summary

Most professionals now use an AI assistant every week. Very few can say why the same request produced a good answer on Tuesday and a bad one on Wednesday, or how they would prove that changing a prompt made anything better. Prompt Engineering 2026 is a one-day, hands-on workshop for people who have to make AI work in something that matters. It starts from the mechanism, builds the working techniques, and ends with the two artefacts almost nobody has: a scored baseline and a test set they can re-run.

The day is deliberately honest about limits. Where a technique costs more than it returns, the workshop says so. Where a problem cannot be solved by prompting at all, it says that too, and shows what to do instead.

At a glance

Duration	One day · 7 hours of instruction, plus 90 minutes of breaks
Format	Live and hands-on · six labs, 147 minutes at the keyboard
Audience	Business and technical professionals who work with AI tools
Prerequisites	None · a laptop and access to any large language model
Case study	One running scenario, Halcyon Logistics, across all six labs
Checks	Six knowledge checks · 60 questions, marked in the room

How the day builds

1**Baseline**

You write a deliberately lazy prompt, keep the output, and score it. Everything later is measured against that.

2**Build**

You learn the techniques and rebuild the prompt with structure, examples and explicit reasoning.

3**Prove**

You assemble a test set, score the prompt, and improve it one variable at a time.

BUSINESS BENEFIT

Participants leave with a scored baseline, a reusable test set and a hardened system design: the three artefacts that let a colleague change an AI system safely after they have gone.



THE EXPERIENCE

Workshop description

The day runs on one fictional company, Halcyon Logistics, and its 4,000 support tickets a month. Every technique is applied to the same problem, so the room spends its attention on the method rather than on inventing a business context. Participants work in pairs and small groups that change through the day, and each module ends with a ten-question check that the room marks together.

Laptops are open from the first module. The workshop assumes participants already use AI tools and have felt the frustration of results they cannot reproduce, so it spends little time on what the tools are and most of its time on how to make them dependable. No coding is required, and every lab works on a hosted chat interface as well as an API.

Who it is for

- Professionals who already use AI tools and want results they can repeat
- Product, marketing, operations, service, HR, legal, finance and engineering teams
- Managers asked to add AI who need to judge what is realistic and what is not
- Technical practitioners who want the evaluation and security discipline, not only the prompting tips

Four connected learning moves

UNDERSTAND

How a model turns a prompt into an answer, and why that makes wording a design decision rather than a matter of taste.

WRITE

The working techniques: how many examples to give, when to ask for reasoning, and how to build a prompt you can reuse.

COMPOSE

What to do when one prompt is not enough: chains with defined interfaces, branching search, tool use and iteration.

OPERATE

Security, evaluation, cost and governance, so the result survives a model change, a new colleague and an audit.

What participants produce

A scored baseline prompt with its recorded output · a classifier prompt with an explicit output contract · a reusable RFP-answer template · a three-stage chain with a validation step · three documented attacks against their own prompt, each mapped to the control that stops it · a 30-case test set with a baseline score and two single-change deltas · a seven-layer context budget and a seven-clause agent operating contract.

THE CORE MESSAGE

Prompt engineering is the craft. Measurement is the discipline. A prompt you cannot score is a preference, not an engineering decision.



WHAT YOU WILL BE ABLE TO DO

Learning outcomes

Six modules, twenty-four outcomes. Every outcome is practised in a lab on the same day it is taught.

01 Foundations of Prompting 65 minutes • Explain why prompt wording changes output • Work within the real constraints of a model • Separate what prompting can change from what it cannot • Construct a prompt using the anatomy that works	02 Core Prompting Techniques 80 minutes • Choose deliberately between zero-shot, one-shot and few-shot • Build a few-shot set that teaches the right thing • Use chain-of-thought where it earns its cost, and know where it does not • Turn a one-off prompt into a reusable template
03 Advanced and Agentic Patterns 75 minutes • Decompose a task into a prompt chain • Recognise when a task needs search rather than sequence • Build a loop that refines and validates its own output • Choose a pattern on cost, not novelty	04 Prompt Security and Governance 65 minutes • Explain why prompt injection cannot be patched away • Distinguish injection from jailbreaking • Apply defence in depth rather than a clever instruction • Place prompting inside a governance model
05 Optimisation, Evaluation and Cost 55 minutes • Choose between prompting, retrieval and training • Optimise a prompt systematically rather than by taste • Reduce cost and latency without losing quality • Build a test set and score against it	06 The Applied 2026 Landscape 45 minutes • Move from prompt design to context design • Write an operating contract for an agent • Apply the patterns to your own function • Run a prompt library as a product

HOW YOU WILL KNOW IT LANDED

Every module ends with a ten-question check that the room marks together, so you can see which outcomes have taken hold and which need a second look before you leave.



MODULES 1 TO 3

Course outline 1 of 2

01 Foundations of Prompting

65 minutes

Source focus: IBM Think library on generative AI, tokens, context windows, attention and hallucination.

What a large language model is actually doing: predicting the next token rather than looking anything up. Participants trace a prompt through tokenisation, attention and sampling and see why rephrasing changes the answer.

The real constraints then become concrete: what a token is, what a context window costs, and what sampling controls do.

Then the line most teams never draw. Prompting changes behaviour, not knowledge, so hallucination and bias are treated as structural rather than as prompt failures. The module closes with the five-part anatomy and a pre-flight checklist.

Practical application. Participants write a deliberately lazy prompt, keep the output, and score it. Everything in Module 2 is measured against that baseline.

02 Core Prompting Techniques

80 minutes

Source focus: IBM Think prompt-engineering library on zero-shot, one-shot and few-shot prompting, chain-of-thought and in-context learning.

The working toolkit: how many examples to give, when to ask for reasoning, and how to build a prompt you can reuse instead of rewriting.

Participants compare zero-shot, one-shot and few-shot on one task and see the trade-off for what it is. Examples buy consistency with tokens you pay on every call, forever.

Chain-of-thought is taught with its costs attached, separating tasks that need intermediate reasoning from cases where a plausible chain is worse than a direct answer. Structure does double duty, because delimiters become a security control in Module 4.

Practical application. Pairs classify twelve Halcyon support tickets with an explicit output contract, then convert a one-off RFP answer into a template that survives a colleague using it.

03 Advanced and Agentic Patterns

75 minutes

Source focus: IBM Think material on prompt chaining, Tree of Thoughts, meta prompting, ReAct and self-consistency.

What to do when one prompt is not enough. The ceiling first: a prompt that must read, reason, decide and format at once has no opportunity to check its own work.

Prompt chaining decomposes that into stages with defined interfaces. The design lesson is that the contract between stages matters more than the stages themselves.

Tree of Thoughts, meta prompting, ReAct and self-consistency follow, each with its cost attached. A decision guide and a cost ladder close the module, with one discipline: climb only when the rung below is measurably insufficient.

Practical application. Groups build a three-stage chain on Halcyon's operations data (extract, analyse, recommend), then deliberately break it to watch an upstream error propagate.



MODULES 4 TO 6

Course outline 2 of 2

04 Prompt Security and Governance

65 minutes

Source focus: IBM Think material on prompt injection and jailbreaking, NIST's reported findings, and enterprise AI governance.

The module turns the room adversarial. Prompt injection is structural rather than a defect: instructions and data are both text, and the model cannot reliably separate them.

Indirect injection gets the most attention, because it needs no attacker at the keyboard. It needs only a document, page, email or ticket the system reads.

Jailbreaking, leakage, tool abuse and exfiltration follow, then the honest conclusion. IBM states no foolproof fix exists, and even injection detectors can be injected. The response is defence in depth, plus the governance questions of ownership and prohibited data.

Practical application. Pairs attack their own Halcyon classifier through the ticket text, through a poisoned document and through the output, then map each attack to the control that would have stopped it.

05 Optimisation, Evaluation and Cost

55 minutes

Source focus: IBM Think material on prompt optimisation, DSPy, prompt caching, RAG, fine-tuning and evaluation metrics.

The engineering module. Four ways to change a model's behaviour (prompting, retrieval, prompt tuning and fine-tuning) compared on what each changes and what each costs.

Participants meet the recommendation the workshop holds throughout: fix the prompt first, and climb only when the evidence says the rung below is insufficient.

Caching is taught with its limits intact, since IBM's own example shows a cached call taking longer than an uncached one.

Evaluation closes the module with a stratified test set, a metric you cannot game, and re-runs when the model, index or guardrails change.

Practical application. Pairs build a 30-case Halcyon test set, score their Module 2 prompt against it, then improve the score by changing one variable at a time.

06 The Applied 2026 Landscape

45 minutes

Source focus: IBM Think and Anthropic material on context engineering, agentic prompting and MCP, plus 2026 enterprise risk reporting.

The module that changes how participants describe their job. Prompt engineering has not disappeared; it has stopped being a sufficient description of production work, because the unit of design is now the system.

Context engineering is the bigger craft: the smallest high-signal token set that maximises the outcome. The context stack is taught with one detail people get wrong, since permission filters apply before retrieval, not after generation.

Agents are introduced as contracts rather than conversations, and autonomy is granted in rungs. The module closes with the operating model (a prompt library, versioning, ownership, a review rhythm) and the 2026 risks that make it necessary.

Practical application. Groups write a seven-layer context budget and a seven-clause operating contract for the Halcyon assistant, then stress-test each other's designs and report which layer or clause was missing.



THE APPROACH

How you will learn

HANDS ON, NOT A DEMO

Six labs, 147 minutes at the keyboard, roughly 35 per cent of the day. Every technique is applied in the module that teaches it.

ONE RUNNING CASE

Halcyon Logistics carries every exercise, so no time is lost inventing a business context and results are comparable across the room.

CHECKS THAT INFORM

Six ten-question checks, marked together in the room. They exist to expose disagreement, not to grade anyone.

MIXED ROOMS

Pairs and groups change through the day, so business and technical participants work on the same problem from different angles.

What you take away

- A scored baseline prompt and its recorded output, so improvement is measurable rather than asserted
- A 30-case test set you can re-run against any future model, index or parameter change
- A hardened system design: a context budget, an operating contract and the controls that contain damage
- A prompt library schema and the governance questions to take to your risk owner

Your first 30 days

- Week 1** Rewrite your most-used prompt using the five-part anatomy, and keep the output.
- Week 2** Build a test set and score the prompt. If you do only one week, do this one.
- Week 3** Threat-model it: direct injection, indirect injection, jailbreak, and the control for each.
- Week 4** Write the context budget and the operating contract for the system around it.

DELIVERY AND ENQUIRIES

Prompt Engineering 2026 runs as a live, instructor-led workshop, on site or remote, for business and technical teams. The case study can be replaced with your own tickets, documents and systems. For dates, fees and in-house delivery, contact Agilizing on the website or WhatsApp number in the footer of every page.