

AI Foundations, Prompt Engineering and Context Engineering

A practical, non-technical two-day workshop

FA-0568 · 2 days

Agentic & Generative AI · AI for Leaders & Business

Understand AI systems, practise prompt/context design and review retrieval, agents, evaluation and organisational oversight.

Introduction

This two-day workshop develops practical AI literacy without requiring programming or machine-learning mathematics. Participants examine how model behaviour, instructions, retrieved information, tools and system context interact.

Day 1 covers foundations, language models, limitations and selected prompt exercises. Day 2 explores context, retrieval, agents, connectivity, evaluation and organisational choices at a conceptual level. The goal is informed use and system review, not complete production-system design or guaranteed control of model behaviour.

Learning outcomes

The workshop teaches participants to:

- Explain common AI approaches and distinguish model behaviour from conventional rules and workflows.
- Describe language-model training/inference, tokens, context and information limits.
- Recognise variability, hallucinations, interpretability limits, bias and tool/security risks.
- Write, test and refine scoped prompts with clear constraints and output requirements.
- Select and organise relevant context while identifying stale, conflicting or untrusted material.

- Explain retrieval, tool-using agents and MCP client/server connectivity at a high level.
- Choose evaluation and oversight questions appropriate to the use case.
- Discuss build/buy, cost, responsibility and when AI may be unsuitable.

Prerequisites

- General computer literacy and familiarity with digital tools.
- No programming, data science, or machine learning background required.
- Willingness to think clearly about language, instructions, and decision-making.
- Curiosity about how AI systems behave in real-world environments.

Training outline

1. Day 1 – Foundations and practical prompting

1. AI concepts and boundaries

The problem with the word "AI"

- Why AI is not a single thing
- How marketing language can obscure capabilities and limitations.

AI vs traditional software

- Rules-based and model-based behaviour: determinism, probabilistic outputs and overlaps.
- Rules, logic, workflows, and automation

A practical view of AI approaches, not a single universal definition

- Prediction, generation and goal-oriented behaviour in selected systems.
- Pattern learning vs explicit instruction

Common categories of AI use

- Prediction and classification
- Generation (text, images, code)
- Decision support and optimization
- Planning and task execution

Limits of capability claims

- Distinguish observable task performance from claims of human-like understanding.
- Do not infer human intent from fluent output.
- Assess reasoning-like outputs against evidence and task results.

2. How model-based systems work

The basic AI system pipeline

- Data, model design, objectives, training and inference: a simplified lifecycle.

Learning from examples

- Learning patterns alongside rules, objectives and other system components.
- The role of probability in AI behavior

Training vs using a model

- Training, possible updates/fine-tuning and repeated inference: different lifecycle activities.

Why AI outputs change

- Randomness and probability
- Context sensitivity
- Environmental inputs and system state

Predictability and reproducibility in model-based systems

- Deterministic components, generation settings and statistical behaviour.

3. Ways to classify systems

Classification by learning approach

- Supervised learning
- Unsupervised learning
- Self-supervised learning
- Reinforcement learning

Classification by capability

- Narrow task-specific AI
- General-purpose models
- Multi-modal systems

Classification by autonomy

- Assistive systems
- Semi-autonomous systems
- Tool-using systems with configured autonomy and oversight.

Risk and impact as a contextual assessment, not a universal legal taxonomy

- Lower-consequence productivity examples.
 - Decision-support examples with relevant risks.
 - Consequential or autonomous tasks requiring stronger safeguards.
- Why classification matters for governance, cost, and trust

4. Language models

What an LLM actually is

- Text prediction at scale
- Tokens as text units: not necessarily whole words.

How LLMs “generate” responses

- Probability distributions over possible next tokens
- Why fluent language does not equal correctness

Training lifecycle

- Pretraining on large corpora
- Instruction tuning
- Preference and alignment tuning

Why LLMs sound confident even when wrong

- Fluency bias
- Lack of grounding

Practical limits of LLMs

- Context window constraints
- Limits of information learned during training versus current retrieved sources.
- Sensitivity to phrasing and order

5. Limitations and risks

Non-determinism

- Why the same input does not always produce the same output
- Reproducibility challenges

Black-box behavior

- Interpretability techniques and limits; generated explanations are not proof of internal causation.
- The danger of over-interpreting justifications

Hallucinations

- Fabricated or unsupported information: possible causes and verification.
- Conditions that increase hallucination risk

Emergent behavior

- Unexpected capabilities
- Apparent performance changes: check evaluation methods and evidence.

Over-reliance and automation bias

- Human trust failures

- Deskilling risks

Security risks

- Prompt injection
- Data leakage
- Tool misuse in agentic systems

6. Prompt design and refinement

What prompt engineering actually is

- Instruction design, not magic words

Anatomy of a good prompt

- Task definition
- Role specification
- Constraints and boundaries
- Output format control

Prompt clarity and ambiguity management

- Assumptions
- Definitions
- Scope limits

Prompt patterns to test for the selected model and task

- Decompose a complex task into manageable parts when useful.
- Structured output prompts
- Request relevant evidence, checked calculations or concise rationale rather than assume displayed reasoning proves correctness.

Prompt anti-patterns

- Overloaded prompts
- Conflicting instructions
- Implicit expectations

Prompt lifecycle management

- Versioning
- Testing
- Iterative refinement

2. Day 2 — Context, connected systems and review

7. Context engineering

Why prompts alone are not enough

- The difference between instruction and environment

What “context” means in AI systems

- System messages
- Retrieved knowledge
- Tool outputs
- Conversation history
- Hidden operational constraints

Context vs memory

- Short-term context
- Persistent information or memory mechanisms where the application provides them.

Designing effective context

- Relevance filtering
- Ordering and prioritization
- Noise reduction

Context failure modes

- Context overload
- Conflicting signals
- Stale or misleading information

Context engineering for non-technical roles

- Business rules as context
- Policies and guardrails
- Brand/tone and applicable policy context; prompt text alone does not enforce compliance.

8. Retrieval and grounding

- Why source checking and evaluation remain necessary

Retrieval-augmented generation (conceptually)

- External knowledge sources
- Use retrieved evidence as context; retrieval does not guarantee factual correctness.

- Embeddings and similarity (high-level intuition)

Designing traceable knowledge flows

- Source selection
- Update frequency
- Attribution and traceability

Risks of poor grounding

- Confident misinformation
- Outdated or biased sources

9. Agentic workflows

What agentic AI means in practice

- Goals, plans, actions, feedback
- Difference between chatbots and agents
- Single-agent vs multi-agent systems

Tool-using agents

- APIs, databases, browsers, workflows

Autonomy spectrum

- Suggestion-only systems
- Human-approved execution
- Execution within configured permissions and limits, with risk-appropriate oversight.

Risks unique to agentic AI

- Runaway behavior
- Compounding errors
- Actions based on unreliable outputs or an incorrectly specified objective.

Guardrails for agentic systems

- Permissions
- Limits
- Human-in-the-loop controls

10. MCP connectivity

- The integration problem in AI systems
- Why tools, data, and models need standardized interfaces

Conceptual overview of MCP

- Client/server access to context resources and tools, not automatic shared memory between agents.
- Tool access
- Separation of responsibilities

- When MCP-style approaches make sense

Security and governance implications

- Trust boundaries
- Auditability
- Access control

11. Evaluation and monitoring

- Why conventional software tests need complementary model/task evaluations.
- Defining “good enough” behavior

Evaluation dimensions

- Accuracy
- Usefulness
- Safety
- Consistency

- Human vs automated evaluation

Monitoring AI in real use

- Drift
- Degradation
- Misuse

- Incident response for AI failures

12. Responsibility and governance

- Ethics as a practical and organisational concern

Core ethical principles translated into practice

- Fairness
- Accountability
- Transparency
- Privacy

Organizational responsibility

- Who owns AI decisions
- Who approves deployment

- Bias and representational harm
- Legal and compliance awareness (high-level)
- Building trust without over-promising

13. Organisational design choices

- Build vs buy decisions
- Choosing the right level of AI sophistication
- Cost, risk, and value tradeoffs

Cross-functional collaboration

- Product
 - Legal
 - Security
 - Operations
- Change management and AI adoption

14. End-to-end review

- Framing the problem correctly
- Selecting the right AI approach
- Designing prompts, context, tools, and oversight together
- Embedding safety and evaluation from day one
- Knowing when not to use AI