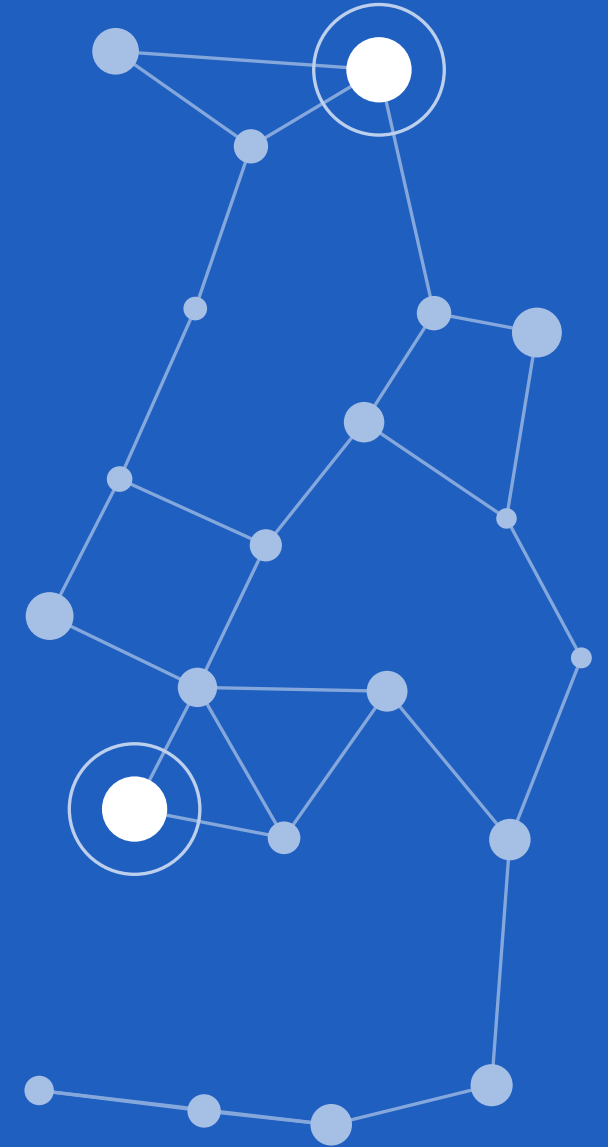


AI PROJECT ONBOARDING

Shared vocabulary turns AI ambition into sound project choices

Key concepts for understanding LLMs and AI agents



A successful AI project designs the full system, not only the model

1	Model capability	Select a model and define what it can see in each interaction.
2	Grounded knowledge	Retrieve trusted, current enterprise information rather than relying on model memory alone.
3	Controlled action	Give the model narrow, observable tools instead of unrestricted system access.
4	Measurement and control	Evaluate quality, enforce guardrails, and monitor cost and latency in production.

Decision requested: Use this vocabulary to frame scope, owners, evidence, and trade-offs before selecting a solution.

Source: OpenAI — Tokens and tokenization ; Lewis et al. — Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks ; OpenAI — Function calling guide ; NIST — AI Risk Management Framework

Tokens and the context window set what the model can process in one interaction



Tokens are model-readable pieces of text

Text is split into tokens before processing; tokenization varies by model and language.



The context window is the model's working set

Instructions, conversation history, retrieved knowledge, and output all compete for the same finite window.



Context is a product design constraint

Decide what to include, summarize, retrieve, or omit so the model sees the relevant evidence.

Training builds a model; inference applies it to each request



Training changes model parameters

The learning process adjusts parameters from data to create general capabilities.



Inference applies the trained model

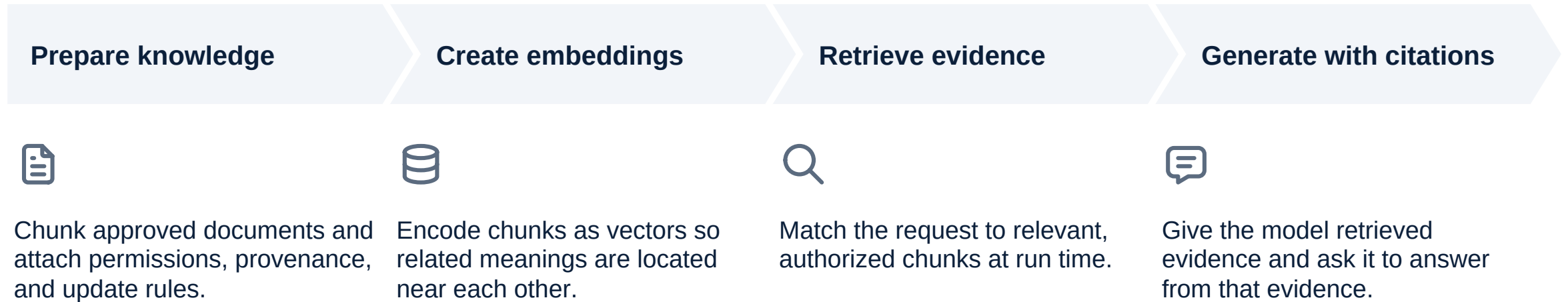
Each request combines an input and context with the existing model to produce an output.



The distinction changes project choices

Most enterprise work improves inference through prompts, retrieval, tools, and controls rather than training a model.

Embeddings make meaning searchable, and RAG uses that search to ground answers



Source: Lewis et al. — Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks

Prompting steers a model now, while fine-tuning changes behavior through training

	RECOMMENDED Prompting Instructions and examples at request time	Fine-tuning Additional training on representative examples
Best when	The task or policy changes often	Behavior needs a stable, repeatable pattern
Changes	Input context and instructions	Learned model behavior
Typical project work	Design prompts, templates, retrieval	Curate, label, split, and test a training set
Main governance need	Version prompts and test regressions	Manage data quality, privacy, and evaluation

Verdict: Start with prompting and retrieval; fine-tune only when evidence shows a persistent gap.

Source: [OpenAI — Prompt engineering](#); [OpenAI — Fine-tuning guide](#)

Hallucinations are plausible but unsupported outputs, so systems need multiple reduction layers



Ground each answer in approved evidence

Retrieve trusted sources and require citations where users need to verify an answer.



Constrain important outputs

Use structured schemas, validation, and policy checks before a response or action is accepted.



Clarify or escalate ambiguity

Ask for missing context and route high-consequence cases to a qualified person.

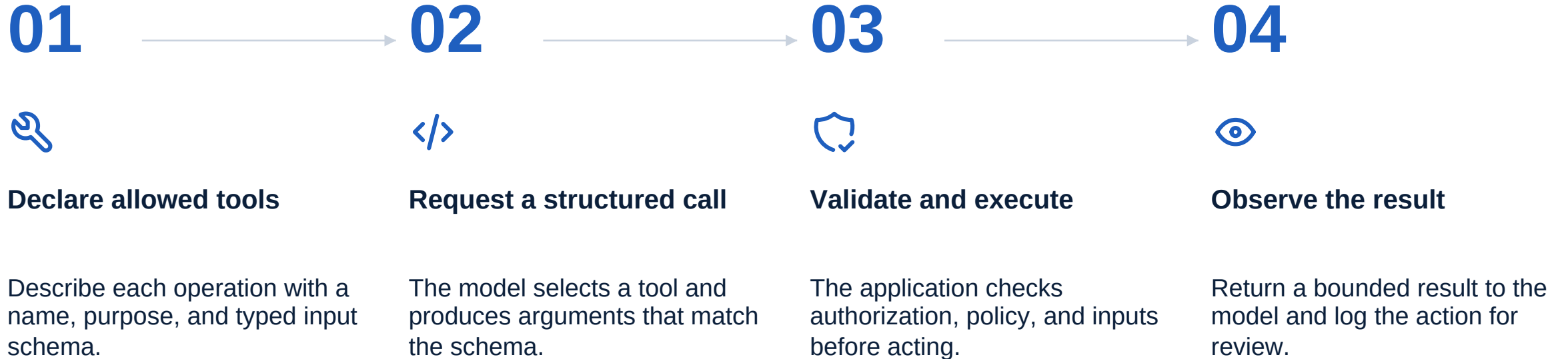


Govern the knowledge base

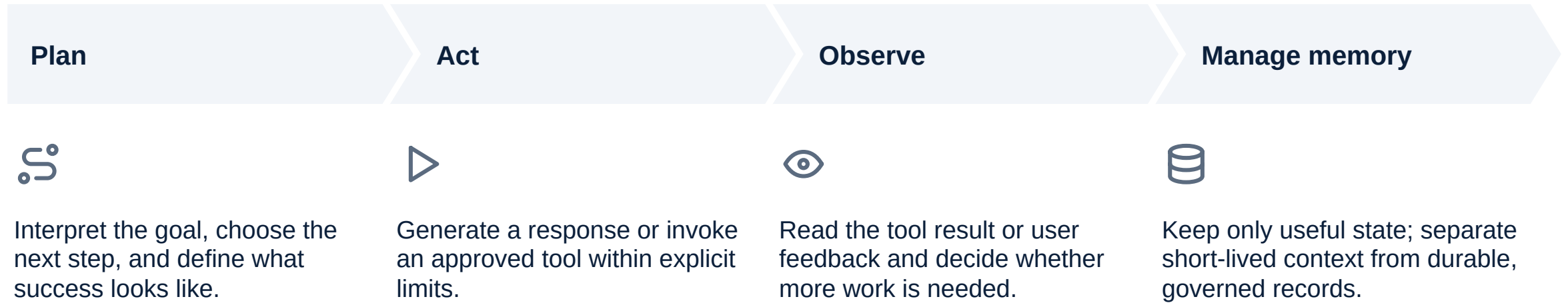
Set provenance, access, freshness, and review rules for the sources used at run time.

Source: Lewis et al. — Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks ; OpenAI — Function calling guide; NIST — AI Risk Management Framework

Tool use lets a model act, and function calling supplies the controlled contract



Agents repeat plan–act–observe loops, with memory treated as managed state



Source: Anthropic — Building effective agents; NIST — AI Risk Management Framework

Evaluation and guardrails make quality, safety, and operational control testable



Define success and failure

Build representative tasks, expected outcomes, and unacceptable failure modes into an evaluation set.



Test before release

Compare quality, safety, and tool-use behavior against agreed acceptance criteria.



Guard at run time

Apply access controls, content policies, schemas, approvals, and human escalation.

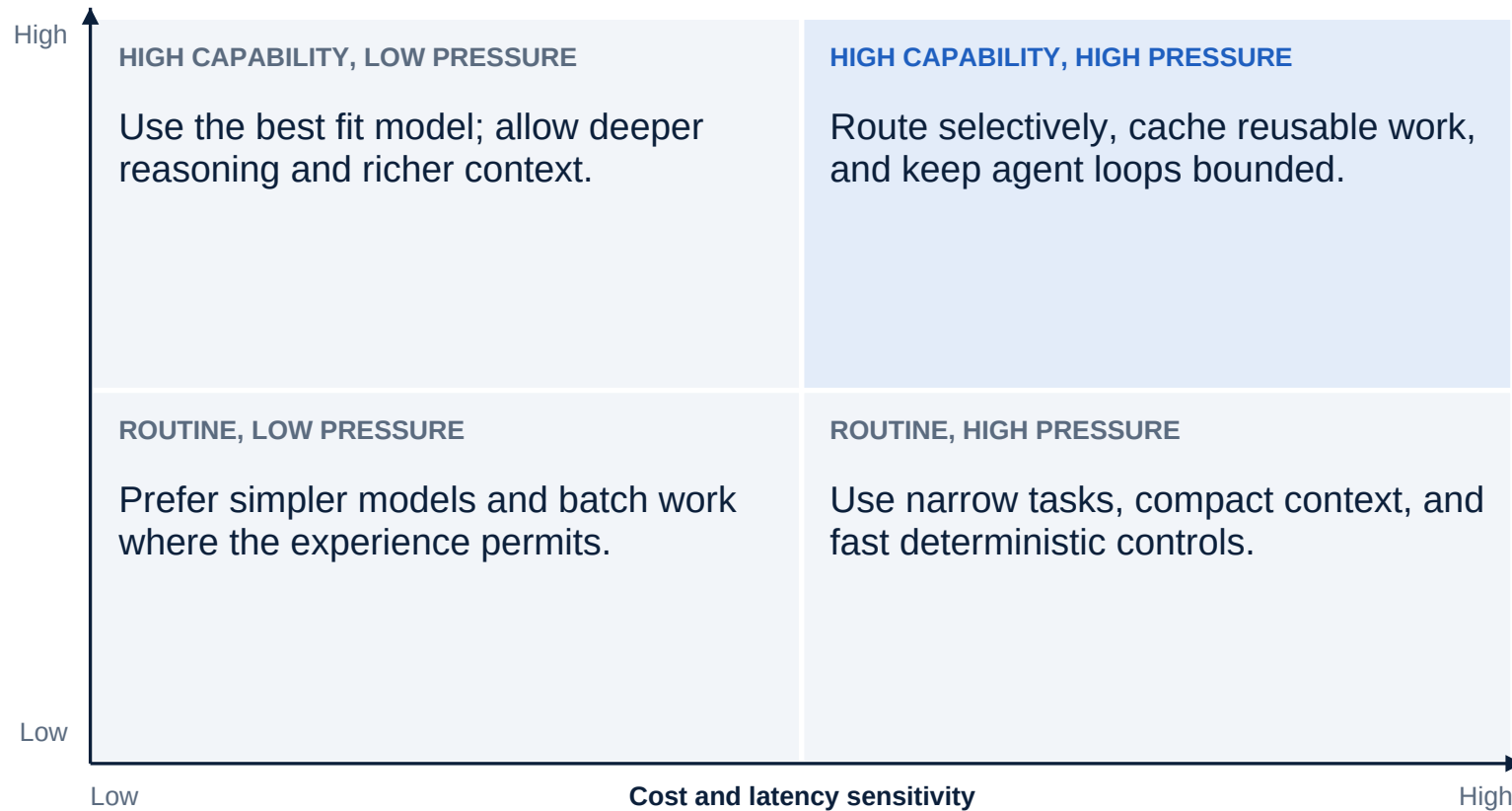


Monitor in production

Log outcomes, analyze failures, and refresh evaluations as the product and knowledge change.

Capability, cost, and latency form a design trade-off to optimize deliberately

Required capability and judgment



Set a service-level target first; then choose a model, context strategy, and control loop to meet it.

Source: OpenAI — Latency optimization; OpenAI — Tokens and tokenization

A short discovery sequence turns the vocabulary into an AI project charter

Next step	Owner	Due
Select a narrow user problem and define the decision or outcome it should improve	Business sponsor	Discovery start
Specify the context, trusted knowledge, tools, and permissions the system may use	Product and engineering	Solution design
Define evaluation cases, failure thresholds, guardrails, and escalation paths	Product and risk owner	Before pilot
Set quality, cost, and latency targets before choosing the production architecture	Technology lead	Before build

Source: OpenAI — Evaluation best practices; NIST — AI Risk Management Framework; OpenAI — Latency optimization