AI Development Skills Diagnostic Framework

Premise

In the AI era, product-level technical decisions impact products, customers, and organizations. This diagnostic visualizes the capabilities of engineers who can understand that impact and take ownership of those decisions.

Depth in a specific domain is not required, but cross-domain structural understanding and the ability to make judgment calls under uncertainty are assumed.

Purpose

Visualize individual engineers' skill profiles to enable matching with appropriate roles and opportunities, and to inform development planning.

This is not a ranking where "higher scores mean better engineers," but a matching diagnostic: "this profile fits this role or opportunity." Whether used for project assignment, team composition, or individual growth roadmaps is determined by the user of the framework.

How to Use

1. Read the Overall Structure to understand the two-layer model and assessment axes
2. Review the Scoring Guide for Low/Mid/High judgment criteria
3. Assess each sub-item using the assessment criteria, recording evidence for each rating
4. Plot results on a radar chart (3 foundation + 6 specialized axes) and interpret the profile shape
5. Cross-reference the Foundation–Specialized Dependencies table to flag ceiling risks

Overall Structure

Skills are separated into two layers: **Foundation Skills** and **Specialized Skills**.

Foundation Skills

Universal fundamentals that existed before AI—but redefined to be effective in the AI era.

Weak foundations lower the ceiling of specialized skills. Conversely, someone with high specialized skills but weak foundations can only perform under specific conditions.

Assessment axis: **Capability only** (single axis). These abilities are exercised daily as a working engineer; the question "do you have experience or not" doesn't apply, so no Experience axis is used.

Each sub-item is assessed on a 3-level scale: **Low / Mid / High**.

#	Category	Sub-items
1	Software Engineering Fundamentals	Design / Implementation / Cross-Layer Decision Making / Testing & Quality Assurance / Operations & Infrastructure / Security Fundamentals
2	Articulation & Communication	Structured Thinking / AI-Directed Articulation / Human Communication & Consensus Building
3	Efficiency Optimization	ROI Judgment / Waste Identification & Elimination / Continuous Improvement Cycles

Specialized Skills

Practical capabilities in AI development. Built on top of foundation skills.

Assessment axes: **Capability** and **Experience** (dual axis). In the AI domain, experienced practitioners are scarce, and a large population "knows the concepts but has no field experience." Separating capability and experience surfaces high-potential talent and improves matching and development decisions.

Each axis and sub-item is assessed on a 3-level scale: **Low / Mid / High**.

#	Category	Sub-items
1	Problem Framing & Requirements Definition	Problem Structuring & AI Applicability / Requirements Refinement / Constraint Identification & Consensus Building / Hypothesis Design

2	AI Domain-Specific Technical Skills	AI Characteristics & Model Selection / Context Engineering / Agent & Workflow Construction / AI-Specific System Architecture
3	Quality Assurance & Evaluation	AI Output Judgment / Eval Design / Product Quality vs. Technical Quality / Per-Case Quality Assurance
4	Operations & Maintenance	LLMOps / Observability / Data Quality Management / Maintainability Design
5	Process Design & Optimization	AI-Native Development Process Design / Feedback Loop Design & Operation
6	AI Security & Governance	AI Security Design / Data Boundaries & Privacy / Regulatory & Compliance / Operational Guardrail Design

Design Rationale

Two-layer structure — Separates ceiling causes: when specialized skills plateau, identifies whether the root cause is foundation or specialized. Foundation skills are also portable beyond AI assessment.

Dual axes for specialized only — AI domain has many "knows but hasn't done" engineers. Capability-only misses the PoC-vs-production distinction. Experience-only misses high-potential talent.

Radar chart over level system (Lv1–5) — Level systems produce single-line rankings. AI development profiles are uneven—strong in requirements but weak in operations, etc. Radar charts visualize these shapes for matching.

High = "runs without me" — High does not mean "can do it well personally" but "can codify tacit knowledge into guidelines, standards, and automated checks so that quality is maintained without depending on any individual."

Scoring Guide

Low / Mid / High Judgment Criteria

Level	General Standard	Evidence Required

Low	Lacks awareness or habit in this area	Observable gaps in daily practice or work output
Mid	Can execute effectively within own scope	Verifiable in specific projects: code, designs, decisions
High	Can build mechanisms that scale beyond the individual	Persistent artifacts: guidelines, automated checks, templates, operational frameworks

Score based on verifiable evidence—not self-assessment alone. The line between Mid and High is whether knowledge has been externalized into reusable, persistent artifacts.

Reading Capability × Experience (Specialized Skills)

Capability	Experience	Interpretation	Suitable Roles & Opportunities
High	High	Strong in both ability and track record	High-difficulty technical decisions, team technical lead
High	Low	Capable but lacking field experience	New exploration / PoC, stretch assignments with experienced support
Mid	High	Experienced but with room for capability growth	Stable operations, known-pattern work
Low	Low	Development target	Hands-on work under mentorship, learning opportunities

Engineers with High Capability / Low Experience are particularly common in the AI domain. Identifying this segment and designing opportunities for them to gain experience is key to elevating organizational AI development capability.

Foundation–Specialized Dependencies

Foundation skills function as prerequisites for specialized skills. Even with high specialized skills, they cannot be applied effectively if the corresponding foundation is weak.

- **Software Engineering Fundamentals** → Quality Assurance & Evaluation, Operations & Maintenance, AI Security & Governance
- **Articulation & Communication** → Problem Framing & Requirements Definition (consensus building), AI Domain-Specific Technical Skills (context engineering), Process Design & Optimization
- **Efficiency Optimization** → All areas (making technically sound but ROI-negative decisions)

When foundation skills fall below a certain threshold, high specialized skill scores should be treated as potentially locally optimal. Use this table to flag ceiling risks.

Foundation–Specialized Boundaries

Some items cover similar domains across both layers. To prevent double-counting, boundaries are made explicit.

Domain	Foundation Scope	Specialized Scope
Operations	General operational fundamentals (availability, monitoring, incident response)	LLM-specific operations (LLMOps, AI Observability)
Security	General application/infrastructure security fundamentals	AI-specific security (Prompt Injection, data boundaries, etc.)
AI-directed articulation	Foundational ability to communicate purpose, constraints, and output format to AI	Context design as information architecture (rule file structure, context closure, update operation mechanisms)
Data quality	—	Design (information architecture, structuring, selection) under Specialized 2-2; operations (freshness, permissions, drift, update SLA) under Specialized 4-3

Foundation Skills Assessment Criteria

1. Software Engineering Fundamentals

1-1. Design

Decomposing requirements into appropriate structures and designing maintainable, extensible software. A pre-AI fundamental—without it, you cannot make design judgments on AI output.

Capability

- Low: Can build working software but does not consider structural decomposition or separation of concerns.
- Mid: Can divide modules at appropriate granularity and design change-resistant structures. Can explain the reasoning behind design decisions.
- High: Can design with system-wide coherence in mind. Can codify design philosophy into guidelines and automated checks so that design quality is maintained without personal review involvement.

1-2. Implementation

Translating designs into working code, with attention to readability, testability, and performance.

Capability

- Low: Can write working code but lacks awareness of readability and testability.
- Mid: Writes readable code with attention to naming, structure, and error handling. Can write appropriate tests.
- High: Can deliver implementations that achieve a high standard across performance, security, and maintainability simultaneously. Can codify implementation standards into review criteria and guidelines, ensuring implementation quality scales beyond the individual.

1-3. Cross-Layer Decision Making

Backend, Frontend, and Infrastructure each have distinct responsibilities, constraints, and risks. This skill is about making final calls on where to solve what—requiring structural understanding across all layers, not just a primary domain.

Capability

- Low: Confined to a single domain (Backend/Frontend/Infrastructure). Does not understand design intent or constraints of other domains.
- Mid: Understands responsibilities and constraints of adjacent domains in addition to primary domain. Can participate in cross-layer tradeoff discussions.

- High: Can design responsibility boundaries across Backend, Frontend, and Infrastructure, and determine "which layer should solve this" based on tradeoffs. Can implement across layers as needed to resolve bottlenecks.

1-4. Testing & Quality Assurance

Verifying software correctness through test strategy design, test code implementation, and quality standard operation.

Capability

- Low: No testing habit, or only minimal tests.
- Mid: Understands test strategy and can design and implement tests at appropriate granularity and coverage.
- High: Can design overall test strategy. Has judgment criteria for what to test at which layer and to what extent, and can codify those criteria into CI and automated verification.

1-5. Operations & Infrastructure

Knowledge and practical ability to keep software running in production: deployment, monitoring, incident response, and infrastructure understanding. Covers general operational fundamentals; LLM-specific operations (LLMOps, AI Observability) are assessed under specialized skills.

Capability

- Low: Can run software in development environments but does not understand production operation considerations.
- Mid: Understands and practices deployment, monitoring, log design, and incident response fundamentals. Can read the intent behind infrastructure configurations.
- High: Can design infrastructure with explicit tradeoffs across availability, scalability, and cost. Can lead impact scoping and recovery during incidents. Can incorporate operational constraints at the design stage.

1-6. Security Fundamentals

Universal pre-AI application and infrastructure security fundamentals: authentication/authorization, secret management, vulnerability response, and threat modeling. Covers general security fundamentals; AI-specific security (Prompt Injection, data boundaries, etc.) is assessed under specialized skills.

Capability

- Low: Not aware of fundamentals in authentication/authorization, secret management, or vulnerabilities.
- Mid: Understands OWASP Top 10 level risks and can address them in implementation and review. Manages secrets appropriately.
- High: Can incorporate threat modeling perspectives into design. Can codify logging, audit, and permission design into reproducible standards through guidelines and automated checks.

---

2. Articulation & Communication

2-1. Structured Thinking

The prerequisite for all communication, whether directed at AI or humans: logically organizing and structuring one's thinking.

Capability

- Low: Speaks or writes without organizing thoughts. Jumps between topics, omits premises.
- Mid: Can decompose issues and organize them in a premise → claim → evidence structure. Can notice gaps and omissions.
- High: Can structure complex problems from multiple perspectives and adjust abstraction level to the audience's understanding. Can surface implicit assumptions.

2-2. AI-Directed Articulation

Accurately communicating intent to AI: purpose, constraints, expected output format, and corrective instructions—without ambiguity. Covers the foundational ability to articulate for AI; context design as information architecture (rule file structure, context closure, update operation mechanisms) is assessed under specialized skills.

Capability

- Low: Gives vague instructions to AI and cannot recognize when poor results stem from instruction quality.
- Mid: Can clearly communicate purpose, constraints, and expected output format to AI. When results don't match intent, can identify and improve instruction issues.

- High: Can write specification-grade instructions that accurately convey intent based on understanding of AI behavioral characteristics. Can create instruction templates at a quality level reusable by the team.

2-3. Human Communication & Consensus Building

Communicating technical content to people and gaining understanding and agreement. Includes explanation to both technical and non-technical audiences, and codifying tacit knowledge into documents and guidelines so that decisions proceed without personal involvement.

Capability

- Low: Struggles to explain technical content in own words. Relies on jargon.
- Mid: Can explain technical content calibrated to the audience's knowledge level. Can organize tradeoffs and present options.
- High: Can codify tacit knowledge (design philosophy, judgment criteria, review perspectives, etc.) into documents, guidelines, and standards. Can create a state where the team operates autonomously without requiring personal involvement in reviews or decisions.

---

3. Efficiency Optimization

3-1. ROI Judgment

Achieving maximum results with limited resources (time, cost, people), considering business-level ROI beyond purely technical concerns.

Capability

- Low: Prioritizes completing the task at hand. Does not consider overall ROI.
- Mid: Considers the balance between cost and outcomes when making technical selections and implementation decisions.
- High: Can determine technical investment priorities by working backward from business impact. Can lead resource allocation optimization, including "don't do it" decisions.

3-2. Waste Identification & Elimination

Recognizing and improving inefficient processes, designs, and work—including spotting excessive automation and unnecessary effort.

Capability

- Low: Follows existing approaches as-is. Does not notice inefficiency.
- Mid: Recognizes and improves inefficiency within own work scope. Can make decisions to skip unnecessary work.
- High: Can structurally analyze inefficiency across the team or project and drive improvements. Can see through the negative effects of excessive automation or blind adoption of practices.

3-3. Continuous Improvement Cycles

Sustaining efficiency improvements beyond one-off efforts, maintaining reflection, measurement, and improvement cycles autonomously.

Capability

- Low: Makes improvements as one-off efforts. Does not measure effectiveness; improvements don't stick.
- Mid: Measures improvement effectiveness and connects it to next actions. Has a reflection habit.
- High: Can build improvement cycles as systematic mechanisms. Can foster a team improvement culture that runs autonomously.

Specialized Skills Assessment Criteria

1. Problem Framing & Requirements Definition

1-1. Problem Structuring & AI Applicability

Transforming "we want to use AI" into "a question worth solving with AI." Decomposing business problems and sorting which parts AI should solve and which it should not.

Capability

- Low: Throws business problems directly at AI. Frames questions as "can AI do something about this?"
- Mid: Can decompose problems and sort which parts AI should solve versus which should use rule-based approaches or human judgment.

- High: Beyond problem structuring, can identify cases where AI application would be counterproductive (accuracy requirements, cost, operational burden) and make non-adoption decisions.

Experience

- Low: No track record of making AI applicability decisions independently.
- Mid: Has been involved in AI adoption/non-adoption decisions in a specific project.
- High: Has led such decisions across multiple projects and verified the outcomes.

1-2. Requirements Refinement

Bridging the gap between Product Manager resolution and engineer resolution, translating product requirements into technically feasible engineering specifications.

Capability

- Low: Attempts to implement Product Manager requirements as-is. Begins work with technical ambiguity unresolved.
- Mid: Can surface technical constraints and unknowns against Product Manager requirements and concretize them as specifications.
- High: Can read the intent behind Product Manager requirements and restructure them into a form that is both technically feasible and high in business value. Can elicit requirements the Product Manager didn't state but actually needs.

Experience

- Low: No experience leading requirements refinement independently.
- Mid: Has participated in a requirements definition phase and contributed specification from a technical perspective.
- High: Has led requirements definition multiple times including AI-specific uncertainty (accuracy variance, model dependency, etc.).

1-3. Constraint Identification & Consensus Building

Surfacing technical, cost, operational, and stakeholder constraints, organizing tradeoffs, and finding realistic landing points amid complex interests.

Capability

- Low: Recognizes only technical constraints. Overlooks cost, operational, and stakeholder constraints.

- Mid: Can surface technical, cost, and operational constraints, organize tradeoffs, and explain them to stakeholders.
- High: Can navigate and reach consensus on complex constraint conditions including per-customer requirement differences and inter-organizational conflicts of interest.

Experience

- Low: No experience leading constraint negotiations.
- Mid: Has experience with constraint negotiation within a team or project.
- High: Has led constraint negotiations spanning multiple stakeholders and customers.

1-4. Hypothesis Design

The prerequisite for eval design: defining what "success" looks like and determining what should be measured.

Capability

- Low: Begins implementation with vague success criteria. Recognition is limited to "accuracy should be high."
- Mid: Can define quantitative success criteria. Can determine what to measure and by which metrics before implementation.
- High: Can design staged hypotheses: "first verify this metric at this threshold, then validate from this perspective"—structuring steps including go/no-go criteria.

Experience

- Low: No experience designing hypotheses independently.
- Mid: Has been involved in success criteria design for a specific project.
- High: Has led the full cycle from hypothesis design through verification to pivoting based on results, multiple times.

---

2. AI Domain-Specific Technical Skills

2-1. AI Characteristics & Model Selection

Understanding each model's capabilities, constraints, and cost characteristics to select optimal configurations for product requirements. Requires holistic judgment across quality, cost, latency, safety, and availability.

Capability

- Low: Uses the default model as-is. Not aware of differences between models.
- Mid: Can select different models based on task characteristics. Understands speed/quality/cost tradeoffs.
- High: Can make production configuration optimization decisions balancing quality, cost, and latency (including selection among strategies such as routing, cascading, distillation, and model switching). Can evaluate the impact of new model releases on existing configurations and make migration decisions.

Experience

- Low: No experience making model selection decisions independently.
- Mid: Has practiced model differentiation within a project.
- High: Has led model selection decisions multiple times including production model switching and migration.

2-2. Context Engineering

Designing the information architecture of what to provide AI, when, in what order, and how. Goes beyond prompt design to include rule files, context closure, and information selection and structuring.

Context Closure: a state where all product-related information can be searched, referenced, and updated entirely within the LLM's workflow.

Context update and maintenance operations are assessed under 4-3 (Data Quality Management).

Capability

- Low: Uses AI through one-off prompts. Has no concept of context design.
- Mid: Can structure prompts, design rule files, and design the delivery of required information in appropriate order. Understands and can implement context closure.
- High: Can design context as information architecture. Can design the overall structure of what, when, and in what order to deliver to the model. Understands that unnecessary context becomes noise and can make subtraction decisions.

Experience

- Low: No experience with intentional context design.
- Mid: Has designed rule files or context closure for a project.

- High: Has designed context strategy across multiple projects.

2-3. Agent & Workflow Construction

Designing and implementing workflows and agents that coordinate multiple models and tools—building systems that achieve objectives through multi-step configurations rather than single requests.

Capability

- Low: Only uses single-model, single-request interactions.
- Mid: Can design and implement multi-step workflows. Can configure tool integrations and inter-model handoffs.
- High: Can build robust workflows including multi-agent configuration design, inter-agent coordination and task division, and fallback design for failures.

Experience

- Low: No experience building agents or workflows.
- Mid: Has built a workflow for a specific project.
- High: Has designed and operated multi-agent configurations running in production.

2-4. AI-Specific System Architecture

LLM-incorporating systems require design decisions that differ from traditional software architecture—around data pipelines, RAG configurations, and API design. This skill covers end-to-end system structure design with AI components.

Capability

- Low: Does not grasp design considerations when incorporating AI components into applications.
- Mid: Understands and can apply AI-specific architecture patterns such as RAG configurations, data pipelines, and API design.
- High: Can position AI components within overall system architecture and select configurations based on holistic judgment across scalability, latency, cost, and maintainability.

Experience

- Low: No experience involved in AI-specific architecture design.

- Mid: Has designed and implemented RAG or data pipeline solutions for a specific project.
- High: Has led architecture design for production-scale AI systems multiple times.

---

3. Quality Assurance & Evaluation

3-1. AI Output Judgment

Judging the correctness and validity of AI output: hallucination detection, edge case recognition, and discerning trustworthy from untrustworthy output domains.

Capability

- Low: Adopts AI output as-is. Has no judgment criteria for correctness.
- Mid: Recognizes the possibility of hallucination and can verify output validity. Can find obvious edge cases.
- High: Can detect subtle errors and seemingly-correct-but-inappropriate output by combining domain knowledge and technical understanding. Understands the boundary between domains where AI output is trustworthy and where it is not.

Experience

- Low: No experience systematically verifying AI output.
- Mid: Has practiced AI output review and verification within a project.
- High: Has deployed output judgment criteria to a team and maintained judgment quality at an organizational level.

3-2. Eval Design

Designing evaluation systems—what to measure, how to measure it, and how to automate the process. Covers metric definition, test case design, and evaluation pipeline construction.

Capability

- Low: Has no concept of designing evaluation metrics or methods. Judges by manually reviewing output and deciding it "looks good."
- Mid: Can define quantitative evaluation metrics and design test cases. Can build automated evaluation pipelines.
- High: Can design multi-dimensional evaluation frameworks integrating accuracy, response speed, cost, and user experience, and continuously improve them.

Experience

- Low: No experience with eval design.
- Mid: Has designed and executed evals for a specific project.
- High: Has built and operated continuous eval pipelines in a production environment.

3-3. Product Quality vs. Technical Quality

Distinguishing between user-experience quality and technical quality, ensuring both, and making tradeoff decisions when they conflict.

Capability

- Low: Does not distinguish between product quality and technical quality. Focuses on only one or the other.
- Mid: Can distinguish and discuss user-experience quality (response consistency, speed, usability) and technical quality (maintainability, performance efficiency, reliability) separately.
- High: Can make priority decisions when the two conflict. Can articulate tradeoffs between short-term product quality and long-term technical quality and make decisions.

Experience

- Low: No experience with quality management considering both dimensions.
- Mid: Has made quality decisions considering both dimensions within a project.
- High: Has driven the dual pursuit of product quality and technical quality at an organizational level.

3-4. Per-Case Quality Assurance

Ensuring quality not just in aggregate but per individual case and customer. Preventing "average looks good, but specific conditions cause breakdown."

Capability

- Low: Only considers overall average accuracy. Does not recognize the need for per-case quality assurance.
- Mid: Understands that required quality levels differ per customer and use case, and can design per-case evaluation.
- High: Can design the balance between global optimization and per-case optimization —including quality SLO definition, exception handling, and configurations that meet

specific customer requirements while maintaining overall system quality.

Experience

- Low: No experience with per-case quality assurance awareness.
- Mid: Has customized quality criteria for specific customers.
- High: Has designed and operated quality assurance systems spanning multiple customers.

---

4. Operations & Maintenance

4-1. LLMOps

Building deployment, operation, and update mechanisms for AI systems: model version management, eval integration into CI/CD, and continuous deployment strategy.

Capability

- Low: Does not understand model deployment or operation mechanisms.
- Mid: Can design model version management, deployment strategy, and rollback. Can integrate eval into CI/CD.
- High: Can design production model update strategies (canary releases, A/B testing, etc.) and build mechanisms for continuous deployment while maintaining quality.

Experience

- Low: No practical LLMOps experience.
- Mid: Has been responsible for model deployment and operations in a specific project.
- High: Has built and operated LLMOps pipelines in production, running model update cycles.

4-2. Observability

Designing and building AI-specific monitoring systems: output quality monitoring, anomaly detection, usage pattern analysis, and quality degradation early warning.

Capability

- Low: Not aware that AI-specific monitoring is needed.

- Mid: Can design and implement output quality monitoring and anomaly detection mechanisms. Can define what should be monitored.
- High: Can design comprehensive monitoring systems from usage pattern analysis through quality degradation early warning. Can connect monitoring results to feedback loops.

Experience

- Low: No experience building AI-specific monitoring.
- Mid: Has designed and operated monitoring for a specific project.
- High: Has built and continuously operated monitoring systems for production-scale AI systems.

4-3. Data Quality Management

Continuously maintaining and improving input data and context quality. Input quality determines AI output quality. Includes data drift response, context freshness and permission management, and update operations. Context design (information architecture, structuring, selection) is assessed under 2-2.

Capability

- Low: Weak awareness that input data quality directly determines AI output quality.
- Mid: Can define data quality validation rules and build mechanisms to detect quality degradation. Can manage context freshness.
- High: Can build comprehensive data quality management systems including data drift response, input data quality improvement cycles, context update SLAs, permission management, and breaking change detection.

Experience

- Low: No practical data quality management experience.
- Mid: Has performed data quality verification and improvement for a specific project.
- High: Has built and operated data quality management mechanisms in production, achieving continuous quality maintenance.

4-4. Maintainability Design

Designing AI systems for long-term maintainability: model generation transitions, API change resilience, and dependency management. Unlike traditional technology selection, stability through established defaults is not available—continuous adaptation is required.

Capability

- Low: Prioritizes building something that works; low awareness of maintainability.
- Mid: Can design with consideration for resilience to model changes and API changes. Can manage dependencies appropriately.
- High: Can design with long-term system evolution in mind. Beyond model generation transitions and phased architecture migration, can embed abstractions that absorb external dependency changes and change detection mechanisms into the design.

Experience

- Low: No experience with AI-specific maintainability-aware design.
- Mid: Has designed responses to model updates or API changes for a specific project.
- High: Has formulated and executed maintenance strategies for long-running AI systems.

---

5. Process Design & Optimization

5-1. AI-Native Development Process Design

Redesigning existing team structures and role assignments with AI as a premise, encompassing influence on both Discovery and Delivery phases.

Capability

- Low: Simply layers AI tools on top of existing development processes. No concept of rethinking the process itself.
- Mid: Can redesign role assignments and workflows with AI utilization as a premise. Recognizes inefficiencies in traditional team structures and can propose improvements.
- High: Can redesign processes across both Discovery and Delivery phases with AI as a premise. Can develop phased transition plans while understanding organizational status quo bias.

Experience

- Low: No experience involved in development process design.
- Mid: Has led process improvements within a team or project.

- High: Has led and established organization-level development process transformation.

5-2. Feedback Loop Design & Operation

Running verification cycles when AI acts autonomously and reflecting results back into context, across both product and process dimensions. Encompasses rapid hypothesis validation and channeling lessons from failures back into rules and processes.

Capability

- Low: No mechanism to reflect execution results into next actions. Treats implementation as complete once shipped.
- Mid: Can design cycles across both product and process dimensions that verify execution results when AI acts autonomously and reflect them into context.
- High: Can drive feedback loop automation and progressively reduce human involvement. Can build and operate mechanisms that channel both hypothesis validation results and process improvement results back into rules and context.

Experience

- Low: No experience with feedback loop design.
- Mid: Has built a feedback cycle for a specific project.
- High: Has built feedback loops across both product and process dimensions and progressively reduced human involvement.

---

6. AI Security & Governance

6-1. AI Security Design

AI introduces attack surfaces that differ from traditional application security: Prompt Injection, data leakage, and tool execution permission management. This skill covers understanding those surfaces and designing countermeasures.

Capability

- Low: Not aware of AI-specific security risks (Prompt Injection, data leakage, etc.). Can only think within the extension of traditional application security.
- Mid: Understands AI-specific attack surfaces and can design basic countermeasures including input/output sanitization, least privilege, and sandbox design.

- High: Can design security encompassing agent tool execution permission design, permission propagation control in multi-step workflows, and attack resilience verification through red teaming.

Experience

- Low: No experience with AI-specific security design.
- Mid: Has implemented Prompt Injection and data leakage countermeasures for a specific project.
- High: Has comprehensive security design and operation track record including production security incident response and red teaming.

6-2. Data Boundaries & Privacy

Establishing judgment criteria for what may be passed to models and what may be stored or reused—covering PII, confidential information, and knowledge source trustworthiness, freshness, and permission management.

Capability

- Low: Has no judgment criteria for what may be passed to models or what may be stored/reused.
- Mid: Understands PII, confidential information, and copyright handling; can define data boundaries. Can judge what may be included in training data or search targets.
- High: Can design comprehensive data governance including multi-tenant data isolation, per-customer data boundary design, and knowledge source trustworthiness, freshness, and permission management.

Experience

- Low: No experience designing AI systems with data boundary or privacy awareness.
- Mid: Has defined data boundaries or implemented PII countermeasures for a specific project.
- High: Has designed and operated customer data isolation and privacy requirement responses.

6-3. Regulatory & Compliance

The essence is not regulatory knowledge itself, but translating AI-related regulations into audit-ready technical design. Covers embedding compliance requirements into system architecture.

Capability

- Low: Not aware of AI-related regulations, or believes they are irrelevant to own work.
- Mid: Understands risk classification and requirements; can enumerate necessary logging and accountability items. Can judge which risk category own project falls into.
- High: Can lead designs that embed audit trails (logs, data flows, model change history, evaluation results) into systems. Can realize explainability assurance and risk assessment process construction from the technical side.

Experience

- Low: No experience involved in AI-related regulatory compliance.
- Mid: Has confirmed compliance requirements or performed documentation for a specific project.
- High: Has led audit response or regulatory compliance design for regulated AI systems.

6-4. Operational Guardrail Design

Designing safety mechanisms for AI autonomous execution: risk-level-based judgment criteria for human involvement and incident response flows.

Capability

- Low: Not aware of the need for guardrails on AI autonomous execution.
- Mid: Can design basic guardrails including human approval flows for high-risk operations, output filtering, and automatic shutdown on anomalies.
- High: Can design tiered guardrails based on risk level. Can define judgment criteria for what to automate and where to involve humans, and design the full scope including incident response flows.

Experience

- Low: No practical guardrail design experience.
- Mid: Has implemented approval flows or filtering for a specific project.
- High: Has designed and operated risk-level-based guardrails in production, including incident response.

Operational Guidelines

Visualization

Visualize using radar charts with axes corresponding to the 9 major categories (3 foundation + 6 specialized). Sub-item scores serve as supporting evidence for development feedback. Specialized skills can use separate Capability and Experience radars, or overlay both on a single chart.

Phased Introduction

There is no need to fully operationalize all items from the start. Begin by using the framework, identify items where assessment is inconsistent or misaligned with practice, and improve iteratively.

Maintenance

The AI development domain evolves rapidly. Specialized skill sub-items and behavioral indicators are designed to be reviewed on a 6-month to 1-year cycle as tools and practices evolve. Foundation skills change more slowly, so review frequency is lower.

2026 Context

This diagnostic reflects industry trends as of February 2026:

- Context engineering has conceptually subsumed prompt engineering and evolved into information architecture design.
- LLMOps, Observability, and eval are positioned at the center of operations.
- Regulations such as the EU AI Act are in phased application, and the importance of security and governance is growing.
- As AI has lowered implementation costs, the engineering bottleneck has shifted from "can you build it" to "can you make the right decisions."