AI implementation partner checklist

Technology, Software & Digital Infrastructure

Prepared by the RankGlobal Research Team · Reviewed 27 October 2026

Hire AI implementation partners who can fit an evaluated feature into your product's identity, data and release architecture. Require evidence of the proposed engineers' production contribution, not only prompt experiments. Compare retrieval or model engineering, integration, observability, reusable components, customer-support enablement and ongoing model-change costs across proposals.

WHO THIS BUYER GUIDE IS FOR

SaaS product leaders, enterprise software teams, managed-service providers and cloud or infrastructure operators.

Fill the fields and checkboxes in a compatible PDF reader, then save a separate copy for each candidate. You can also print this guide. No information is submitted.

Project / AI workflow

Candidate AI partner

Decision owner

Review date

01 AI service hiring and delivery checks

- [ ] Define the AI feature, user outcome and permitted tool actions.
- [ ] Verify named engineers' production AI integration and operating experience.
- [ ] Audit tenant permissions, retrieval sources and application API contracts.
- [ ] Test groundedness, action failures, isolation, latency and task-level cost.
- [ ] Budget models, search, observability, evaluation and support workloads.
- [ ] Record third-party licences, accelerator boundaries and bespoke source-code ownership.
- [ ] Train engineers and support staff on traces, incidents and model changes.
- [ ] Accept reproducible deployment, regression datasets, documentation and an exit plan.

Open the current Technology, Software & Digital Infrastructure provider directory

Read the full guides: AI use cases & scope · Costs & ROI · Choosing a partner · Build your brief

AI module cost context & offline quote notes

Illustrative upfront engineering budgets: the topic-specific effort assumptions below are multiplied by Upwork's published $35–$60/hour marketplace band. These are planning examples, not surveyed module prices, provider quotations or enterprise fees. Tools, hardware, tax and ongoing support are extra; supplier minimums may exceed these examples.

Tenant-aware knowledge-feature pilot · $2,240–$7,200

64–120 assumed engineering hours. Use one prepared knowledge collection and test tenant to demonstrate cited answers, abstention and role-isolation checks.

Reviewed support-triage module · $5,600–$17,280

160–288 assumed engineering hours. Connect one support route to permitted account context, classification, an exception queue and confirmed case updates.

Incident-investigation assistance release · $13,440–$45,600

384–760 assumed engineering hours. Deploy one approved telemetry/context workflow with evidence-linked hypotheses, engineer review, failure diagnostics and rollback tests.

Source: Upwork: AI engineer cost guide

Optional offline quotation notes: record actual supplier figures in one currency and identify included scope, usage limits and exclusions.

Discovery & readiness · one-time	
Build & integration · one-time	
Evaluation & deployment · one-time	
Training & handover · one-time	
Third-party tools, models & APIs · monthly	
Infrastructure & monitoring · monthly	
Maintenance & support · monthly	

Usage assumptions, exclusions and questions to resolve

Add the one-time rows for initial investment. Annualize monthly rows for operating cost; include variable usage and human review. Avoid counting the same managed-service charge both as a platform fee and a direct API expense.

Compare AI project scope

Qualitative scope inclusion, not measured cost or complexity. Core = explicitly scope the work; Conditional = confirm missing inputs; Defer = outside this narrowly bounded scenario.

Workstream	One product-support pilot	Connected product assistance	Production AI application service
Feature contract	Core	Core	Core
Live integration · Identity and authorization services	Defer	Core	Core
Live integration · Product data and search layer	Conditional	Conditional	Core
Tenant and release regression tests	Core	Core	Core
Live support · Production AI application service	Conditional	Core	Core

One product-support pilot

Evaluate a defined support or incident-assistance task using approved product knowledge and historical cases.

Boundary: Use approved samples covering feature contract. Live identity and authorization services access is excluded unless the acceptance test requires it; the named pilot reviewer retains operating responsibility.

Connected product assistance

Connect current knowledge and one support or product action with user permissions and traceable outcomes.

Boundary: Include the agreed identity and authorization services connection. Confirm whether product data and search layer access is needed for this release; additional sites, departments or channels need a separate scope decision.

Production AI application service

Maintain agreed tenants, releases or product surfaces with evaluation, telemetry and on-call handover.

Boundary: List the assets, volumes and users covered by production AI application service. Agree monitoring, handover and ongoing support for feature contract before expanding coverage.

02 How do we add an AI product feature without breaking tenant permissions, release controls or unit economics?

Treat an AI feature as a product integration with versioned behavior: carry tenant identity into retrieval and tool calls, enforce permissions server-side, and retain traces for review. Define an acceptance set covering grounded answers, denied access, failed actions and latency before release. Ask the partner to deploy behind staging controls with rollback. Model the complete request cost, including context, retries, agent steps, search and evaluation traffic; agree ownership of indexes, configurations and model-change regression tests.

Feature contract

Ask the AI team to inspect: Representative user tasks, unacceptable outputs and the application's authorization boundary.

Estimation impact: A read-only answer feature differs materially from an agent executing customer actions.

Data and tenancy

Ask the AI team to inspect: Actual source formats, tenant separation rules and deletion requirements.

Estimation impact: Permission-aware retrieval and data lifecycle work can dominate a seemingly small assistant.

Operating profile

Ask the AI team to inspect: Concurrency, response-time targets and complete tool-call traces for representative tasks.

Estimation impact: Latency and consumption need architecture-level estimation, not model-price multiplication alone.

03 Systems, tools and AI integration boundaries

Identity and authorization services

AI service requirement: Carry the user and tenant scope into retrieval and tool calls.

Scope check: Test denied access, session changes and server-side enforcement.

Product data and search layer

AI service requirement: Provide permissioned context and record source changes.

Scope check: Confirm freshness, deletion propagation and index ownership.

Application APIs and release systems

AI service requirement: Execute bounded actions and deploy versioned AI behavior.

Scope check: Test transaction recovery, feature flags, tracing and rollback in staging.

04 External software and operating costs

Model inference

Charging unit or cost driver: Tokens, requests or provisioned capacity

Ask before approving the quote: Estimate full context, retries, agent steps and concurrency rather than one ideal request.

Retrieval infrastructure

Charging unit or cost driver: Index storage, embeddings, search and reranking

Ask before approving the quote: Include document updates, permission filters and deletion processing.

Evaluation and observability

Charging unit or cost driver: Test executions, tracing volume and retention

Ask before approving the quote: Separate pre-release regression traffic and production incident investigation.

Managed service and support plans

Charging unit or cost driver: Platform entitlement and agreed response coverage

Ask before approving the quote: Record mandatory vendor tiers and ownership of production accounts and billing.

05 Reusable components, licensing and ownership

Agent or RAG framework

Check licence, maintenance status and exportability; framework reuse does not supply application permissions or evaluation evidence.

Partner-owned orchestration accelerator

Document what remains supplier IP, what source is delivered and how your team can replace or operate the component.

06 Staff training, documentation and support

Training and adoption

- Train engineers to trace retrieval, model and tool failures independently.
- Have support staff practice answering a customer incident without exposing another tenant's information.

Documents and technical assets

- Architecture decisions, dependency register and permission-flow diagrams.
- Evaluation dataset, scoring rationale and release regression instructions.
- Deployment configuration, model-change approval, incident response and supplier-exit procedures.

Maintenance and support

- Assign ownership for model-provider changes, tool schema drift and source-access failures.
- Agree maintenance scope for evaluations and migrations separately from a defect warranty.

07 From business requirement to working AI service

AI implementation partners for software companies: ship a feature your team can maintain

Software companies buy AI engineering to deliver a specific product or internal capability, not simply to expose a model API. The implementation must fit tenancy, permissions, latency, observability and product-support expectations. A successful prototype can still fail when customers use different data, invoke tools concurrently or require an explanation of an incorrect result.

Discovery should separate retrieval, generation, prediction and tool execution. Ask the partner to compare existing product features or managed services with custom engineering, then estimate the evaluation and operating layers. Source-code ownership, infrastructure accounts, regression datasets and release documentation are part of the deliverable, not optional extras after a demo.

08 How an AI implementation partner should scope and deliver the work

Treat the AI feature as supported product software with explicit permissions, evaluation and ordinary engineering ownership.

1. Define the product behavior and boundary

Map the user task, affected tenant data and successful outcome. Decide where the AI answers, recommends or calls a tool, and define failure behavior.

Accountable team: Product owner and AI architect

Data / system: User journey, authorization model and existing service contracts

2. Audit data and service dependencies

Inspect retrieval sources, identity propagation and tool APIs. Compare managed capabilities with custom components and document accounts, licences and deployment constraints.

Accountable team: AI engineer and backend architect

Data / system: Source repositories, search services, application APIs and identity provider

3. Evaluate before feature rollout

Create representative and adversarial tasks, including unanswerable questions and failed actions. Test grounded output, tenant isolation, response time and total task consumption.

Accountable team: Evaluation engineer and product reviewers

Data / system: Versioned test set, staging environment and traces

4. Release into ordinary engineering operations

Add monitoring, rollback and model-change regression tests. Train support and developers to investigate incidents and maintain the feature after the initial supplier engagement.

Accountable team: MLOps lead and product engineering owner

Data / system: CI/CD, telemetry, feature flags and incident runbooks

09 AI use cases and evaluation requirements

1. Customer knowledge feature

Ground product answers in the tenant's authorized documents and current product version.

Inputs: Versioned documentation, tenant metadata, permissions and realistic support questions.

AI partner deliverables: A source-linked feature with permission and stale-document tests.

Measure the result: Grounded answer rate, retrieval failures and cross-tenant access attempts.

Failure to test: An answer can be fluent and still reveal a document the user is not permitted to access.

2. Support triage

Classify incoming cases, retrieve relevant product context and suggest routing or a draft for an agent.

Inputs: Historical tickets, routing labels, product versions and reviewed resolutions.

AI partner deliverables: A support workflow with reason codes and override capture.

Measure the result: Wrong queue assignments, reassignment effort and agent acceptance.

Failure to test: Ticket closure or customer messages should not happen merely because a classifier is confident.

3. Incident investigation assistance

Connect a service symptom to relevant deployment events, telemetry and approved runbooks for an engineer to inspect.

Inputs: Correlated traces, metrics, logs, release events and service ownership.

AI partner deliverables: A read-only incident assistant with source links and safe investigation boundaries.

Measure the result: Time to useful evidence and false diagnostic suggestions.

Failure to test: A generated causal story must not be treated as proof or used to issue uncontrolled remediation commands.

4. Bounded product actions

Help a user perform an explicit task such as preparing a configuration change, with confirmation before commitment.

Inputs: API contracts, authorization checks, tool schemas and test accounts.

AI partner deliverables: An action flow with preview, approval, audit trail and recovery.

Measure the result: Completed tasks, incorrect actions and recovery from partial failures.

Failure to test: Retrying a timed-out write must not duplicate a customer operation.

10 Specialist team and relevant deployment evidence

AI product architect and RAG or agent engineer

Chooses bounded retrieval, generation and tool components that fit the product's user task and failure behavior.

Evidence to request: A production feature architecture with justified managed-versus-custom choices and component-level evaluation of retrieval or agent actions.

AI backend and authorization engineer

Carries tenant and user permissions through data retrieval and executable tools while preserving application transaction semantics.

Evidence to request: Denied-access tests, deletion propagation and recovery from failed or repeated actions across representative customer tenants.

AI evaluation and application test engineer

Creates release evidence for groundedness, task success, latency, harmful failures and complete per-task consumption.

Evidence to request: A versioned regression dataset, explained scoring decisions and reproducible investigation of a prior model or tool regression.

MLOps and AI observability engineer

Integrates model changes into deployment, feature flags and incident handling that the internal engineering team can own.

Evidence to request: Reproducible infrastructure, useful traces, rollback evidence and a successful handover using customer-controlled repositories and accounts.

11 AI implementation milestones and acceptance gates

Choose a product boundary

Select one user task and document who can access or change which records.

Acceptance gate: The feature has an owner, prohibited actions and examples from real usage.

Create a versioned evaluation harness

Build held-out requests covering product versions, permissions, ambiguous inputs and API failures.

Acceptance gate: The internal team can run it without the supplier and inspect error classes.

Release behind a control

Use a limited tenant or feature flag, with fallback to the existing product behavior.

Acceptance gate: Latency, quality and incident visibility satisfy the agreed product thresholds.

Transfer operational ownership

Rehearse a model or index rollback and a support investigation before broad rollout.

Acceptance gate: Internal engineers can identify the failed configuration and restore service.

12 Project scope and quotation questions

Tenant and access complexity

Document-level permissions, regional segregation and many customer-specific schemas can dominate implementation.

Ask in the quotation: Which access checks are native to the existing platform and which require custom engineering?

Evaluation breadth

Each product version, language and action adds regression scenarios beyond a happy-path demo.

Ask in the quotation: Who maintains tests as our product APIs and help content change?

Usage shape

Long context, repeated retrieval, concurrent requests and tool chains influence cost per completed task.

Ask in the quotation: Can we see task-level usage and enforce limits without breaking the experience?

Operational coverage

Availability commitments and out-of-hours support require monitoring and incident ownership.

Ask in the quotation: Is support for the model integration included in the same service responsibility as the application?

13 Hiring and comparing AI implementation partners

Production AI engineering contribution

Ask: Which deployed AI feature did the named engineers build and operate?

Useful evidence: A redacted architecture, evaluation history and explanation of their code and incident responsibilities.

Warning sign: A collection of model demos replaces evidence of product integration.

Candidate notes

Architecture and dependency judgment

Ask: How will you separate retrieval failures, model failures and failed tool actions?

Useful evidence: Component-level tests, traces and a documented build-versus-buy assessment.

Warning sign: All errors are treated as prompt-tuning issues.

Candidate notes

Transparent workload economics

Ask: What happens to cost when context length, retries or concurrent customers grow?

Useful evidence: A task-level estimate covering model, search, storage, evaluation and observability.

Warning sign: Only the cheapest advertised model request is used for the budget.

Candidate notes

Team independence after handover

Ask: Can our engineers deploy, test and change the feature without your private accounts?

Useful evidence: Owned repositories, reproducible infrastructure, evaluation assets and a completed handover exercise.

Warning sign: Critical orchestration remains hidden inside a supplier-controlled service.

Candidate notes

14 Demonstrations, trial terms and hiring commitments

Ask whether a no-cost demonstration, sandbox or representative-sample evaluation is available. A free consultation does not imply free implementation; custom pilots are often paid. Agree workflow scope, what is free, usage limits, when billing starts, data handling, acceptance evidence, output ownership and the exit path before supplying data.

Relevant demonstration: A source-linked feature with permission and stale-document tests.

Failure to introduce: An answer can be fluent and still reveal a document the user is not permitted to access.

15 Practical AI buying tips

- Include support engineers in acceptance; they need enough context to investigate errors without reading sensitive customer content unnecessarily.
- Test a permission revocation after content has already been indexed and cached.
- Avoid a parallel production stack unless there is an explicit architectural reason and a team to operate it.

16 Questions buyers ask about AI implementation

What should an AI engineering discovery phase deliver?

Expect a feature boundary, data and permission assessment, architecture options, evaluation design and operating-cost model. The output should explain which managed, reusable and custom components are justified. A read-only answer feature differs materially from an agent executing customer actions. Permission-aware retrieval and data lifecycle work can dominate a seemingly small assistant.

How should we test a product knowledge assistant for grounded answers and tenant isolation?

Evaluate answers against sources the specific user and tenant can access, including denied documents and deleted content. Score unsupported answers and retrieval failures as well as helpful responses. Test account or session changes server-side, not just hidden interface elements. A grounded-looking answer that exposes another tenant's information fails acceptance regardless of fluency or aggregate model quality.

What must an AI feature integrate with identity, search and our release pipeline?

Carry authenticated identity into retrieval and tool calls, and connect source changes or deletions to index maintenance. Define bounded application actions with traceable results and retry behavior. The release integration should support staging, feature flags, versioned evaluation and rollback. Ask the developer to demonstrate denied access and failed actions using your architecture before estimating the feature as a standalone chatbot.

How do we estimate production AI costs?

Use complete successful tasks, including retrieval, reasoning, tool calls, retries and review. Separate development and evaluation traffic from customer usage, then include monitoring and support effort. Estimate full context, retries, agent steps and concurrency rather than one ideal request. Include document updates, permission filters and deletion processing.

How do context length, agent steps and concurrency affect AI feature running costs?

Count the full request path: input and output context, retrieved passages, retries, reranking and every agent step. Concurrency can change capacity requirements even when average request volume is modest. Add index refresh, traces and regression traffic to inference charges. Require per-workflow assumptions and observed test consumption so a low model price is not mistaken for the feature's actual operating cost.

What should we own when an AI development partner uses its existing application framework?

Separate the framework licence from code written for your product and from third-party model or hosting accounts. Agree access to prompts, orchestration, permission mappings, indexes and evaluation assets. Confirm what can be modified or transferred and what fees continue. A reusable shell can accelerate delivery, but your identity model, transaction behavior and release acceptance remain implementation work that must be priced and verified.

How do we assess an AI developer’s production experience?

Ask for the named engineer’s role in integration, evaluation, deployment and incident handling. Review redacted traces or test artifacts and have them explain a failure and its correction. Useful evidence includes a redacted architecture, evaluation history and explanation of their code and incident responsibilities. Ask the proposed team which deployment and support responsibilities they personally held.

What should AI feature handover include?

Receive source, infrastructure configuration, dependency licences, evaluation datasets and runbooks. Your team should deploy a change, investigate a trace and roll back a model release during acceptance. Train engineers to trace retrieval, model and tool failures independently. Agree maintenance scope for evaluations and migrations separately from a defect warranty.

What should a production AI engineer demonstrate beyond a working model demo?

Ask the engineers to demonstrate the feature inside a test tenant, then attempt retrieval using a user denied access to the source. Trace a failed tool call, inspect the evaluation record and roll back the deployed behavior. Request full request-cost evidence, including retries and search, plus a clear explanation of their code, integration and incident responsibilities. A model endpoint or prompt collection is insufficient.

17 Primary sources and editorial scope

1. OpenTelemetry: Signals

Primary technical background on traces, metrics and logs; the product evaluation and architecture recommendations are original buyer guidance.

https://opentelemetry.io/docs/concepts/signals/

2. Google SRE: Monitoring Distributed Systems

Operational reference for monitoring and user-facing symptoms; it is not a guarantee of reliability for a proposed AI feature.

https://sre.google/sre-book/monitoring-distributed-systems/

3. Microsoft: RAG design and evaluation

Primary software architecture and evaluation guidance; service-selection recommendations are editorial.

https://learn.microsoft.com/en-us/azure/architecture/ai-ml/guide/rag/rag-solution-design-and-evaluation-guide

4. Microsoft: RAG evaluators

Primary documentation of evaluator categories and their limitations; no benchmark performance is implied.

https://learn.microsoft.com/en-us/azure/foundry/concepts/evaluation-evaluators/rag-evaluators

Original AI-service buying guidance. Verify current interfaces, licences, data rights and provider evidence. No supplier certification, project quotation or guaranteed outcome is implied.