

✦ TEMPLATE · TEMPLATES

Architecture Review Template

A structured review for AI system architectures: context, components, quality attributes, data and model choices, risks and decisions — scored, recorded and signed.

FORMAT	LEVEL	READ TIME	UPDATED
Template	Beginner	Fill-in	Oct 2026

OVERVIEW

What this template covers

An architecture review is cheapest before the first production release and most expensive after it. This template guides a review of an AI system — RAG, agent, or AI feature inside a product — against the quality attributes that matter in production: correctness, security, latency, cost, operability and change. It ends with recorded decisions, not just comments.

KEY TAKEAWAYS

- Review against explicit quality attributes, prioritized for this system.
- Separate the parts that change often (prompts, models) from those that should not.
- Every model and data choice should be swappable without a rewrite.
- Record decisions with their reasons so the next team does not reopen them blindly.
- Score honestly; a review where everything is green taught nobody anything.

WHO IT'S FOR

- Solution and enterprise architects
- Engineering leads and principal engineers
- CTOs approving designs
- Platform and SRE teams

HOW TO USE IT

The presenting team fills sections 01–02 and shares them a few days ahead; reviewers score section 03, raise risks in section 05 and record decisions in section 06 before everyone leaves the room. Every box and table is shown filled in with a worked example — a policy Q&A assistant for a fictional hospital group — so you can see the level of detail expected. Replace the examples with your own.

CONTENTS

01 Context and drivers
02 Architecture overview
03 Quality attribute scorecard
04 Design checklist
05 Risks and open questions
06 Decisions and sign-off

01 / CONTEXT

Context and drivers

Reviewers need the why before the boxes and arrows.

System and purpose

01

What it does, for whom, and the business outcome it supports.

WORKED EXAMPLE

Policy Assistant — a chat assistant on the staff intranet that answers nurses', doctors' and admin staff's questions from the group's 2,400 clinical and HR policies, with a citation to the exact policy section. Goal: cut time spent hunting for policies and calls to the quality office.

Key requirements and constraints

02

Expected users and load, latency needs, data residency, regulatory obligations, approved providers, budget.

WORKED EXAMPLE

- 6,000 staff across five hospitals; peak about 40 questions a minute at shift change
- p95 answer time under 6 seconds
- Data stays in India-region hosting; approved model providers only
- Some policies are restricted to roles (e.g. pharmacy, HR investigations)
- Run budget ceiling ₹2.5 lakh per month

Top three quality attributes, in priority order

03

e.g. "1. Answers never leak restricted documents. 2. p95 response under [x] seconds. 3. Swap model providers within a sprint."

WORKED EXAMPLE

- Never show a restricted policy to someone outside its role group
- Every answer cites a current policy version; superseded versions never appear
- p95 response under 6 seconds at shift-change peak

Review scope

04

What is up for review now, and what is fixed or out of scope.

WORKED EXAMPLE

In scope: retrieval, access control, model gateway and evaluation for the pilot release. Fixed: the intranet single sign-on and the document management system as the source of truth. Out of scope: patient-specific questions, which the assistant refuses.

✦ 02 / DESIGN

Architecture overview

Attach the diagram. Use the layers below as a prompt for what it should show.

LAYER 01 **Experience**	Web / mobile / chat · Auth & session · Feedback capture
↓	
LAYER 02 **Orchestration**	API gateway · Workflow / agent loop · Prompt registry · Guardrails
↓	
LAYER 03 **Intelligence**	Model gateway · Retrieval & rerank · Tool adapters · Caching
↓	
LAYER 04 **Data**	Sources & connectors · Indexes · Operational DB · Audit logs
↓	
LAYER 05 **Operations**	Tracing · Evaluation in CI · Cost & latency dashboards · Secrets & IAM

Components and responsibilities

01

Each component, what it owns, and what it must not do.

WORKED EXAMPLE

- **Chat API** — auth, session, rate limits; never calls a model directly
- **Orchestrator** — query rewrite, retrieval, prompt assembly, citation check
- **Retrieval service** — hybrid search + rerank, filters by the user's role groups
- **Model gateway** — the only path to model providers; logging, fallback, cost caps
- **Ingestion job** — syncs policies nightly and on publish; removes superseded versions

Model and data choices

02

Models per task and why; embedding and index choices; how each could be replaced.

WORKED EXAMPLE

- Answering: a mid-size hosted model via the gateway; a second provider configured as fallback
- Query rewrite and refusal check: a small, fast model
- Embeddings: multilingual model, 1,024 dimensions; index in the existing Postgres with a vector extension
- All model names live in gateway config, so a swap is a config change plus an eval run

Integration points

03

External systems, protocols, auth method and failure behavior for each.

WORKED EXAMPLE

- Intranet SSO (OIDC) — role groups read from the token; fail closed if groups are missing
- Document management system (REST, service account) — on failure, keep the last good index and alert
- Model providers (HTTPS via gateway) — 8-second timeout, then fallback provider

✦ 03 / SCORING

Quality attribute scorecard

Score each attribute 1–4 using the scale. Write evidence, not opinion.

Scoring scale

SCORE	MEANING
4 — Strong	Designed for, tested, and evidenced
3 — Adequate	Designed for; evidence partial or pending
2 — Weak	Acknowledged but not addressed in the design
1 — Missing	Not considered

Scorecard

ATTRIBUTE	KEY QUESTION	SCORE	EVIDENCE / GAP
Answer quality	Is there an evaluation set gating releases?	3	220-question golden set exists; not yet wired into CI
Security & access	Are permissions enforced at retrieval and tool level?	4	Role filter in retrieval query; tested with 40 cross-role probes
Privacy	Is personal data minimized, masked and deletable?	3	Staff IDs hashed in logs; deletion job designed, not built
Performance	Is the latency budget allocated per stage?	2	Only end-to-end target; no load test at shift-change peak
Cost	Is cost per request known, capped and monitored?	3	About ₹0.40 per answer measured; monthly cap set, no alerts yet
Reliability	What happens when a model provider fails?	4	Gateway failover tested in staging
Observability	Can any answer be traced end to end?	3	Traces include chunks and prompt version; no search UI yet
Changeability	Can models, prompts and indexes change independently?	4	Models in gateway config; prompts versioned; blue-green index

✦ 04 / CHECKS

Design checklist

Common gaps we find in AI architectures.

BOUNDARIES

- [ ] Model calls go through one gateway, not scattered SDK calls
- [ ] Prompts are versioned outside application code
- [ ] Generation never reaches past retrieval into raw sources

DATA

- [ ] Re-indexing is a job, not a migration
- [ ] Deletes in source systems reach the index
- [ ] Index and cache hold access metadata

OPERATIONS

- [ ] Timeouts and fallbacks on every external call
- [ ] Evaluation runs in CI on prompt, model and retrieval changes
- [ ] Per-tenant cost and rate limits

✦ 05 / RISKS

Risks and open questions

Capture every concern raised in the session, with an owner.

#	RISK OR QUESTION	IMPACT	OWNER	NEXT STEP
01	No load test at shift-change peak	High	Platform lead	Load test at 2× peak by 6 Nov
02	Golden set not gating releases	High	ML engineer	Add eval job to CI; block on regression

#	RISK OR QUESTION	IMPACT	OWNER	NEXT STEP
03	Superseded policies may linger if a sync fails	Medium	Data engineer	Alert on sync failure; nightly diff
04	Deletion of staff chat logs not built	Medium	Privacy officer	Build 90-day purge job before pilot
05	Role-group changes need a full re-index	Low	Architect	Measure re-index time; decide in Q1
06	How should the assistant answer clinical dosing questions?	Medium	Director of Quality	Agree a refusal-and-redirect wording

06 / DECISIONS

Decisions and sign-off

Write each decision as a short architecture decision record.

Decision 1 — context, decision, consequences

01

e.g. “Use hybrid search with reranking because exact product codes must match; adds one hop of latency.”

WORKED EXAMPLE

Context: restricted policies must never leak. **Decision:** enforce role filters inside the retrieval query, not after generation; chunks carry role-group metadata. **Consequences:** re-index needed whenever role groups change; slightly smaller candidate sets.

Decision 2 — context, decision, consequences

02

WORKED EXAMPLE

Context: staff search by policy codes like “HR-POL-114”. **Decision:** hybrid search (keyword + vector) with a reranker. **Consequences:** adds about 300 ms per query; exact codes now match reliably.

Decision 3 — context, decision, consequences

03

WORKED EXAMPLE

Context: a single provider outage would take the assistant down at shift change. **Decision:** second provider behind the gateway with automatic failover. **Consequences:** both providers must pass the eval set before each release.

Outcome

04

Approved / approved with conditions / revise and re-review. Conditions and re-review date.

WORKED EXAMPLE

Approved with conditions: close risks 01 and 02 before pilot; re-review on 20 Nov 2026 with load-test results.

Reviewers — names, signatures, date

05

WORKED EXAMPLE

- D. Rao, Principal Architect · approved with conditions · 14 Oct 2026
- L. Thomas, Head of Information Security · approved with conditions · 14 Oct 2026
- J. Varghese, Director of Quality · approved · 15 Oct 2026

✦ GET IN TOUCH

Want us to run this with your team?

We run independent architecture reviews of RAG systems, agents and AI products, and leave you with scored findings and recorded decisions.

- ✦ Independent architecture reviews
- ✦ Model and retrieval design
- ✦ Production-readiness hardening

BOOK AN AI STRATEGY SESSION →

EMAIL
hello@splendenslabs.com

WEB
splendenslabs.com

OFFICE
Sona Incubation Foundation, Salem