Keep assurance current as the system changes.

A review cadence for AI evaluations, access controls, dependencies and operating evidence.

Recurring assurance register

Review	Evidence to refresh
Model and prompt changes	Evaluation results against the current baseline
Access changes	Service identities, privileged roles and revoked users
Dependency changes	Exposure assessment and verified remediation
Recovery readiness	Recent restoration and escalation exercises

Trigger reviews with meaningful changes

A calendar review is useful, but a new model, a privileged integration or a major dependency change may need evidence before its release.

Retain the decision history

Keep the scope, findings and accepted limitations of each review. The next review should explain what changed rather than restart from an empty template.

Choose the evidence that needs to stay current

Assurance work should focus on the conditions that can change after an initial review. These may include application dependencies, user permissions, AI source collections, model configurations and recovery procedures. Identify the controls relied on by the business and the evidence needed to show that those controls still work.

Create a review schedule based on the nature of the change and its consequence. Some checks belong in every release, while others follow a meaningful infrastructure change or an incident. Record the owner of each check and where results are kept. A calendar alone is not an assurance process if nobody can explain what is being checked or what happens when it fails.

Investigate changes in results

Compare new results against a known baseline. If response time has increased, investigate the affected workflow and environment before treating the change as normal variation. If an AI evaluation regresses, identify which cases changed and whether the failure affects the approved operating scope.

Keep the underlying evidence available. A green dashboard or aggregate percentage may conceal a failure in a small but important group of tasks. Review exceptions by business consequence and assign corrective work. Where a result is accepted temporarily, record the reasoning, owner and review date. Repeated exceptions should prompt a design discussion rather than become an invisible part of normal operation.

Use the review to support a decision

Each assurance cycle should conclude with a clear account of what was examined, what passed, what did not and what happens next. Distinguish observed evidence from areas outside the review scope. Do not describe a limited technical check as a certification of the entire business or application.

Link findings to the delivery backlog and follow them through to verification. Share a concise explanation with the people responsible for the affected workflow, with supporting detail available to engineers. The value of recurring assurance is that it keeps operating decisions grounded in current evidence. Producing the same report every month without examining change provides little protection against new failure modes.

Does a previous review cover a new model release?

Only where its evidence remains applicable. Re-evaluate the behaviour and controls affected by the change.

Read the current resource online

https://cobnex-review.rgcsekaraa.workers.dev/plans/enterprise/interna/