BEYOND THE POC

IDEA → POC → PILOT → PRODUCTION → SCALE

Building governed GenAI applications

A practical architecture- and governance-driven cheat sheet for moving from isolated experimentation to repeatable success.

ARCHITECTURE

Direction + technical coherence

Defines the production destination and the reusable technical decisions that keep rapid experimentation aligned with enterprise realities.

- **Target state**
  Define users, systems, service levels, deployment environment, data boundaries, and who owns the production capability.
- **Reusable patterns**
  Use approved components and reference patterns for prompts, RAG, agents, integrations, evaluation, and observability.
- **Integration strategy**
  Map APIs, identities, data flows, upstream/downstream dependencies, fallbacks, and failure handling.
- **Security posture**
  Apply least privilege, data classification, secrets management, isolation, threat testing, and secure defaults from day one.
- **Scale + portability**
  Test volume, latency, cost, model substitution, provider lock-in, resilience, and deployment options before scale.
- **Production ownership**
  Name the teams responsible for releases, monitoring, incidents, model/data changes, and long-term support.

CONTINUOUS FEEDBACK & LEARNING

Evidence into insight. Insight into better decisions.

CHECK THE SIGNAL → DIAGNOSE THE LAYER → CHANGE THE RIGHT THING

A weak output may be caused by the use case, workflow, data, retrieval, prompt, model, control, or operating environment.

GOVERNANCE

Visibility + decision discipline

Creates shared decision rights, guardrails, evidence, and cross-team visibility so fast-moving work produces accountable and consistent outcomes.

- **Decision rights**
  Define who approves use cases, models, data access, controls, releases, exceptions, and changes in autonomy.
- **Ownership**
  Assign accountable business, product, technical, risk, data, security, and operational owners—not only contributors.
- **Policies + guardrails**
  Translate enterprise policy into permitted uses, prohibited actions, review thresholds, and enforceable controls.
- **Risk + controls**
  Assess privacy, security, bias, hallucination, harmful action, vendor, legal, and operational risks; map mitigations.
- **Human oversight**
  Specify when people review, approve, override, or escalate decisions; preserve traceability and meaningful control.
- **Cross-team visibility**
  Maintain a common intake, decision log, architecture record, risk status, metrics, exceptions, and reusable lessons.

THE EXECUTION SYSTEM

01 STRATEGY & TARGET STATE

Solve the right problem and define the outcome before optimizing the technology.

- **Problem + outcome**
  Define the current pain, affected users, business impact, and desired future state.
- **Why GenAI**
  Compare with search, rules, workflow automation, analytics, and traditional ML. Use GenAI only where it adds unique value.
- **Scope + autonomy**
  Set in/out boundaries, users, decisions, and whether the system informs, recommends, or acts.
- **Success measures**
  Baseline quality, time, cost, adoption, experience, and risk. Set pilot and production thresholds.

OUTPUT Use-case brief + outcome scorecard

02 STAKEHOLDERS & WORKFLOW

Align the people, workflow, ownership, and adoption path—not only the feature backlog.

- **Outcome owner**
  Name the sponsor accountable for value, priorities, tradeoffs, adoption, and acceptance.
- **Workflow fit**
  Map current/future steps, handoffs, review points, exceptions, downstream impact, and human judgment.
- **Decision lenses**
  Represent users, domain, product, architecture, data, security, risk, legal, evaluation, and cost.
- **Engagement cadence**
  Use demos, design reviews, risk forums, testing, and feedback channels to expose misalignment early.

OUTPUT Workflow map + ownership/RACI

03 DATA & KNOWLEDGE

Ground the application in trusted, current, accessible information and make its evidence visible.

- **Authoritative sources**
  Identify systems of record, content owners, freshness, conflicts, and fallback behavior.
- **Access + handling**
  Apply entitlements, privacy, minimization, retention, residency, and use restrictions.
- **Retrieval + context**
  Define chunking, metadata, ranking, context assembly, citations, limits, and abstention.
- **Quality + lineage**
  Track source-to-output lineage, relevance, staleness, coverage gaps, and knowledge-base

OUTPUT Data/knowledge plan + source map

04 EVALUATION & OPERATIONS

Prove quality, operate reliably, and convert production evidence into prioritized improvement.

- **Evaluation design**
  Use representative test sets, task metrics, human review criteria, thresholds, and baseline.
- **Safety + reliability**
  Test groundedness, hallucination, bias, harmful output, injection, edge cases, and failures.
- **Monitoring + economics**
  Track quality, latency, availability, token use, infrastructure cost, drift, incidents, and support.
- **Change control**
  Version prompts, models, data, and policies. Regression-test changes before release.

OUTPUT Evaluation scorecard + production runbook

REQUIRED FEEDBACK LOOPS

Use each loop to decide whether to continue, correct, constrain, redesign, or scale.

BUSINESS VALUE
Are we still solving the right problem?
Review realized outcomes, adoption, benefit versus baseline, and changing priorities.

USER + WORKFLOW
Does it fit how people actually work?
Observe friction, trust, overrides, handoffs, training needs, and unmet user needs.

DATA + MODEL
Why did the system produce this result?
Inspect sources, retrieval, context, prompts, model behavior, citations, and abstentions.

ARCHITECTURE
Are we moving toward a viable target state?
Review dependencies, performance, resilience, cost, scale, portability, and technical debt.

GOVERNANCE + RISK
Are boundaries and controls still appropriate?
Review incidents, exceptions, policy changes, control effectiveness, autonomy, and accountability.

OPERATIONS + ECONOMICS
Is it reliable, supportable, and sustainable?
Track uptime, latency, cost per task, support load, release quality, and production failures.

PORTFOLIO LEARNING
What should other teams reuse or avoid?
Publish patterns, decisions, benchmarks, lessons, anti-patterns, and reusable controls.

Every decision lens must be represented. Every lens does not require a different person.

ALIGNED → VISIBLE → CONSISTENT → TRUSTED → REPEATABLE