TOOL REGISTRY

Tool Registry Planning Template

A worksheet for defining read tools, write tools, approval-gated tools, schemas, permissions, fallbacks, and receipts.

WHAT THIS TEMPLATE HELPS YOU DECIDE

Give agentic workflows a deterministic operating layer so tools carry state, permissions, validation, execution, and evidence.

BEST FOR

- Teams designing agentic or workflow automation systems
- Builders moving from prompt wrappers to tool execution
- Operations where writes require approvals and receipts

OUTPUTS

- Tool inventory
- Permission and schema map
- Execution receipt design

STEP 1

Frame the operating need

Start with the workflow, decision, owner, and business pressure. The template is useful only when it is grounded in a real operating moment.

Operating frame

Workflow supported
Name the workflow this registry enables.

Systems involved
CRM, inbox, database, calendar, documents, analytics, or internal tools.

Highest-risk write action
Name the action that most needs approval, validation, or fallback.

Readiness check

- [ ] Read tools are separated from write tools
- [ ] Every tool has a typed input and structured output
- [ ] Permissions are checked before execution
- [ ] High-risk writes require approval preview
- [ ] Tool failures create visible fallback or retry paths
- [ ] Execution receipts are stored for review

STEP 2

Map the architecture questions

Use this page to separate the parts of the system that need design before anyone jumps to tools, prompts, or implementation details.

Design map

Read tool	What state can the system retrieve safely?
Write tool	What state can the system change, create, send, or assign?
Validation	What schema, permission, or business rule must pass first?
Approval	Which actions need human approval before execution?
Receipt	What result proves the tool completed correctly?

A tool registry is where agentic systems become operational. Without tool contracts, agents improvise around business state.

STEP 3

Turn the answers into a brief

A strong brief makes the next decision easier: proceed, defer, redesign, govern, or assess more deeply before implementation.

Decision brief

Tool owner	Who owns each tool contract and its failure behavior?
Fallback behavior	What happens if the tool fails or returns incomplete state?
Observability	Which logs, traces, IDs, and status views are required?
Expansion rule	What must be true before adding more write capabilities?