Free appendix from Agents in Production

Tool and MCP server design checklist

Free appendix from Agents in Production: the checks that make a tool easy for a model to use and hard for an attacker to abuse

Written by AI agents (Claude), directed by Yann

2026 to 2027 edition

B

Tool and MCP server design checklist

The checks that make a tool easy for a model to use and hard for an attacker to abuse

Builder Solo Ops

The reader of a tool is a model: a vague name, a stack trace or 40,000 tokens of JSON confuse it. This checklist turns Chapters 3, 4 and 11 into review items.

At a glance: what you will get

- Design checks: names, descriptions, parameters, returns, errors.
- Safety checks: side effects, auth, supply chain.
- MCP checks for revision (2026-07-28, as of late 2026).

How to use this checklist

Use it at design review and as the pull request template. Each item points at code, schema or an eval result.

1. Shape and granularity

- [ ] The tool maps to an agent task (`schedule_event`), not one raw endpoint (Anthropic).
- [ ] Steps that always run together are one tool.
- [ ] A few high-impact tools, not every endpoint.

- [ ] No confusable near-duplicate tool exists.

2. Names and descriptions

- [ ] The name is namespaced by service or resource (`asana_projects_search`).
- [ ] The description says what the tool does and when (not) to use it.
- [ ] The description shows one correct call.
- [ ] It gives no instructions about other tools.

3. Parameters

- [ ] Parameter names are unambiguous (`user_id`, not `user`).
- [ ] Enums replace free text where values are known.
- [ ] Required and optional parameters are explicit, with defaults.
- [ ] The schema uses JSON Schema 2020-12, the default since the 2025-11-25 spec.
- [ ] The server validates all inputs.

4. Return values

- [ ] Results carry names, not bare UUIDs.
- [ ] Large payloads offer a `response_format` (concise or detailed).
- [ ] Results are paginated, filtered or truncated with a default size cap.
- [ ] `structuredContent` is provided when the caller parses the result.
- [ ] Empty results are explicit ("0 rows matched filter X").

5. Errors

- [ ] Errors say how to fix the call (parameter, valid values), not a stack trace.

- [ ] Validation failures return as tool execution errors so the model can self-correct (SEP-1303).
- [ ] Upstream HTTP errors map to tool errors; a 500 is never success.
- [ ] Results distinguish retryable from non-retryable errors.

6. Side effects and safety

- [ ] Read-only tools set `readOnlyHint: true`; others are writes, with `destructiveHint`, `idempotentHint` and `openWorldHint` explicit (defaults assume the worst).
- [ ] The server enforces scopes and blocks writes itself: annotations are untrusted hints.
- [ ] Write tools are idempotent or accept an idempotency key.
- [ ] Destructive or external actions offer a dry run or human confirmation.
- [ ] Dangerous capabilities that are not needed are removed, not guarded.
- [ ] Rate limits and quotas protect downstream systems from loops.

7. MCP transport, auth and state

- [ ] Remote servers use Streamable HTTP over HTTPS, not the deprecated HTTP+SSE transport.
- [ ] The server checks every token was issued for it (no token passthrough).
- [ ] Scopes are minimal per tool, no wildcards; step-up uses `WWW-Authenticate`.
- [ ] Registration uses Client ID Metadata Documents; deprecated Dynamic Client Registration stays only for old clients.
- [ ] Nothing depends on `Mcp-Session-Id` or `initialize` (both removed); the server implements `server/discover` and keeps state in server-minted handles.
- [ ] New servers avoid deprecated Sampling, Roots and Logging.

- [ ] State handles are bound to the user: holding a handle is not authentication.
- [ ] OAuth and outbound URLs are validated (http(s) only, private ranges blocked, no shell).
- [ ] `tools/list` returns tools in a deterministic order to help prompt caching.
- [ ] A dated migration plan serves the previous revision until main clients support 2026-07-28 (Claude support rolling out, late 2026) and drops deprecated features within 12 months.

8. Operations

- [ ] The server propagates OpenTelemetry trace context from `_meta`.
- [ ] Each call emits a span: tool name, latency, result size, error status.
- [ ] Local stdio servers run sandboxed, with filesystem and network allowlists.
- [ ] Secrets come from a vault or keychain, never config literals.
- [ ] Server and dependencies are pinned; releases log description changes.

9. Evals

- [ ] Realistic multi-step tasks exercise the tool, with held-out cases.
- [ ] Evals track accuracy, tool calls, tokens and errors; someone reads transcripts.
- [ ] A description or schema change runs the tool's evals in CI (end-to-end tests: Chapter 9).

Anti-pattern **The CRUD wrapper server**

A server generated from a 120-endpoint OpenAPI spec gives the model 120 tools: wrong picks, five calls where one would do, flooded context. Cut to the tools the agent needs. "Date must be ISO 8601; you sent 28/11" is cheaper than a model change.

Appendix B on one page

Tool and MCP server design checklist

Design tools for a model reader: few, task-shaped, clearly named, and hard to abuse.

Key takeaways

- Workflow-shaped tools beat one tool per endpoint
- Return less, and return errors that teach the model how to recover
- Treat every non-read-only tool as a write and align servers with the 2026-07-28 spec

Do this on Monday

- Pull last week's traces and list the tool calls that failed or were retried
- Rewrite your three most-failed tool error messages
- Have a teammate read one tool description cold and say when they would call it

Keep going: the full book

This was a free excerpt from **Agents in Production**, a playbook for taking agents from pilot to production and keeping them there. The full book has a prologue, 21 chapters in nine parts, an afterword on how the book was made, and the production readiness checklist, plus a companion workbook and free online extras.

The full book covers:

- **Mental models:** what an agent actually is, and when a plain workflow beats one.
- **Building blocks:** tools agents can actually use, MCP after the 2026-07-28 revision, skills, subagents and plugins, context engineering and memory.
- **Frameworks and architecture:** a vendor-neutral map of the 2026 framework landscape, a decision guide, and reference architectures for event-driven and durable agents.
- **Production:** reliability harnesses, evals in CI, observability and unit economics, security for agents and MCP, governance and the EU AI Act.
- **Business:** no-code automation recipes, solo playbooks that make or save money, and a build, buy and staff plan for leaders.
- **The agentic software factory:** agents that take software from idea to merged PR, test it end to end and keep improving it, with a capstone that ships one system through every station.
- **The road ahead:** a use-case atlas, the evidence on productivity, jobs, cost and energy, and sourced predictions.
- **Appendix:** the production readiness checklist (59 checks, 19 must pass). Online and free: a tool and MCP server design checklist, a threat model template, the 2026 field notes, a glossary and the briefs of every agent that wrote the book. The companion workbook holds the exercises and their solutions.

Four reading tracks (Builder, Leader, Solo, Ops) tell you exactly which chapters to read and which to skip. Exercises, a running project and self-checks come with full solutions.

Written by AI agents (Claude, a model made by Anthropic) and directed by Yann.

Get it at {{STORE_URL}}