The BenchSlap Tool Catalog

Eight AI tools, one verification kernel — what each tool does, the contract it honors, and how the kernel keeps them honest.

Author: Richard L. Sanders · Utah Bar #15728 **Role:** Sole inventor and implementer
Version: 1.0 — 2026-05-13 **Companion papers:** All BenchSlap subsystems (00-08); this paper is the surface-level catalog. **Source:** `tools/*.js`, `routes/tools/*.js`, `lib/advocacy-algorithm.js`, in the public repository at <https://github.com/Benchslap/Benchslap>

Abstract

BenchSlap exposes eight user-facing AI tools, each tuned to a distinct legal-practice task: chat assistant, document drafter, citation auditor, evidence preprocessor, case-strength strategist, deep-context advisor, multi-AI commission, and elite council. They share one verification kernel (V4 + AEGIS + AEGIS PRIME), one corpus (3.2M opinions + 1.9M rules), and one identity (the user's silo). The differentiation is in what each tool *outputs*, not in how each tool reasons.

This paper documents the catalog: input contract, output shape, model tier, gate behavior, and the case for why eight tools and not one. The seven companion papers describe the kernel each tool plugs into; this paper describes the surface a user touches.

---

1 · Why eight, not one

A lawyer who needs a verified citation does not need the same shape of answer as a lawyer who needs a draft motion. A lawyer asking "is this case worth pursuing?" does not need the same shape of answer as a lawyer asking "what should the panel vote say in my appeal brief?" The kernel reasoning is the same — the same V4 algorithm, the same closed corpus, the same AEGIS-pinned authorities. The product surface is different because the task is different.

Consolidating to one chat tool would force the user to phrase every task as a question. Splitting into eight specialized tools lets the user select the SHAPE of output they want: a draft (Drafter), a verification verdict (Auditor), a quick health score (Strategist), a long-

form case theory (Advisor). The reasoning is the same; the deliverable is what the user came for.

Two architectural rules apply to every tool:

1. **The verification gate fires on every tool's output.** No tool returns an answer to the user that has not passed AEGIS text containment (paper 01) and AEGIS PRIME structural verification (paper 08). The gate is at the post-stream layer (`lib/post-stream-gate.js`) and is not bypassable.
2. **Every tool reads from the same silo context.** Linked silos (`lib/silo-context.js`) propagate documents and facts across related matters so a tool answering a question on Matter A can see the depositions filed in Matter B if the two are linked.

---

2 · The catalog

2.1 — Reckoner (Consigliere)

Endpoint: `POST /api/chat` **File:** `tools/consigliere.js` **Tier:** Free + PRO **Purpose:** General-purpose chat assistant with silo context.

Reckoner is the front door. The user asks any question — about a case in their silo, about a legal rule, about the meaning of a phrase in a document — and Reckoner answers in conversational prose with citations attached. It is the most-used tool by volume because every other tool's output is also editable by Reckoner: "rewrite this paragraph more aggressively," "shorten the procedural-posture section," "explain this citation in simpler terms."

Input shape: a free-form question plus an optional silo ID. Output shape: a streamed chat response with inline citations, each AEGIS-verified before display.

2.2 — Drafter

Endpoint: `POST /api/draft` (generate) + `POST /api/draft/improve` (refine an existing document) + `POST /api/draft/chain` (multi-step chain) **File:** `tools/drafter.js` **Tier:** Free + PRO **Purpose:** Generate or improve legal documents.

Drafter is the workhorse. It produces motions, briefs, complaints, answers, affidavits, declarations, demand letters, settlement proposals, deposition outlines, and every other adversarial-prose artifact a litigator generates. Output is markdown with structural elements (caption block, signature block, certificate of service, exhibit references) that respect the jurisdiction's procedural rules — Utah URCP for Utah, soft advisory for other jurisdictions per the `feedback_compliance_gate.md` design (see paper 09's companion test file `tests/jurisdiction-drafting.test.js`).

The `/api/draft` endpoint generates a new document from a prompt + silo facts. The `/api/draft/improve` endpoint takes an existing document and refines it (replaces the legacy `/api/fix` endpoint, which remains as a backwards-compatibility alias but is not enumerated as a separate tool). The `/api/draft/chain` endpoint sequences multiple drafts that build on each other — the motion, then the proposed order, then the certificate of service, all consistent with each other.

Output shape: streamed markdown with structural awareness; every citation AEGIS-verified.

2.3 — Auditor

Endpoint: `POST /api/audit` **File:** `tools/auditor.js` **Tier:** Free + PRO **Purpose:** Verify documents and flag issues.

Auditor reads an existing document (drafted by Drafter, drafted by opposing counsel, or drafted by the user themselves) and returns a structured report: every citation classified (VERIFIED / FABRICATION / FICTION / NON_SEQUITUR / UNVERIFIED per paper 05's V4 truth table), every factual claim cross-checked against silo evidence, every rule reference checked against the local corpus.

The output is a markdown audit report with two halves: a quantitative summary (X claims verified, Y fabricated, Z unverifiable) and a qualitative narrative (the specific claims, the specific authorities, the specific risk to the user). The Auditor is what you run before you file. It is also what you run on the brief opposing counsel just served you, so you can quote-tweet their fabrications back at them in your reply.

Output shape: streamed markdown report with severity-ranked issues.

2.4 — Preprocessor

Endpoint: `POST /api/preprocess` **File:** `routes/tools/preprocessor.js` **Tier:** Free + PRO **Purpose:** 7-phase document classification pipeline.

Preprocessor is the ingestion layer. When a user uploads a 200-page deposition, a 50-page complaint, or a dump of discovery files, Preprocessor runs the seven-phase pipeline:

1. **Classify** — what kind of document is this (motion / order / deposition / contract / discovery response / pleading / exhibit)?
2. **Index authorities** — every case + statute + rule cited in the document, with pinpoints.
3. **Freeze facts** — every assertion that could be tested in cross-examination, with source pin.
4. **Extract dates** — deadlines, hearing dates, statute-of-limitations triggers.
5. **Detect parties** — every named person and entity, with their role.

6. **Map exhibits** — every exhibit reference and which document it points to.
7. **Score completeness** — what's missing, what's suspicious, what needs human attention before the tool surface is safe to use.

The output is structured JSON that downstream tools (Drafter, Auditor, Advisor) consume as silo context. The Preprocessor's job is to make the rest of the catalog smarter without the user having to do anything other than upload.

Output shape: structured JSON written to silo metadata; user sees a "ready" indicator.

2.5 — Strategist (PRO only)

Endpoint: `POST /api/strategy` **File:** `tools/strategist.js` **Tier:** PRO **Purpose:** Quick case strength analysis, high-level strategy.

Strategist is the 90-second answer. The user asks "how strong is my case?" and Strategist returns: a 0-100 strength score, the top three arguments for the user's side, the top three counter-arguments to expect, the risk band (settle / litigate / withdraw), and a confidence interval on each. Output is one screen of digest, calibrated to be the answer a senior partner gives a junior associate on the elevator.

It is intentionally not Advisor (next item). Strategist is for the quick sanity check before a status hearing; Advisor is for the deep dive before trial prep.

Output shape: structured JSON + a short prose summary; sub-2-minute response time.

2.6 — Advisor v2.0 (PRO only)

Endpoint: `POST /api/advisor` **File:** `tools/advisor.js` **Tier:** PRO **Purpose:** Deep 4-layer analysis with V4 Advocacy Analytics, Totems, and opponent-pattern detection.

Advisor is the long-form analyst. It runs four sequential cognitive layers:

- **Layer 1 — PARSE.** Pull every structured artifact from the silo: documents, dates, parties, prior-tool outputs.
- **Layer 2 — EXTRACT.** Identify the load-bearing facts, amounts, and authorities. Build the case-fact matrix.
- **Layer 3 — ANALYZE.** Run V4 Advocacy Analytics (standing / claims / asymmetries / leverage / evidence / scenarios), detect opponent patterns (30 catalogued patterns across 8 categories — see CLAUDE.md), score each pattern's likelihood and severity.
- **Layer 4 — ADVISE.** Surface the action items, the theory of the case, the next steps, the risks the user did not see coming.

Advisor maintains **Totem Memory**: hot/warm/cold rings of facts referenced in the last 7 / 8-30 / 30+ days. The ring assignment is a function of recency and importance; the ring determines retrieval cost in subsequent advisor sessions. Totems are the reason a follow-up

Advisor session on the same matter resumes coherently instead of re-discovering the same facts.

Output shape: markdown long-form analysis (~3-5 min response time); JSON action-item list; updated totem state.

2.7 — Commission Inquest (PRO only)

Endpoint: `POST /api/inquest` **File:** `tools/inquest.js` **Tier:** PRO **Purpose:** Multi-AI deliberation with flagship models for ad-hoc decisions.

Commission Inquest is the "I need a second opinion" tool. The user submits a decision (design choice, code review, strategy question, risk assessment) and Commission convenes three flagship models — Claude Opus, Gemini Pro, GPT-Pro per the current canon — under formal protocol: each model assesses independently, the results are merged under a quorum rule (standard / strict / expedited), and the user gets back a structured response: assessment (FAVORABLE / UNFAVORABLE / NUANCED), confidence (HIGH / MODERATE / LOW), key findings, recommendations ranked by priority, concerns, and reasoning.

The output is auditable: every Commission run is logged to `peer_review_audit` with the per-provider responses, the consensus, and the verdict. A user who wants to know "why did Commission tell me to settle?" can pull the full deliberation transcript.

Output shape: structured JSON with per-provider sub-responses + a synthesized verdict; logged to audit table.

2.8 — Option.Black Council (top-tier only)

Endpoint: `POST /api/tools/option-black` **File:** `routes/tools/option-black.js` **Tier:** Rainmaker / Enterprise / Admin **Purpose:** Six-model conclave deliberation via the option.black domain.

Option.Black is Commission's premium sibling. Where Commission runs three flagship models, Option.Black runs six — the full conclave. Where Commission's output is structured JSON, Option.Black's output includes the council's deliberation transcript so the principal can see the dissent, not just the verdict. The tool is gated to top-tier users because the per-call cost is materially higher and the use cases (existential strategic decisions, settlement framing on bet-the-firm matters) match the gating.

Output shape: structured JSON + a full deliberation transcript; per-model audit available.

---

3 · What every tool inherits from the kernel

The eight tools differ in surface. They are identical in three deeper layers:

Verification. Every tool's output passes through AEGIS (text containment) and AEGIS PRIME (structural facts). No tool can ship a citation that is not in the corpus or a disposition claim that disagrees with the structured fact. The user does not see the unverified version.

Silo context. Every tool reads the user's silo context (`lib/silo-context.js`). If the silo has 200 pages of deposition transcript, every tool can quote from it correctly. Linked silos extend the context graph: a discovery dispute in matter A can pull authority from sibling matter B if the two are linked.

Model tier. Every tool's model selection is governed by `server/config/model-tiers.js`. PRO surfaces use SOTA models (Opus 4.7 primary, Gemini 3 Pro + GPT-5.2 Pro peer review). Free surfaces use cost-optimized kernel models (Gemini 3 Flash primary, GPT-5.4 Mini audit). The same tool returns the same shape of answer on either tier; the difference is the underlying reasoning surface.

---

4 · How the tools chain

The catalog is composable. A typical workflow is:

1. User uploads documents → **Preprocessor** indexes silos
2. User asks a question → **Reckoner** answers with silo context
3. User says "draft the motion" → **Drafter** generates a draft
4. User says "check the draft" → **Auditor** flags issues
5. User says "is this case worth filing?" → **Strategist** scores it
6. Before trial, user asks "what am I missing?" → **Advisor** runs deep analysis
7. Strategic crossroads → **Commission** or **Option.Black** convene a panel

The user does not have to learn this sequence. Reckoner can suggest the next tool ("Want me to draft the motion now?"), and the surfaces are wired into each other so output from one is input to the next without copy-paste.

---

5 · The architectural commitment

A vendor whose architecture is "one large model behind a chat box" cannot deliver this catalog without rebuilding from scratch:

- **One-tool design** cannot specialize the output shape per task without forcing the user to phrase every task as a question to a chat box.
- **Generate-then-verify** cannot ship pre-extracted structural facts because there is no ingest phase to populate them.

- **Open-corpus retrieval** (RAG against the open internet) cannot ship AEGIS hashes because the source documents are not under the vendor's control.
- **Stateless models** cannot ship Totem memory because there is no per-user persistent context.

BenchSlap is the conjunction. Eight tools, one kernel, one corpus, one identity per user. The catalog is the surface; the seven companion papers describe the kernel under it.

---

6 · Document index — the ten-paper canon

#	Paper	Subject
00	`00-platform-overview.md`	Umbrella — surfaces, tools, subsystems
01	`01-deterministic-closed-corpus-verification.md`	AEGIS (text containment + tamper evidence)
02	`02-opinion-harvester.md`	Cascading harvester, 70 state sources
03	`03-commission-inquest.md`	Multi-AI deliberation under quorum
04	`04-chomper-evidence-ingestion.md`	Universal evidence ingestion + extract-and-destroy
05	`05-v4-advocacy-algorithm.md`	V4 truth-table + 8 parallel calculi
06	`06-citation-gravity.md`	Closed-corpus self-audit + landmark harvest
07	`07-vox-omnium.md`	Multi-model translation cascade
08	`08-aegis-prime-structural-verification.md`	AEGIS PRIME — disposition / panel / treatment
09	`09-benchslap-tool-catalog.md`	This paper — the eight tools
