VERITROOPER Audit Suite - User Manual

Scout | SitRep | Watchtower | Organize your data | Governance records
09 August 2026
Manual version 1.4

This manual explains what each tool does, when to use it, and how to run it. It assumes no prior knowledge of the product and no machine-learning background. If you read only one section, read What the suite is for.

Contents (click a section to jump)

What the suite is for

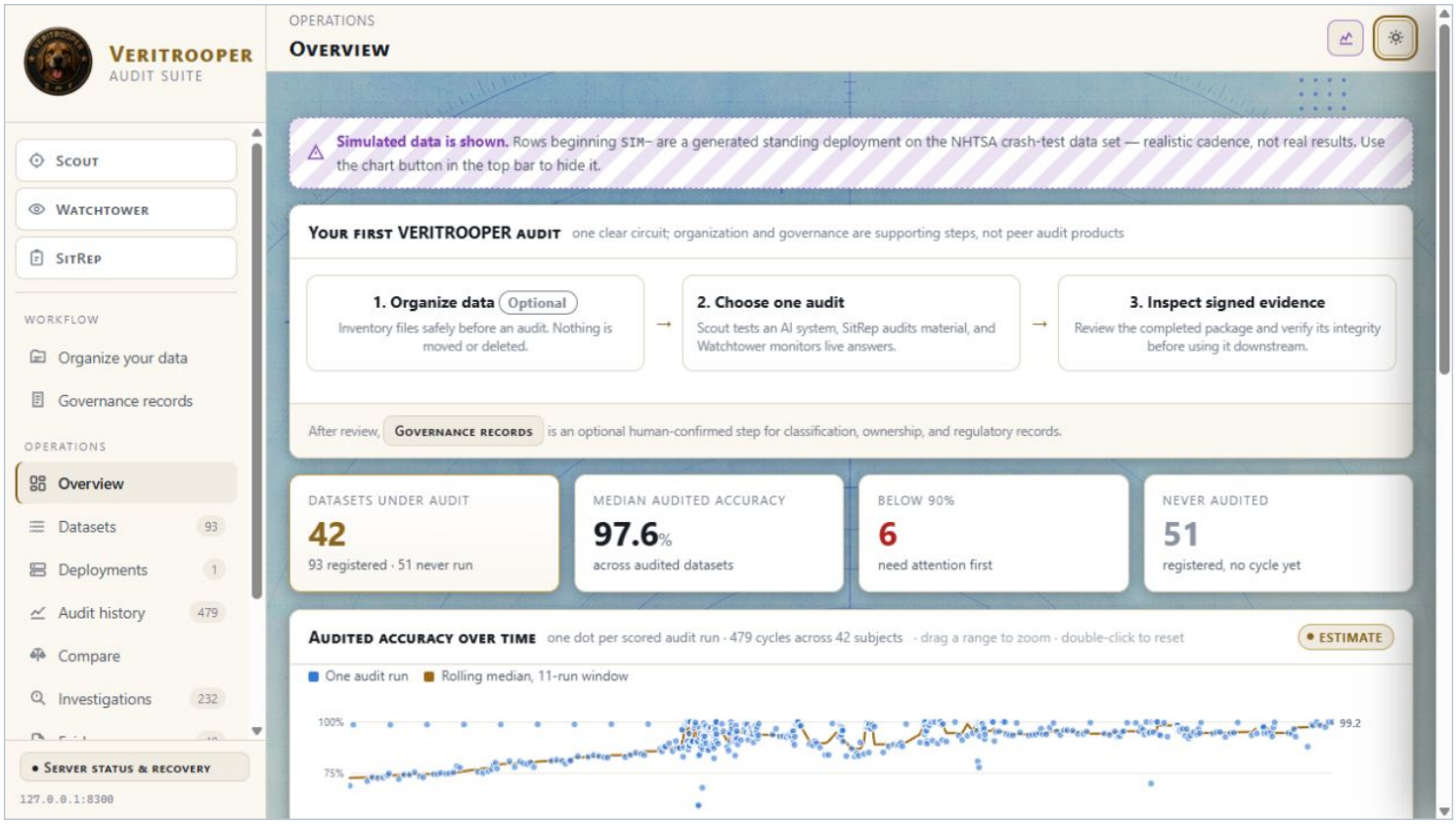
An AI system that answers questions about your documents can be wrong in two different ways, and they need two different tools.

The model can be wrong about the document. It invents a figure, misreads a table, or answers confidently from nothing. That is what **Scout** and **Watchtower** measure.

The document itself can be wrong. Two files disagree. A claim is not supported anywhere in your corpus. A question your users keep asking has no answer in the material at all. No amount of model accuracy fixes that, and that is what **SitRep** measures.

Tool	Question it answers	When you use it
Scout	**Is my AI right about my data?**	**Before you deploy. Generates a test set from your own material and grades the model against it.**
Watchtower	**Is my deployed AI right, in production?**	**After you deploy. Watches the real traffic your users generate.**
SitRep	**Is my data right?**	**Whenever the corpus is the thing you doubt. Audits the documents, not the model.**
Console	**Where is everything?**	**The front door: launches the tools, holds the shared keys, shows what is running.**

Every successfully completed, seal-eligible workflow produces a **sealed evidence package** — a report you can read, machine-readable findings you can pipe somewhere, and a signed manifest that lets anyone prove the package has not been altered since it was made.



Overview shows the operating circuit, recent audit activity, and the three audit products.

Installing and first run

Run the installer and launch the **Console**. Everything the engine needs — its Python runtime, every library, the OCR engine, and a model to think with — is bundled. There is no separate install, no package manager, and no internet connection required to install or to run.

Shared deployments and optional workstation execution. An administrator may install VERITROOPER once on a shared server and enroll selected user computers as approved local workers. Enrollment uses a short-lived, one-use code and remains pending until an administrator approves that specific worker. A normal browser link by itself does not authorize or install anything on a user's computer.

When an enrolled worker is running, it reports only the facts needed to decide whether local execution is safe: operating-system and processor description, logical processor count, total and available memory, free disk space, detected graphics-card description, and whether the VERITROOPER local runtime and model are installed. VERITROOPER does **not** use an IP address or MAC address as identity, and this check does not inspect documents, applications, browser history, or other user content. Administrators can approve, decline, or revoke the worker.

If the approved computer meets the local requirements, Scout offers **This computer** when the operator chooses where the audit will run. The audit is assigned with an expiring lease; the worker can run only a typed VERITROOPER audit, never an arbitrary remote command. When the audit needs the bundled model, the worker starts it for that audit and stops the local model server afterward to release memory and graphics resources. If no approved eligible worker is detected, the shared server remains the automatic fallback. In this release, source and output paths used for workstation execution must be available to both machines under the same path (normally an approved network share).

Installing the workstation worker and its 4.92 GB local model/runtime is a separate, administrator-controlled installation—not something the website performs when a user opens a link. The worker never downloads a model silently. If the runtime or model is absent, the hardware report says so and **This computer** is not offered; an administrator may then install the approved offline worker package under the organization's normal software-deployment and change-control process. Removing or revoking the worker returns that user to shared-server execution without changing their authorization to the Console.

You can skip the keys entirely. VERITROOPER is **air-gapped by default**: out of the box, every invoked model role runs on the model shipped inside the product, while code-only roles call no model. A computer that has never been online can install this, run a full audit, and produce a sealed evidence package. If that is how you intend to work, you are already done — go straight to whichever tool you need.

Only if you want to use a cloud model: open Keys & Tokens in the Console and add your API keys or endpoints. All three tools read the same shared configuration, so you do it once and never again. See Air-gapped operation for exactly what leaves your machine when you do.

System requirements

The installer includes the local model whether you use local or cloud connections. Cloud-only operation reduces memory use while an audit is running, but does not reduce the installed size.

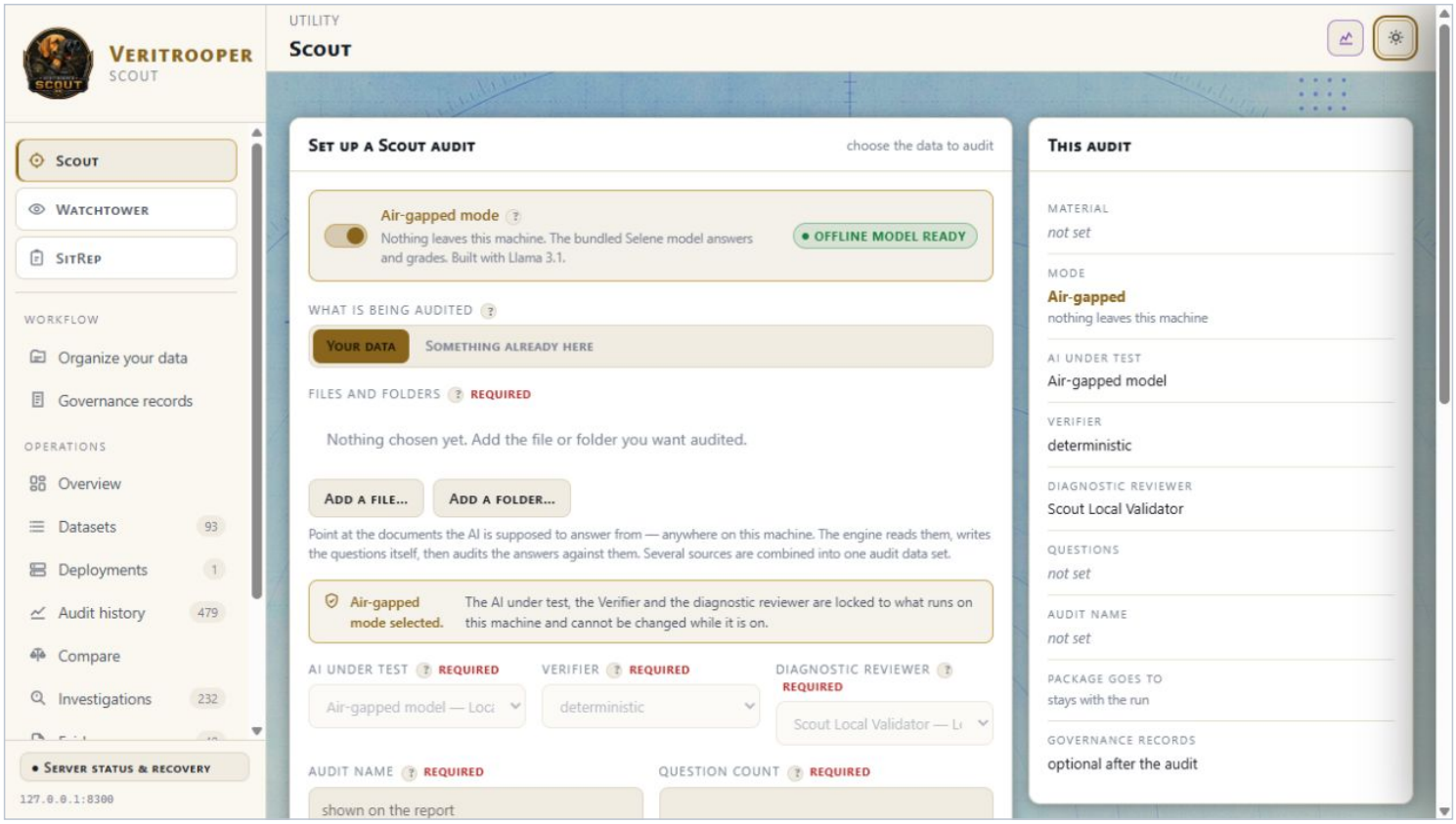
Requirement	Air-gapped — the default	All-cloud instead
Operating system	Windows 10 / 11 (64-bit)	Windows 10 / 11 (64-bit)
Processor	Any 64-bit x86 processor	Any 64-bit x86 processor
Memory	16 GB minimum; 32 GB recommended	8 GB
GPU	Optional; 8 GB VRAM recommended for faster local operation	Not required
Disk	8 GB free minimum, plus space for audit packages	8 GB free minimum, plus space for audit packages
Network	None. Ever.	Yes — to reach your model vendor

The bundled **Selene-1-Mini-Llama-3.1-8B Q4_K_M** model is 4.92 GB on disk. Acceptance testing measured about 6 GB of GPU memory with full offload and about 8.9 GB of process memory in true CPU-only operation. A 16 GB computer is a functional minimum, not generous headroom. With no graphics card an audit still completes, but substantially more slowly. Hardware changes speed, not the audit rules or output format.

Scout — auditing an AI before you deploy it

Scout generates a purpose-built test set **from your own material**, puts your model through it, and grades every answer against the source document.

It does not use a public benchmark. A model that scores well on a generic quiz can still be hopeless on your contracts, and it is your contracts you are deploying it against.



Scout setup in the light theme: choose material, identify model roles, name the audit, and set a question count.

Step by step: running an audit

1. Choose what is being audited. Select **Your data** to add files or folders from this machine, or **Something already here** to reuse a registered dataset and its existing questions.

For new material, use **Add a file** or **Add a folder**. Each selected source appears in the list and can be removed individually before deployment.

2. Name the audit. Enter the required **Audit name** shown on the report.

3. Choose where the finished package goes. Package output folder is optional. Leave it blank to keep the package with the run, or browse to an existing destination folder.

4. Set the number of Q&A pairs. In **Question count**, type how many question-and-answer pairs Scout should generate from your data — any positive whole number starting at **1**. The full Web UI does not impose a fixed upper limit.

This box starts empty and you must fill it in. If you leave it blank, Scout refuses to start. Values below 1 are invalid. Very large audits can take substantial time and provider cost, so begin with a bounded representative count and expand after review.

More questions means a longer run and a more thorough audit. A few hundred is a normal working audit; several thousand can take hours.

> **Evaluation trial:** the trial build caps this at **250** questions per audit, and the box stops you there. Nothing else is held back.

5. Choose the models. Three dropdowns:

Dropdown	What to pick
AI under test	**The AI you want audited. The list holds saved connections that can serve as an audit target.**
Diagnostic reviewer	**Air-gapped mode locks this to the bundled local model. With air-gapped mode off, choose a suitable saved connection.**
Verifier	**Air-gapped mode uses deterministic code. With air-gapped mode off, choose a verifier type or a named saved connection.**

Under the dropdowns a line tells you whether your choices are genuinely independent — for example Independent verifier [OK] - coded judge, independent of the Anthropic model under test. If you pick a verifier from the same vendor as the model under test, it warns you in red instead, because then no one is checking anyone's homework.

> **What leaves your machine.** The two graders are air-gapped by default and stay on your machine. The exception is **the AI you are auditing**: if it is a cloud model, Scout has to ask it your questions — that is the audit — so those questions go to that vendor, on your key. Point Scout at a model running on your own hardware and nothing leaves at all.

6. Record the governed-system identity. Choose a named Connection or enter the official system name, version, and intended purpose manually. These fields identify the system in the evidence; they are separate from the friendly connection name and actual model.

7. Click Deploy Scout. Required fields are marked before submission. Scout then shows a confirmation summary; verify it before starting the run.

8. Watch it run. The screen shows the stage it is in — Generating questions…, Auditing…, Diagnosing…, Verifying answers…, Writing report… — with a progress bar, a running pass/fail count, and each question and answer as it is graded.

> **Setup is not Stop.** Returning to setup or closing the browser leaves the engine job running. Use **Stop** on the run screen, or **Stop audit** from Deployments, only when you intend to interrupt it.

9. Review and sign off. Open the completed result, read the report and evidence, then choose **Sign off** when a named reviewer is ready to attest to that exact result set.

The sign-off is an explicit operator action and is hash-bound to the results. Do not sign until the reviewer has actually completed the review. An unsigned package remains audit evidence, but it does not contain a human attestation.

Use **Audit history** to reopen retained results and their output locations, or begin a new audit from Scout.

What the numbers mean

Scout reports two figures and the gap between them.

Figure	What it is
Baseline	**What your model scores answering from a plain retrieval system — the honest floor.**
Pipeline	**What it scores with the VERITROOPER pipeline in front of it. Your report labels these two rows *Without VERITROOPER (baseline)* and *With VERITROOPER (pipeline)*.**
The delta	**The measured difference between the pipeline and baseline results for this test set. Read it with the sample size, scope, and detailed findings.**

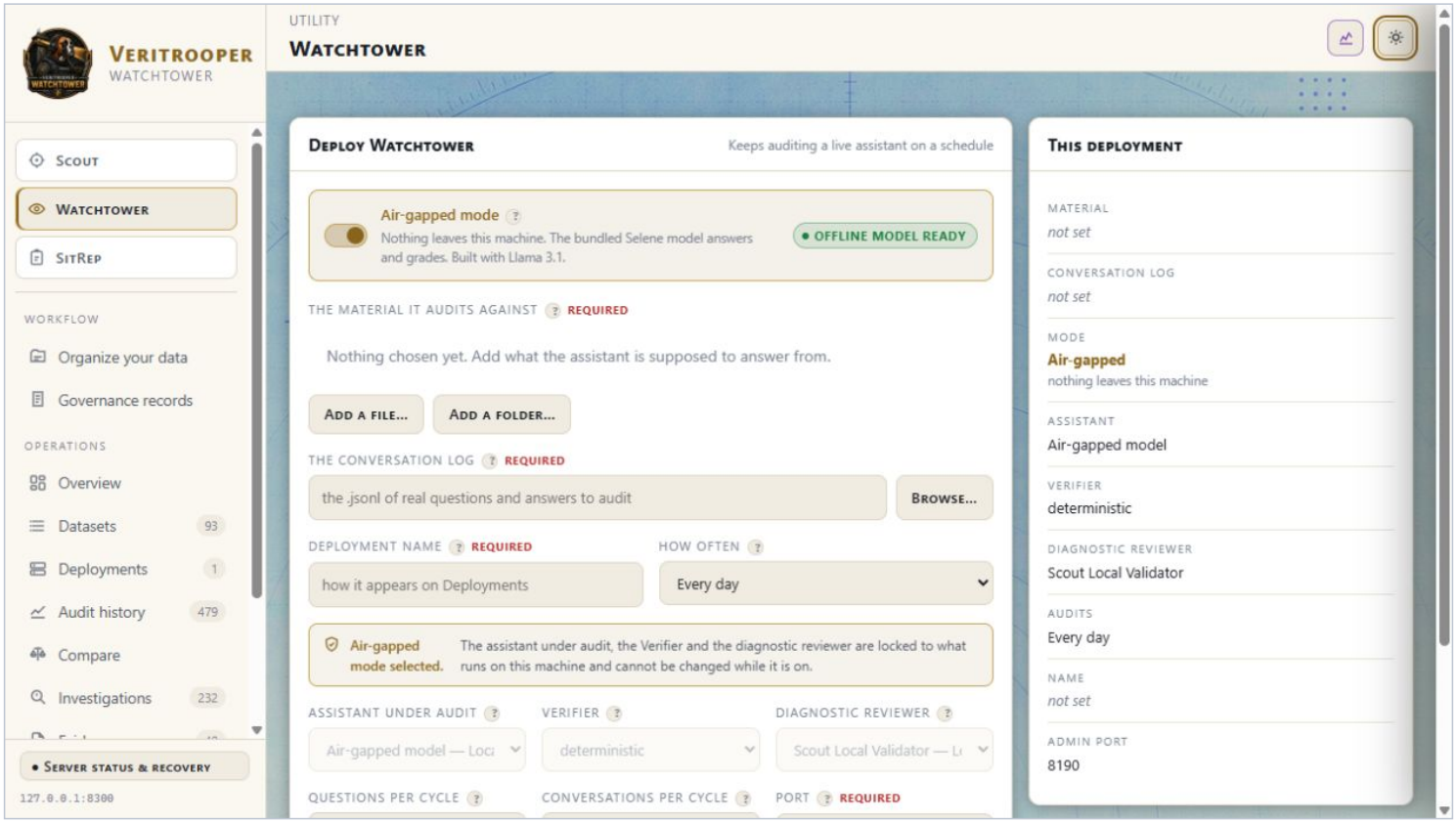
A score without its scope and findings tells you little. Use the detailed failures and referrals to decide what should be investigated or corrected.

Watchtower — monitoring an AI you have already deployed

Scout audits a model before it meets your users. Watchtower watches what happens **after**.

It observes the real traffic your deployment generates, grades those answers against your corpus, and reports what your live system is getting wrong — continuously.

A finding here is worth more than a finding in a test: it is something your system actually told a real person.



Watchtower deployment setup: source material, conversation log, cadence, and monitoring roles.

Step by step: deploying a monitor

1. Attach the material the assistant is supposed to answer from. Use **Add a file** or **Add a folder** under **The material it audits against**.

2. Choose the conversation log. This required .jsonl file contains real questions and answers handled by the assistant. The assistant integration appends new traffic to it; each cycle audits the new captured conversations.

3. Enter a Deployment name. Use the assistant or deployment identity operators will recognize on the Deployments page.

4. Choose how often it runs. How often offers Every hour, Every day (the default), Every week, Every month, and Only when I ask. The last option remains deployed but runs a cycle only when an operator chooses **Audit now**.

5. Choose the model roles. In the default air-gapped configuration Watchtower audits captured traffic passively: it does not call the deployed assistant. The diagnostic reviewer uses the bundled local model and the verifier uses deterministic code. Turn air-gapped mode off only when you deliberately want to select saved connections or enable probing.

Dropdown	Default — and what it means
Assistant under audit	**In default passive monitoring, answers come from the required conversation log and Watchtower does not call the assistant. Selecting a connection with air-gapped mode off enables active probing.**
Verifier	**Deterministic code in air-gapped mode, or a selected verifier/connection when air-gapped mode is off.**
Diagnostic reviewer	**The bundled local model in air-gapped mode, or a selected saved connection when air-gapped mode is off.**

With air-gapped mode on, Watchtower calls the bundled diagnostic model on this computer but does not call the deployed assistant or send material to a remote provider.

6. Set cycle sizes and the admin port. Questions per cycle defaults to 100; these grounded probe questions are generated once and reused for comparable cycles. **Conversations per cycle** defaults to the adaptive sampling floor shown below. Each standing deployment also needs its own available local **Port**; 8190 is the initial default.

7. Click Deploy Watchtower. Review the confirmation summary. The monitor starts and appears in the deployments list as **[ACTIVE]**.

How much does each cycle audit?

A cycle can examine captured production conversations and the stable grounded probe set. The form exposes both **Questions per cycle** and **Conversations per cycle**; leaving either blank uses the engine defaults below.

Setting	Value
Q&A pairs graded per cycle	at least 500, rising with your traffic
Grounded probe questions generated and cached (probe mode only)	100 by default

500 is a floor, not a ceiling. Each cycle grades at least 500 of the captured turns, and more as your traffic grows — **5% of everything captured**, once that is the larger number. So a deployment holding 20,000 turns has 1,000 of them graded each cycle, not 500. It never grades more than it has captured: if fewer turns exist than the floor, it grades all of them.

The deployments list shows how much traffic has been captured so far, beside the deployment name, as e.g. · 240 Q&A.

Running and stopping a cycle

Click a deployment in the list to select it, then:

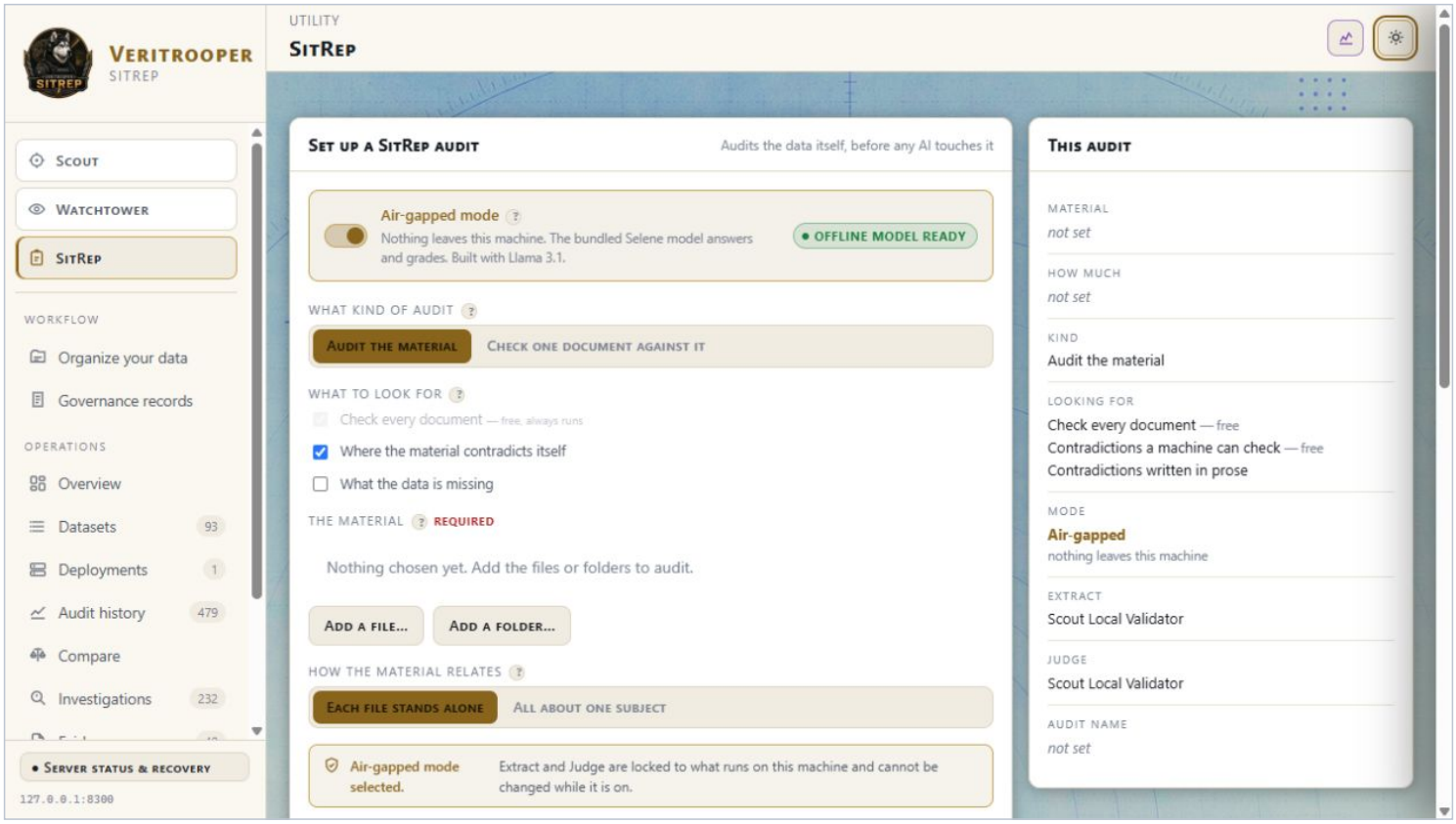
Button	What it does
Audit Now	**Runs a cycle immediately, without waiting for the schedule. Greyed out while a cycle is already running.**
View current monitoring	**Opens the deployment's current monitoring state.**
Re-index data	**Re-reads the source material used as ground truth.**
Check connections	**Tests the deployment's configured model routes.**
Stop instance	**Shuts the standing monitor down completely.**

During a cycle, **View current monitoring** shows its progress and latest evidence. After a cycle finishes, open the retained result and use **Sign off** only after a named reviewer has actually reviewed that cycle.

Closing Watchtower does **not** stop a deployment. Scheduled monitors — hourly, daily, weekly, monthly — keep auditing on their cadence with the app shut. A **manual** monitor also keeps running: it stays listening and collecting the traffic sent to it, and does not grade on its own until you ask. Every deployment is a standing service; only **Stop instance**, on that instance in Deployments, ends one. Closing the window is not a stop.

SitRep — auditing the data itself

SitRep does not score the performance of an AI under test. It audits the **documents** and may use selected models to examine them. It has four views.



SitRep setup separates authoritative source material, requested checks, model roles, and output settings.

View	What it finds	Cost
Census	**Empty files, near-duplicates, truncated documents, encoding damage, files nothing can read**	**Free. Zero model calls, 100% of the documents.**
Verify	**Claims in one document that your corpus does not support — or contradicts**	**Model calls**
Consistency	**Documents that contradict each other**	**Model calls**
Coverage	**Deterministic completeness and breadth profile of what the corpus is missing**	**No coverage-model calls**

Start with the census. It costs nothing, makes no model calls, attempts every document, and explicitly reports material it could not read. A clean census applies only to the deterministic file-health checks it performed.

Step by step: running an audit

1. Attach the material you treat as authoritative. Use **Add a file** or **Add a folder**. Each selected source appears in the list and can be removed individually.

2. Choose what kind of audit you want. There are two modes:

Button	What it does
Check one document against it	**Checks *one finished document* against the corpus, grading every claim in it Supported, Contradicted or Unsupported. Picking this reveals a second box, Document to verify, with its own Browse button — put the document to be checked there, not in the corpus box.**
Audit the material	**Audits the corpus itself.**

3. If you chose Audit the material, pick the checks. Where the material contradicts itself is selected initially. You may also select **What the data is missing**, or clear both to run only the deterministic document census.

Neither is required. With nothing selected you get the free census on its own, and the screen tells you so: Free Census only — reads every document, finds duplicates, superseded versions and unreadable files. No AI, no cost. **The census always runs and cannot be switched off**, whatever else you pick.

4. Set the options:

Control	Default	What it does
Air-gapped mode	**On**	**Locks the active model roles to the bundled local model and keeps corpus content on this machine. Turn it off to choose saved model connections.**
Audit name	**Required**	**Names the report.**
How the material relates	**Each file stands alone**	**See below — on the wrong setting, the consistency audit looks for the wrong thing.**
Extract / Oracle / Judge	**Local defaults while air-gapped**	**With air-gapped mode off, choose saved connections. Oracle appears only for the one-document check. Judge may instead run in deterministic code.**
Package output folder	**Optional**	**Leave blank to retain the package with the run, or choose an existing destination folder.**

Corpus scope is the one setting worth thinking about, because it changes what counts as a contradiction:

Choice	Use it when	What a disagreement means
Each file stands alone	**Your files are independent records — one per customer, per invoice, per patient.**	**Two files saying different things is normal. Only a document contradicting *itself* is a finding.**
All about one subject	**Your files are facets of one thing — chapters of a manual, exhibits of a filing, a set of policies.**	**Two files saying different things is a contradiction, and finding those is the whole point.**

Get this wrong on a policy manual and the audit will not look for the very thing you bought it for. The default is the cautious one: it will not accuse two documents of disagreeing when they were never talking about the same subject.

5. Decide how much material to examine. Everything is selected initially. Choose **A sample** for a bounded setup check or exploratory run; sampled results cannot clear material that was not examined.

Sample control	What it limits
Passages to read	**Maximum source passages examined in sample mode.**
Comparisons to judge	**Maximum candidate claim pairs sent through contradiction judging; leave blank to use the engine's measured/default limit.**

The form measures the passage population and states whether the selected run is complete or selective. **Everything can mean a very long run** on a large corpus. A sample of 25 or 50 passages is useful for checking a setup before committing to a complete examination.

6. Click Run audit. Missing or invalid required fields are marked and the first problem is brought into view before anything starts.

7. Check the size, then commit. SitRep reads your corpus and measures the requested scope before any model call is spent. It tells you how many documents and passages it found, what will be examined, and the planned extraction work. If this machine has run an audit before, you also get a time — worked out from **what this machine actually did**, not from a guess. If it has never been measured, it says so plainly instead of inventing a number.

Then it waits for you. **Run this audit** starts it. **Change audit values** returns to setup without starting, so you can adjust the depth or role selections.

> **This matters most on the settings you are most likely to use.** Air-gapped is the default and carries no provider charge, but it is not fast. **Everything** reads every measured passage; candidate comparisons retain their separately disclosed limit unless you set one. A large document set can hold hundreds of thousands of claims, so a complete local passage examination may take **hours or days**. Read the confirmation and the resulting scope record before treating an absence of findings as broad coverage.

8. Watch it work. The shared run screen shows ingestion, census, the stages for the selected checks, and report generation. **Stop** interrupts it. Returning to setup or closing the browser leaves the separate engine process running; reconnect from the Console or use **Stop audit** on Deployments when you intend to end it.

Reading the results

Each finding carries a badge:

Badge	Meaning
SUPPORTED	**The corpus backs this claim.**
CONTRADICTED	**The corpus says otherwise.**
UNSUPPORTED	**The corpus neither supports nor contradicts it.**
CONFLICT	**Two of your documents disagree with each other.**
GAP	**A question your corpus cannot answer.**
HIGH / MEDIUM / LOW	**Census defect severity.**
UNREAD	**A scanned page OCR refused. It was not audited at all — and the report says so rather than passing it in silence.**
REVIEW	**Something we would not decide for you: a value OCR could not read with confidence, or a contradiction the judge would not settle.**

Open the completed run from **Audit history** to view the retained result and its output location. The evidence package contains the report sections generated for the checks that actually ran; omitted checks are recorded in scope rather than represented by empty reports.

After review, choose **Sign off** and record the named reviewer's attestation. The sign-off is bound to the exact result set and causes the updated package to be rendered and re-sealed.

Signing an audit later

If the responsible reviewer is not the person who ran the audit, leave it unsigned. They can open the retained run later from **Audit history**, inspect it, and then choose **Sign off**.

An unsigned package can still carry sealed machine-generated audit evidence, but it contains no human attestation. Do not describe it as human-reviewed until that step is complete.

Scanned documents

A PDF with no text layer is a scan, and SitRep reads it automatically — there is no setting to remember. But a misread digit would be a **false accusation against your own document**, so the rules are strict:

It never guesses. A value that cannot be read with confidence is marked [?] and referred to you.

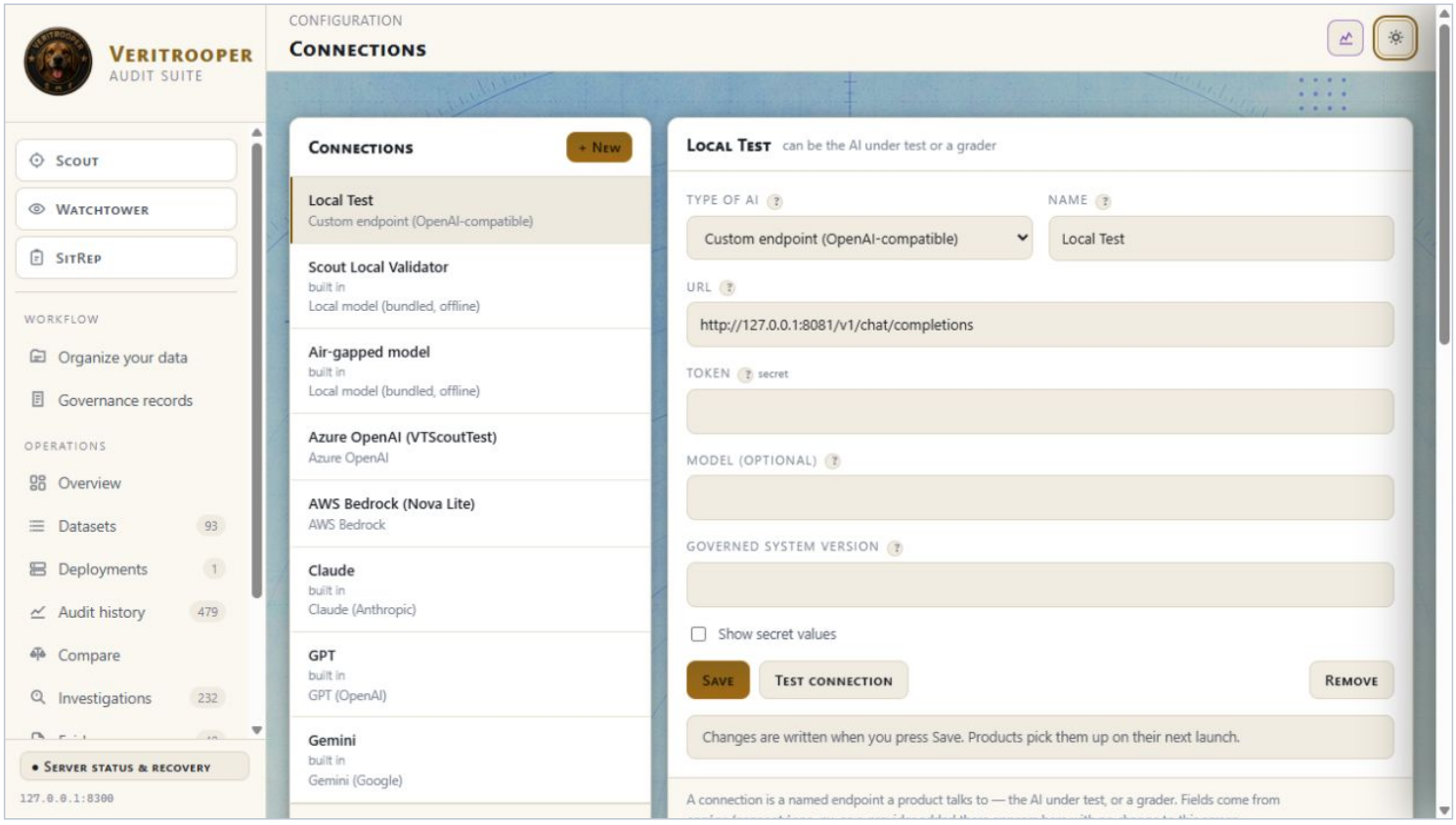
It refuses a bad page outright. A page too degraded to trust is refused whole and reported as an exception. Nothing on it is audited, and the report says so.

The page image remains the record. OCR text is derived evidence, never the source of truth.

The Console

The Console launches the tools, holds the API keys and connections all three share, and shows you every audit running on this machine — including runs whose own window has been closed, since a run outlives its window. This is where you watch a closed run, and where you stop any run early.

It opens on **Overview**. Use the navigation to open **Connections**, **Deployments**, **Scout**, **SitRep**, **Watchtower**, **Organize your data**, or **Governance records**. Unavailable audit products are identified as unavailable; the Console does not present an unlicensed product as installed.



Connections is the shared place to configure model endpoints, role defaults, and provider identity.

Adding a model connection — do this once

All three tools read the same connections, so you set them up here once and never again. If you only ever run air-gapped, you can skip this entirely.

1. Click **Connections**.
2. Click **+ New connection**.
3. From the **Type of AI** dropdown, choose your provider — **Claude (Anthropic)**, **GPT (OpenAI)**, **Gemini (Google)**, **Custom endpoint (OpenAI-compatible)** and others.
4. In **Name**, type something you will recognise later, such as Company Azure GPT-4o. This is the name that will appear in the dropdowns inside Scout, Watchtower and SitRep.
5. Fill in the fields that appear. They change to match the provider you picked: a hosted model asks for an **API key**; your own deployment asks for a **URL** and a **Token**. Keys are masked as you type — tick **Show keys** to read them back.

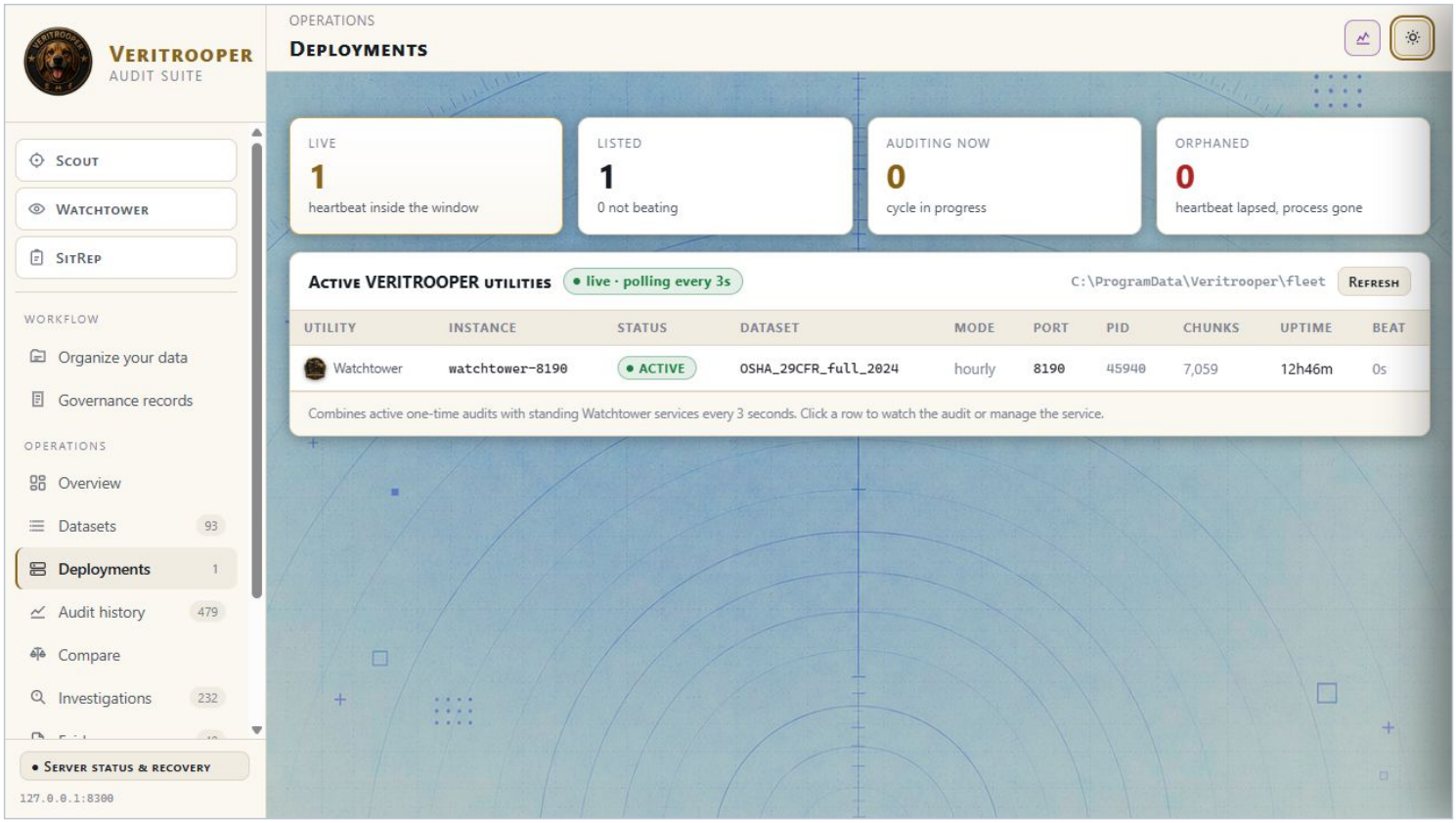
6. Click **Test connection**. You want to see **[OK]**. If you see **[FAILED]**, the message tells you why.
7. Click **Save**. (To change a connection later, select it and click **Update**, which reopens this editor; **Remove** deletes it. Both still finish with **Save**.)

Watching what is running

Deployments lists every audit running on this machine: the **product**, its **status**, the **dataset**, the **mode** it is running in, its **port**, and its **age** — where age is the time since that audit last reported in, not how long it has been running. A number that keeps climbing means the audit has gone quiet. The status word tells you where it is:

Status	Meaning
[LIVE]	**A tool is open and working.**
[ACTIVE]	**A Watchtower monitor is deployed and waiting for its next cycle.**
[INDEXING]	**Reading and indexing your data.**
[AUDITING]	**Running an audit right now.**
[PAUSED]	**Deployed, but not auditing — it will not run until you resume it.**
[DOWN]	**Stopped answering — something has gone wrong.**
[?]	**The tool reported a state this Console does not recognise.**

Select a row and click **Manage** to **Re-index** its data, run **Check Connections**, or **Stop Instance**.



Deployments shows running and standing jobs, their current state, and management actions.

Overview and navigation

The Web UI is the only customer interface. The left navigation is organized around one operating circuit: configure Connections, optionally organize source data, run one of the three audit products, review and sign the evidence, and optionally assemble a governance record. The Overview page shows this circuit and the most recent activity.

On a narrow window the navigation and long stage selectors scroll inside their own rows. The page itself should not scroll sideways. Browser zoom is supported; if a control appears clipped, return zoom to 100 percent and report the viewport and browser version to support.

Use the live-evidence panel to follow a running job. Closing the browser does not cancel the engine process. Reopen the Console to reconnect. Use **Stop** only when you intend to end the job; an interrupted package may be incomplete and is not equivalent to a sealed result.

Connections and model roles

A connection is a saved route to a model or governed AI system. Its **Name** is your friendly label; **Provider** identifies the service type; **Model** identifies the actual model. These are separate facts. Use a name that helps operators recognize the deployment, and always record the exact model or deployment version where the provider permits it.

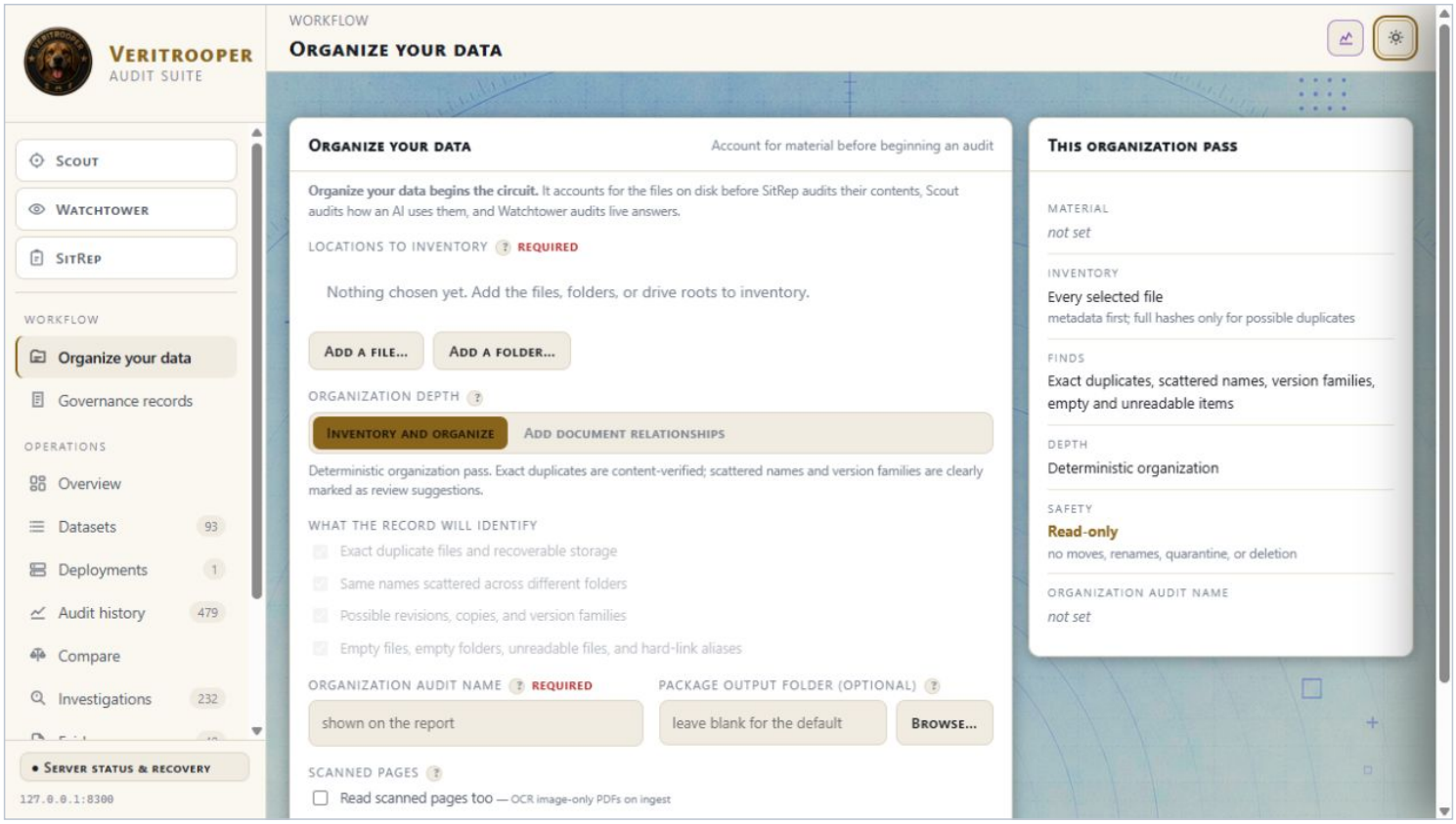
Role	What it does	Selection rule
Model under test	Produces the answers being audited	Select the exact customer system or model being assessed
Diagnostic reviewer	Examines flagged answers and identifies likely causes	Prefer a capable model distinct from the model under test
Verifier	Independently reviews contested diagnostic decisions	Prefer another vendor, or use the deterministic code verifier in air-gapped mode

To add a connection, open **Connections**, choose **New connection**, select the provider, enter a unique name and the required endpoint/model fields, save, and run **Test connection**. A successful transport test proves that the endpoint answered; it does not prove that the chosen roles are independent or suitable. Review the role summary before starting an audit.

Secrets are stored locally in access-controlled configuration files under ProgramData. Do not paste keys into audit names, notes, URLs, screenshots, or support messages. Remove a connection only after confirming that no scheduled Watchtower deployment depends on it.

Organize your data

Organize your data is a shared preparation workflow, not a fourth audit product. It inventories selected files, detects exact duplicates, groups likely revisions or related names, identifies empty and unreadable items, and produces an organization record. It never moves or deletes the source files.



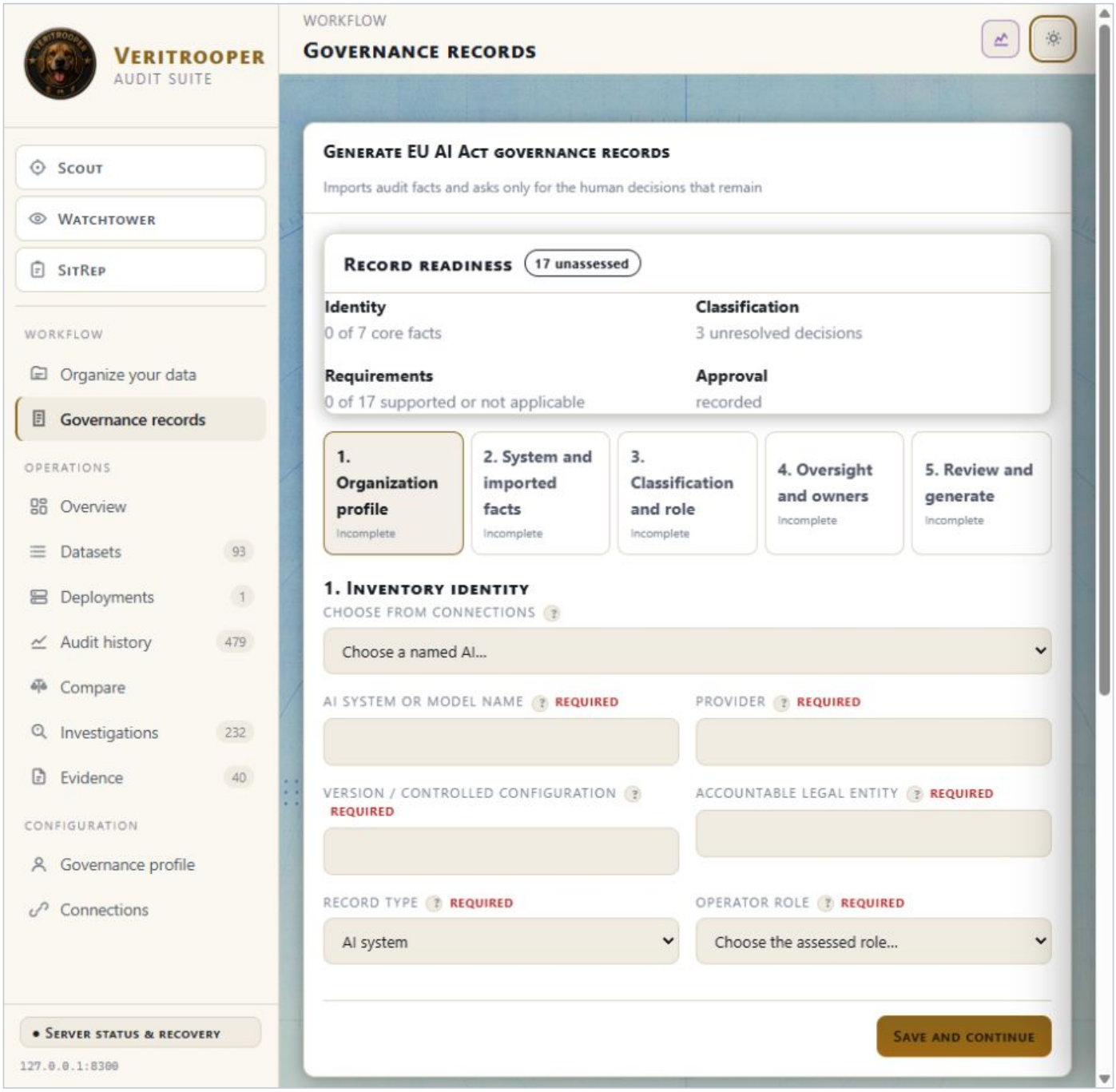
Organize your data inventories source material without moving or deleting customer files.

1. Enter a clear organization-audit name.
2. Choose one or more files or folders. Review the list before continuing.
3. Select the requested inventory/organization mode.
4. Choose an existing output folder that is separate from the source corpus.
5. Review the confirmation summary and start the job.
6. Resolve unreadable-file and revision-family referrals with the responsible data owner.

A duplicate recommendation is evidence for a human decision, not permission to delete a file. Preserve retention, legal-hold, and records-management requirements when acting on it.

Governance records

Governance records is a shared, human-accountable assembly workflow. It imports verified facts and signed evidence packages, records organization and system facts, captures classification and oversight decisions, and produces a readiness record. It does not make a legal determination, perform a conformity assessment, provide legal advice, or certify compliance.



Governance records guides the accountable reviewer through five preserved stages.

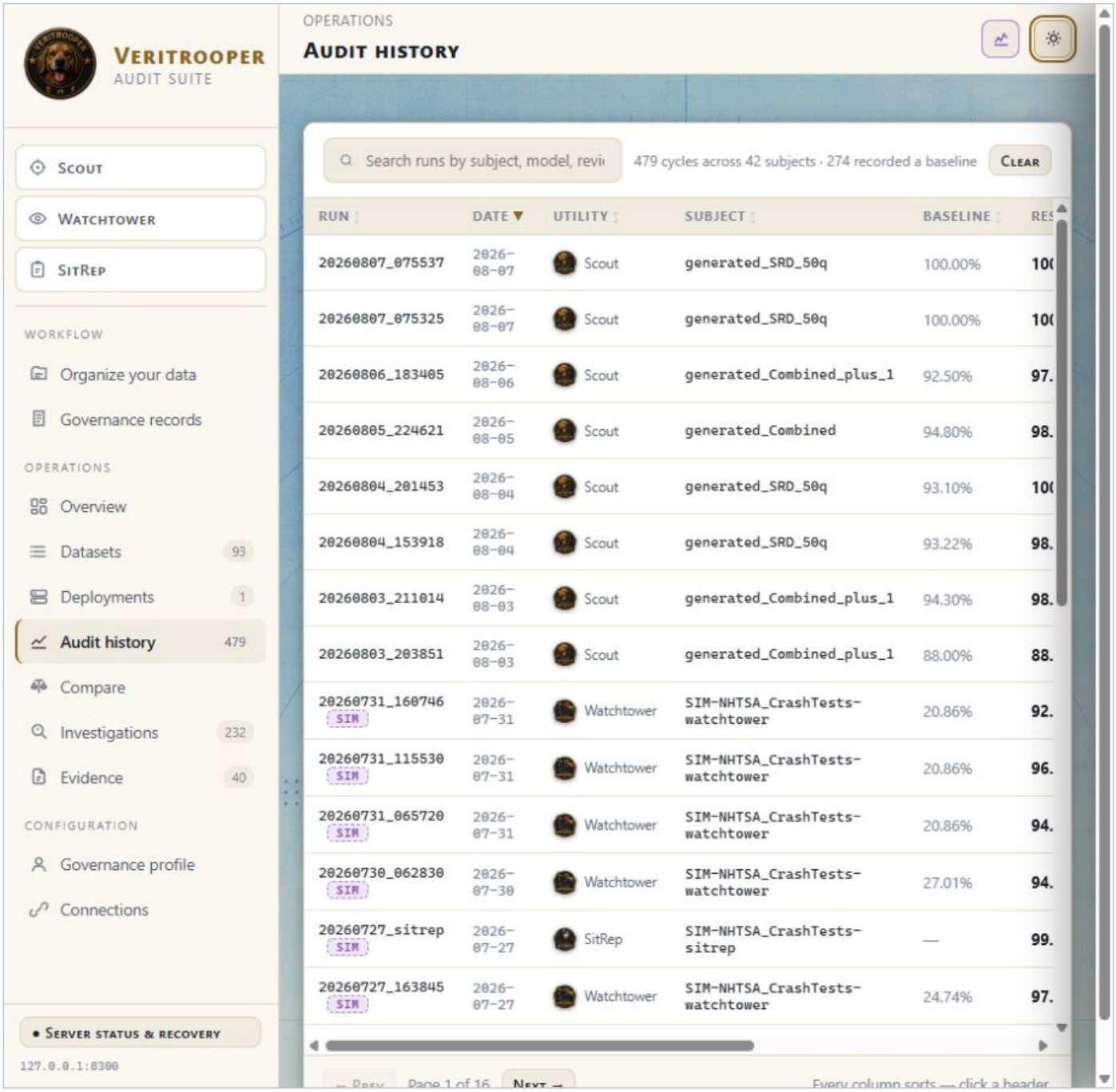
The five stages are:
1. **Organization profile** - identify the accountable legal entity and reusable organization facts.
2. **System and imported facts** - identify the governed system/version and import applicable evidence.
3. **Classification and role** - record the determination made by the accountable legal or compliance owner.
4. **Oversight and owners** - name responsible people and the human-oversight arrangement.
5. **Review and generate** - confirm completeness, attest to human decisions, choose the output folder, and build the record.

Moving backward does not discard entered values. A stage marked complete means its required fields are present; it does not mean the underlying legal or governance decision is substantively correct. Import package folders rather than individual PDFs so the workflow can verify their manifests and preserve provenance.

Managing outputs and retention

Choose a controlled client or engagement folder for outputs. Do not write a new audit into its source corpus or into another signed evidence package. Treat completed packages as records: copy the complete folder, preserve its names and structure, and verify it after transfer.

The package's System files contain the canonical machine record; the top-level report is a rendering of that record. Transparency contains detailed evidence and sign-off history. Integration contains machine-consumable exports. Integrity contains the manifest and checker. Retention and disposal remain the customer's responsibility.



Audit history provides access to completed records and their retained output locations.

Reading an evidence package

Every successfully completed audit produces the same five-drawer package. A run stopped early, or one whose required artifact checks fail, may remain incomplete and unsealed. You do not need every drawer; use the one that matches who is asking.

Drawer	What is in it	Who it is for
(top level)	The bound master report	You, and anyone you hand this to
Individual files	**Customer-readable report sections and companion documents produced by that workflow**	**Auditors and assessors who want one section, not the whole binder**
Transparency	**Detailed audit evidence retained by that workflow and the hash-bound sign-off trail (audit_trail.jsonl) when present**	**Anyone who wants to check our work rather than take it**
Integration	**evidence.json, findings.sarif.json, OpenTelemetry spans**	**Your engineering pipeline**
System files	**The machine-readable record the report was written from — the canonical results, the configuration the run used, and its log**	**Anyone reproducing or re-deriving the result**
Integrity	**The signed manifest, the signer's certificate, and the checker**	**Whoever has to prove the package is untampered**

The signed manifest covers the listed files across the package, including the canonical results in **System files**. The detached signature binds the manifest rather than only the customer-facing PDF.

What a finding actually says

A finding is never just a score. It names the claim, what the document said, what the corpus said, and where the evidence sits — so you can disagree with us.

Referrals are not failures — and not passes

Where the audit **will not accuse and cannot clear**, it says so and refers the item to a human. A referral is neither a finding nor a pass, and folding it into either would misstate the result. If you see referrals, they are work for a person, not a defect in your system.

Proving a package has not been altered

A completed seal-eligible package carries a signed manifest over its listed evidence files, plus an RFC 3161 trusted timestamp when timestamping is enabled and reachable. Air-gapped runs deliberately omit the network timestamp.

The checker travels inside the package. Anyone can run it — including you, months later, and including someone who has never installed VERITROOPER. Open the package's **Integrity** folder and double-click **Check this package.bat**.

It needs no installation, no Python and no network. To check a package from a command line, or to check a folder you were handed, run the same checker directly:

vt_verify.exe "<path to the audit folder>"

It answers that the package is **INTACT**, or it names exactly which file was changed, removed or added.

It also prints the signer fingerprint — compare it. A valid signature proves the package came from us only if that fingerprint matches the one you were given separately: in your agreement, in a signed email, or at veritrooper.com. A checker that rides inside the package is a convenience; the out-of-band fingerprint is the proof of origin.

One warning is expected and is not a failure. Air-gapped runs deliberately disable the timestamp-authority network call, so no RFC 3161 timestamp is attached. The signature and manifest can still show whether the listed evidence has changed; they do not independently establish a trusted signing time.

Air-gapped operation

Air-gapped is the default. You do not switch it on.

You have to go out of your way to send your data anywhere. Out of the box, the work VERITROOPER does happens on your machine, using a model that ships inside the product. There is nothing to configure, no key to enter, and no network call to make. If the machine has never been online, every default in this manual still works.

Air-gapped means exactly what it says: **every model role that is invoked runs on a model hosted on your machine; code-only roles call no model.** No claim, question, or passage leaves the host, and no provider key is required for that run.

What each tool does by default

Tool	Out of the box
SitRep	**Air-gapped mode is on. Active model roles use the bundled local model, and the optional code-only Judge uses no model. Nothing leaves unless you turn air-gapped mode off.**
Watchtower	**Air-gapped mode is on. It passively audits the required local conversation log, uses the bundled local diagnostic model and deterministic coded verifier, and does not call the deployed assistant or a remote provider.**
Scout	**The graders are air-gapped: the Diagnostic reviewer is the bundled offline model and the Independent verifier is a deterministic coded judge that uses no model at all. The one thing that leaves is the traffic to the AI you asked us to audit — see below.**

The one honest exception: the model under test

Scout's job is to audit an AI. If the AI you point it at is a cloud model, then Scout has to ask that model your questions — that **is** the audit. Those questions, and the passages they are drawn from, go to that vendor, under **their** terms, using **your** key.

In that Scout configuration, this traffic goes to a vendor **you** chose. If you point Scout at a model running on your own hardware, nothing leaves at all and the whole audit is air-gapped end to end.

In no configuration does anything come to us. VERITROOPER operates no service. We cannot see your data, your prompts, or your results, and we do not collect them.

It is recorded, not just promised

The evidence package records which mode the run used. A reader can tell whether an audit was air-gapped **fro m the artefact itself**, rather than from anyone's word for it.

The honest trade-off

Air-gapped model calls carry no provider charge, but free is not fast: a local model is much slower than a cloud one, and a large corpus can take hours. SitRep therefore offers **Everything** and **A sample**, with passage and comparison limits for sampled runs.

The other trade-off is depth. A cloud run can put **several vendors' models** on the same question and take the disagreement seriously; air-gapped, one local model is the sole reader of prose. Where it cannot settle a question with confidence, it refers the item to you rather than guessing — **a local model gone quiet is not a local model lying.**

Supported data

Point any tool at a file or a whole folder. It reads:

Kind	Formats
Spreadsheets	.xlsx .xlsm .xls
Tables and data	.csv .tsv .tab .json .jsonl .parquet
Documents	.pdf .docx .pptx .epub .html .htm .rtf
Databases	.db .sqlite .sqlite3 .sql .dump .dbf
Scientific arrays	.nc .nc4 .h5 .hdf5
Archives	.zip .tar .tgz .gz .bz2 .xz
Text and records	.txt .text .md .markdown .log .xml .yaml .yml .eml

Archives are expanded and their contents audited, member by member. A file the tool **cannot** read is **reported**, never silently skipped — an audit that quietly drops a document is worse than one that refuses it, because you would never know.

Troubleshooting

What you see	What it means
“Could not extract readable text ... scanned page”	The scan is too degraded even for OCR. The tool refuses rather than feed a misread digit into your audit as fact. You need a better scan, or a human reading it against the image.
“N file(s) were NOT audited”	Exactly what it says. The named files were not read, and no finding covers them. This is deliberate: silence would be worse.
“An audit is still running”	The app keeps you on this screen until the run ends — press Stop, or wait. Closing the window does not lose the run: it keeps going, and you can watch it from the Console.
“waiting for the local model — … is auditing”	Another audit holds the single local-model slot. Yours will start when it finishes.
“Sign-off failed”	Nothing was recorded and the package was not re-sealed. A failed attestation is never reported as a successful one.

What VERITROOPER does not claim

This section matters more than the rest of the manual, and it is short.

It does not confer compliance. An audit does not by itself certify or confer conformity with any law or standard, including the EU AI Act. Documents mapped to a framework are testing evidence that supports a conformity assessment. They do not replace one.

It does not prove the absence of error. No audit tool can. VERITROOPER reports what it examined and states the limits of that examination. An absence of findings is a conclusion about what was looked at — nothing more.

Its conclusions have limits. Results depend on the selected source material, scope, sampling, model behavior, deterministic rules, and any referrals left for people. A low score may be a valid result; it is not by itself evidence that the audit software failed.

Judgement stays with you. The tool supports a competent person; it does not replace one.

Glossary

Terms used in the Console, reports, and evidence packages.

Term	Plain-language meaning	See
Air-gapped	**All model processing runs on the customer's machine. No audit material is sent to a model provider.**	**Air-gapped operation**

Term	Plain-language meaning	See
Audit	A defined examination of a model, deployed assistant, or body of source material. An audit records its scope and limits.	What the suite is for
Abstention	A deliberate refusal to guess when the evidence cannot support a reliable decision.	Referrals are not failures — and not passes
Baseline	The model's result without the VERITROOPER pipeline. Scout compares this with the pipeline result.	What the numbers mean
Canonical record	The machine-readable source record from which a customer-facing report is rendered.	Managing outputs and retention
Census	SitRep's deterministic inventory and file-health examination. It runs without model calls.	SitRep — auditing the data itself
Connection	A saved route to a model or governed AI system, including its provider, endpoint, model, and friendly name.	Connections and model roles
Corpus	The documents or data treated as the authoritative source material for an audit.	SitRep — auditing the data itself
Coverage	An examination of what the corpus cannot answer or where required data is absent.	SitRep — auditing the data itself
Diagnostic reviewer	The model that examines flagged answers and identifies likely causes. It should be distinct from the model under test when possible.	Connections and model roles
Evidence package	The complete output folder containing readable reports, machine records, detailed evidence, and integrity material.	Reading an evidence package
Finding	A recorded result that states what was examined, what was observed, and where the supporting evidence is located.	What a finding actually says
Ground truth	The answer or fact derived from the designated source material and used to evaluate an answer.	Scout — auditing an AI before you deploy it
Governance records	The human-accountable workflow that records organizational facts, classification, ownership, oversight, and approval decisions.	Governance records
Human sign-off	A named person's attestation that they reviewed the results. It does not turn testing evidence into legal certification.	Signing an audit later
Independent verifier	A separate model or deterministic coded judge that reviews contested diagnostic decisions.	Connections and model roles

Term	Plain-language meaning	See
Integrity check	Verification that a sealed package still contains exactly the files covered by its signed manifest.	Proving a package has not been altered
Manifest	The inventory of files and cryptographic hashes covered by a package's seal.	Proving a package has not been altered
Model under test	The AI system whose answers are being evaluated.	Connections and model roles
Organize your data	A preparation workflow that inventories and groups material without moving or deleting source files.	Organize your data
Passive monitoring	Watchtower mode that grades traffic a deployed assistant already produced without calling or changing that assistant.	Watchtower — monitoring an AI you have already deployed
Pipeline	The model under test operating with the VERITROOPER retrieval and control layer in front of it.	What the numbers mean
Provider	The company or local service that supplies a model endpoint. The provider is separate from the configured model name.	Connections and model roles
Referral	An item sent to a person because the audit could neither support an accusation nor clear the issue reliably.	Referrals are not failures — and not passes
SARIF	A standard machine-readable findings format used by engineering and security tools.	Reading an evidence package
Seal	The signed integrity material binding a package manifest to VERITROOPER's signing identity.	Proving a package has not been altered
SUPPORTED	The source material supports the examined claim.	Reading the results
CONTRADICTED	The source material says something different from the examined claim.	Reading the results
UNSUPPORTED	The source material neither supports nor directly contradicts the examined claim.	Reading the results
UNVERIFIED	The evidence did not establish the point with enough confidence; this is an abstention, not an accusation.	Reading the results
Watchtower cycle	One scheduled or manually started examination of captured production traffic.	Running and stopping a cycle

Index

Term	Pages
output folder	5, 11, 18, 19, 26, 29
Passages to read	12, 29
passive monitoring	8, 27, 29
PDF	13, 19, 22, 24, 29
provider	5, 8, 12, 14, 16, 17, 23, 24, 25, 26, 27, 29
question count	5, 29
referral	7, 18, 22, 25, 26, 27, 29
SARIF	22, 27, 29
Scout	1, 2, 3, 4, 5, 6, 7, 14, 23, 24, 26, 29
seal	2, 3, 13, 16, 21, 22, 25, 27, 29
sign-off	6, 13, 20, 22, 25, 26, 29
SitRep	1, 2, 9, 10, 12, 13, 14, 23, 24, 26, 29
Stop instance	9, 15, 29
trial	5, 29
UNREAD	11, 13, 17, 18, 29
verify	6, 10, 11, 19, 20, 22, 29
Watchtower	1, 2, 7, 8, 9, 14, 15, 17, 23, 27, 29