T.R.U.S.T. PLATFORM

The Code of Trust — an Independent AI Governance Substrate for AI

Enterprise Console

User Manual

Transparent · Reviewed · Unalterable · Sustained · Tested

A guide for operators, compliance officers, and evaluators

Edition 1.0 · November 2026

Patent Pending · © 2026 EARTHX Corporation Ltd.

Welcome

Rogue Prevention Architecture: a Gate between proposed action and execution. Within the enforced integration, actions cannot execute without the required authorization.

The T.R.U.S.T. Platform independently evaluates proposed AI actions against codified boundaries. Connected gateways and runtime restrictions enforce the resulting authorization requirements. Blocked actions enter accountable review; immutable vetoes remain binding. The Console supplies determinations and authorization records, not external execution. Independent, tamper-evident evidence supports verification and audits. This manual introduces the architecture and each surface of the Enterprise Console.

Constitutional training and execution governance

Anthropic's Constitutional AI shapes model behavior during training. CODEX independently evaluates proposed actions at the execution boundary. The Console supplies determinations; connected gateways and runtime restrictions enforce authorization. A blocked action cannot bypass authorization through review, and immutable vetoes remain binding. Evidence documents the Gate's decisions. This distinction concerns Constitutional AI as a training method, not every runtime control offered by Anthropic's broader platform.

What T.R.U.S.T. stands for

- T — Transparent reasoning: the AI's outputs and exposed internal signals are ingested and judged, not taken on faith.
- R — Reviewed by Council: an independent multi-instance Council judges every output against codified precepts by M-of-N consensus, with a constitutional veto.
- U — Unalterable record: every determination is inscribed to an append-only, hash-chained ledger held across independent custodians no single party can edit.
- S — Sustained coherence: behavior is tracked longitudinally for drift, value-instability, and evaluator-gaming.
- T — Tested at the Gate: the Console supplies a determination and proof; external enforcement requires a connected tool gateway and runtime restrictions. There is no human override of an immutable veto.

The load-bearing boundary

Distribution secures the integrity, availability, and enforcement of the record and the rules, not access to an AI's private reasoning. Access and data security remain complementary. CODEX independently evaluates whether a proposed action may proceed. Connected gateways and runtime restrictions enforce that boundary; independent records make the determination and review verifiable.

The Council

The Council is a set of N independent evaluator instances. Every candidate output is judged by all of them, and a determination is reached by M-of-N consensus. Each evaluator returns a vote, a confidence score, its reasoning, and any precept flags it raises.

How a determination is reached

- Each evaluator votes AFFIRM or DENY against the domain's codified precepts, with reasoning and confidence.
- A determination requires M-of-N agreement (the consensus threshold shown in the Console header).
- Any immutable-tier precept violation raises a constitutional veto, which cannot be overridden by consensus.
- Capital (irreversible) actions must additionally pass the Gate of Consequence — a committed proof of consensus on the quorum.
- The composite trustworthiness score summarizes the longitudinal coherence signal alongside the vote.

Model Independence & Assurance

A Council whose seats are all the same model is not a real quorum — the votes are correlated and share one set of blind spots. When more than one model provider is configured, the substrate assigns a distinct model from a distinct provider to each seat, so a majority or a veto means agreement across genuinely independent systems. With only one provider configured, the Council collapses to the historical single-model behavior, unchanged — a zero-config pilot still works exactly as before.

What the engine guarantees

- Failover, never a silent gap: each seat carries an ordered list of backends; if its preferred model is unreachable the seat fails over to the next provider, and if every live model is down it falls back to a deterministic safety check rather than abstaining.
- Reproducible provenance: every record carries the engine version, the exact provider and model behind each seat, the consensus parameters, and cryptographic fingerprints of the request and the precepts in force — so an identical input reproduces an identical determination, and amended frameworks leave older records' fingerprints intact.
- Resistant to manipulation: the candidate output and context are delimited as material to be judged, never as commands to obey, so instructions hidden inside the text under review are ignored by the evaluators and caught by the harm detector.
- Calibrated against humans: every custody disposition is recorded as a labeled outcome, so the Console reports how often the Council agrees with human reviewers — overall, by domain, and recently — alongside an offline benchmark of precision, recall, false-veto rate, and injection resistance.
- Optional cost & latency triage: a fast single-model pre-screen can clear obvious cases before the full Council convenes; it is off by default, every determination records which path it took, and capital actions always convene the full Council.

The Distributed Ledger

Every determination is committed to a behavioral ledger held across independent custodians. Each custodian keeps its own hash-chain copy, and a record commits only when a quorum agrees. To alter a past determination, an attacker would have to recompute and re-publish every subsequent entry on a quorum of custodians simultaneously — an operation the network does not permit.

A Merkle root over each window of committed records is inscribed on-ledger. Any single record can then produce an inclusion proof — a short path that verifies the record belongs to a committed window without revealing the rest of the chain.

Using the Enterprise Console

Evaluate

Submit a candidate AI output for the Council to judge. Choose the deployment domain, paste the input context the AI is responding to and the candidate output under evaluation, optionally provide the AI's self-reported uncertainty, and flag the action as capital if it is irreversible. Convene the Council to see the verdict, the consensus tally, each evaluator's reasoning, the inscription across the validator network, and the record's ledger hash.

First use: preview and integration prerequisites

Start the guided preview from Overview to choose a framework and load an editable sample. Preview requires sign-in, evaluates with the configured Quorum, and saves to the Scratchpad without production inscription or a custody case. A dry-run determination is not permission to execute. The public Gate demonstration is a scripted educational simulation with no real evaluation.

Before connecting real tools, route applicable actions through a gateway and restrict direct tool access and alternate credentials. Check authorization for the exact action, including applicable review, expiry, policy, and evidence conditions. Fail closed when authorization cannot be verified; test blocked, expired, unavailable, and bypass paths. Return execution receipts so an authorization decision is not mistaken for a completed action.

These are planning prerequisites, not a connection wizard. The Console does not provision a gateway or runtime restrictions. Consult the blocked-action review and boundaries sections before planning deployment.

Sandbox example. The Console includes one connected sandbox tool: Plaid Sandbox transfers of up to $25.00, sent once through a trusted gateway and confirmed only from a verified Plaid receipt. It applies these prerequisites to that one tool only. It does not provision gateways or runtime restrictions for other tools, and no real money moves.

Blocked-action review: Stop. Report. Review. Authorize.

A block starts accountable review - not a workaround. Every blocked live evaluation automatically opens a custody case, including denied, vetoed, or held determinations; Scratchpad dry runs do not. Denial is final for the attempted execution but the case is reviewable. The original determination remains intact.

- Stop and report: inspect the original ledger record, rationale, requester, and review case. A case is not permission to retry the denied execution.
- Review: an eligible authenticated custodian supplies evidence and rationale for a linked Council reassessment of the original request. A Council outage or stub cannot provide live release assurance; an immutable veto cannot be overridden by human approval.
- Authorize through Council or Human Council: by default, distinct authorized humans meet a quorum of at least two. The requester cannot self-approve; dissent is recorded. An administrator may

explicitly delegate named domains to live full-Council authorization only for non-capital, non-veto cases with a known requester and no dual-control or applicable checklist policy. Otherwise human quorum is required. Authorization binds the original action and record, expires, and never executes the action.

The Console produces determinations and review authorization; it does not directly block external network or tool calls or grant access to outside resources. The outside resource owner must grant actual access. Mandatory enforcement requires a connected tool gateway and runtime restrictions, which are not delivered by the Console. Unlike instruction-only safeguards, this workflow retains the denied attempt and linked review as accountable records without mistaking instructions for enforcement.

Validator Network

Inspect the custodians that hold the ledger and the quorum required to commit a record. Where demo controls are enabled, tamper with a single validator to watch the network detect the divergence and isolate it while the quorum record stands, then re-sync to restore it.

Ledger & Commitments

Browse the behavioral ledger, verify the integrity of the chain across all custodians, and inspect the Merkle commitments. Enter a record number to produce and verify its Merkle inclusion proof.

Precepts

Review the codified normative framework for each domain. Every precept carries a tier — immutable, semi-immutable, or mutable — a plain operative statement, a regulatory or ethical anchor, and the operative instruction the Council applies. Immutable-tier violations trigger the constitutional veto.

The Console Assistant

The assistant orb, available on every page, answers questions about the Code of Trust grounded in the live model and the shipped precepts. It is informational only: it does not evaluate outputs, alter the ledger, or speak for the Council. Treat its replies as guidance and verify consequential matters against the relevant Console surface.

Compliance & ISO/IEC 42001

ISO/IEC 42001 is the international standard for AI management systems. An accredited auditor certifies your management system once a year — but a certificate is a snapshot, and the standard expects continuous operational evidence between audits. That is what the Console's Compliance surface provides: it goes beyond the certificate.

The Compliance tab maps the substrate's live operation onto twelve ISO/IEC 42001 Annex A controls, each cross-referenced to the EU AI Act and the NIST AI Risk Management Framework. Every control shows its evidence status, drawn directly from the ledger and the custody record — determinations inscribed, constitutional vetoes enforced, coherence assessments, Merkle commitments, validator consensus, capital actions gated, custody dispositions, and custodian calibration.

The Auditor Evidence Packet

A one-click, time-bounded export of the full control mapping with the underlying hash-chained records, ready to hand to your certification body. Signing in is required to download the packet; the mapping and coverage views are open to read. The Console does not certify you — your auditor does. It gives the auditor, and anyone who does not trust you, decision-level proof.

Test Results & Stress Testing

Mandatory safety baseline: all five Council seats must participate before an affirmative determination can authorize; ordinary consensus cannot fall below three of five, and capital actions require unanimity. Invalid Council or validator configuration stops startup rather than silently reducing protection. Validator custody requires at least five configured validators with a matching persisted roster; these validators are simulated in one process, not independent infrastructure. Supplied invalid coherence evidence fails closed.

Operators cannot disable capital dual control or reduce manual blocked-action quorum below two distinct eligible non-requester accounts. Preflight issuance and consumption both recheck full live Council evidence and current custody policy. Policy or roster changes invalidate outstanding permissions. Capital and other human-controlled actions remain held by the direct preflight path; a custody review authorization is not yet connected to an exact-action execution token. Ordinary unconstrained automation remains available within policy. Source/deployment owners can still change software or infrastructure: real external enforcement requires a gateway that owns the tool credentials and classifies actions independently of caller-supplied labels. The Console currently provides advisory preflight, not external execution or a verified action-classification registry.

For planned paid Sandbox or live-model stress engagements, open /console/estimator and complete the projected-usage checklist. Commercial-policy approval was recorded on 19 October 2026 for the standardized methodology, allocation rules and payment wording, preserving the USD 1,500 engagement floor and at least 20% EARTHX net margin after all included costs, not a 20% markup. Provider calls include systems, evaluations per system, each explicitly selected provider model, rounds and retries. The server calculates costs, processing on the full funding amount, taxes, service price and separate unearned reserve only from the current administrator-approved, versioned rate schedule for the mode (drafted and approved by different administrators). Missing, unlisted or expired rates produce an unavailable estimate, never a zero-price offer. Download the server-generated draft estimate JSON after calculation; editing inputs invalidates the result. Routine in-policy estimates need no repeated owner approval once a rate schedule is approved. The finalized combined checkout agreement remains unfinished and the legacy Sandbox agreement is not reconciled; customer acceptance of each exact quote and finalized terms remains required. Payments and execution remain disabled. The estimate does not collect payment, reserve capacity or unlock work.

The separate offline recovery reviewer kit exercises temporary SQLite capacity exhaustion, interrupted WAL transactions, backup restoration and fresh-process integrity checks. Run pnpm --filter @workspace/codex-console exec tsx src/stress/reviewer-cli.ts from the workspace. Default soak coverage is four separate ten-second segments; this is not continuous long-duration uptime. Outputs in generated-artifacts include JSON evidence, a readable Markdown report and a SHA-256 digest. Run the same command with --verify to check the digest and source manifest. Regenerate after source changes. These reports are locally generated engineering evidence, not independent attestation or production disaster-recovery certification.

Administrators can run the Extended fault suite from Test Results. It uses fresh temporary directories only and reports each case as passed, failed or Not tested: an OS file-size write refusal, SIGKILL at

commit boundaries, a killed and restarted stress coordinator, isolated backup restore with tampered and truncated copies, queue and admission pressure, heap, memory and wall-clock ceilings, injected provider failures, permission consumption, one continuous soak of up to 15 minutes and a read-only test-gateway check. Injected provider transport is not real provider behavior, and the file-size limit is not real volume exhaustion. The offline kit includes the same suite with --extended. Use Minimize on a failed result to replay ordered subsets of the same seeded workload and find the smallest preceding history that still fails; the original run is unchanged. Independent reviewers can run --kit and upload the generated submission file under Independent evidence. Uploads are recorded as unverified, with identity self-asserted; only a review recorded by a different administrator changes the status, and that is not certification.

After enabling desktop alerts, use Send test alert in the notification inbox to check your own computer. A successful browser request does not establish that an operating-system popup was displayed. Check notification permissions and Do Not Disturb if it does not appear.

Reports distinguish Passed, Failed, Suspended and Not Tested with reasons. A case with no verified completed assertion is Not Tested, never a pass. A suspended or incomplete run is not a successful test; a Council suspension that matches the scenario's expected safe outcome can still be a passed assertion. Download the readable offline PDF report or detailed JSON evidence after selecting a run.

Review Recommendations before starting and after completion. Begin at the recommended low-intensity baseline, change one setting at a time, and repeat comparable seeded runs before increasing pressure. Higher concurrency is not a better result. Do not overclock hardware, remove resource limits or relax safety boundaries to improve a score. Automatic conservative limits may restrict subsequent runs after resource exhaustion.

The evaluatory process reviews terminal runs automatically. Only predefined, verified operational corrections are recorded as FIXED in Admin; an unresolved software defect or model weakness remains a finding for review. A reduced load setting is not evidence that a judgment failure has been repaired. The application does not rewrite its own code, alter policy or publish changes autonomously.

Use Suggestions to send feedback without credentials or sensitive customer information. Your feedback and responses are private to you and authorized administrators. Administrators review suggestions and record their disposition and response in the administrative review area.

Enable desktop notifications with the app's permission button. Browser and operating-system permissions control delivery. An unread in-app inbox remains available when desktop permission is denied. Polling-based desktop notifications require an open app tab and connectivity; closed-browser background push is not provided by this implementation. Notification messages avoid sensitive test and feedback contents.

Open Test Results from the left navigation in either Console Preview or the signed-in Console, or visit /console/test-results. The public page explains the methodology; authentication is required to run tests and access private run history. A preview is not a completed test.

Choose regression, ramp, burst, soak or fault testing. Set a reproducible seed and bounded request count, concurrency and duration within the limits shown by the service. Start a run, monitor its status, and use Stop to cancel active work. Runs use isolated synthetic data rather than customer records or the production ledger. They do not authorize real execution or payments.

Regression scenarios assess explicit expected outcomes for legitimate and adversarial cases. Attack families include instruction manipulation, concealed intent, authorization replay and expiry, evaluator

failure, evidence integrity and tenant isolation. Read each scenario's observed outcome and coverage status: listing an attack family does not prove every attack in that family is covered. Stub-based results do not measure live-model judgment.

Ramp testing increases pressure, burst testing applies a bounded spike, soak testing applies sustained work within the permitted duration, and fault testing exercises selected interruptions and dependency failures. A short bounded soak is not evidence of long-term production reliability. Simulated dependency failures are not actual provider outages.

The load indicator reports measured pressure and resource usage. The resilience indicator reports outcomes and critical violations, not a universal robustness percentage. A correctly blocked malicious action and a protective rate-limit rejection are expected defenses, not unexpected server failures. Unnecessary blocks of legitimate requests are counted separately from unsafe approvals.

Critical requirements include authorization before execution, fail-closed behavior when evidence or decisions are unavailable, tenant isolation, replay protection, evidence integrity and bounded resource use. A critical violation fails the run regardless of the number of other passing cases. A test executor is not proof that an external production gateway enforces the same boundary.

Inspect request counts, expected rejections, unexpected failures, latency percentiles, throughput, active/queued work and measured resource usage. Unavailable measurements remain unavailable. Recovery and restart results must be interpreted against the specific fault scenario and test environment, not as production service guarantees.

Export the evidence JSON for the configuration, seed, software/policy identifiers where available, expected and observed results, diagnostic details and limitations. Replay repeats the configuration; timing and live-model responses need not be identical. Keep exports access-controlled and review them before sharing. A minimal reproducible case must preserve the failing conditions rather than merely omit inconvenient results.

The separate Live synthetic model evaluation panel is administrator-only. It sends fixed synthetic inputs to one identified provider model (OpenAI gpt-5-mini or Anthropic claude-haiku-4-5) through Replit AI Integrations without API key entry; calls consume project credits. Customer payments remain disabled and live runs never create customer charges or quotes. It is not the Gate council, tools, external executor or customer records. No paid request starts on page load, refresh, recovery, export or replay. Offline and live evidence are stored and downloaded separately.

Live prices come only from the current Estimator TEST rate schedule that one administrator drafted and a different administrator approved; there is no separate rate entry. The model must be listed with nonzero token rates, and an expired, missing or changed schedule disables dispatch. Before any run, an administrator records a separate budget authorization: a provider model, a total USD budget up to 50, an expiry within seven days and no later than the schedule, and an attestation that the approved rates are all-in upper bounds including proxy uplift and hidden/reasoning tokens. Only one authorization is active; revoke it to change models or after rates change.

Each run needs fresh consent and atomically holds its whole cost cap against the authorization. Every attempt durably reserves 4096 input tokens plus the selected output cap and the rounded-up maximum cost before dispatch. Caps: up to 20 synthetic cases, a retry cap of 0 to 2 used only when the provider throttles (honouring Retry-After, each retry fully reserved), a throughput cap of 1 to 30 requests per minute, concurrency up to two, per-request timeouts up to thirty seconds and run duration up to 120

seconds. Errors, timeouts and invalid usage are never retried. Token caps are not guaranteed spend control; the monetary bound is an application reservation that depends on the approved rates and provider caps, not provider-side billing enforcement.

At the end of a run, verified reported usage and never-dispatched budget return to the authorization; unknown usage from timeouts, stops, provider errors, throttling and restarts stays consumed permanently. Without a provider record, provider usage is shown as unavailable, never zero. An administrator may record one observed provider usage amount with a source: it is not an authenticated receipt, it never releases reservations, and usage above the dispatch reservation suspends the authorization. A response that does not verify the requested provider model, including any STUB fallback, is Not Tested and stops further spend. Stop prevents further dispatch but cannot undo charges already incurred. Restart marks active evidence interrupted and never resumes paid work. Not Tested is never a pass. Independent verification, production traffic, external execution and Gate authorization remain Not Tested.

Notes & Boundaries

- The Console is informational and operational tooling, not legal, medical, financial, or safety-critical advice.
- The ledger proves what was determined and that the rules were enforced — it does not, by itself, prove the AI's inner reasoning.
- Demo controls (validator tamper / re-sync) are disabled in production unless explicitly enabled by the operator.
- Determinations are recorded as the substrate evaluated the inputs you supplied; phrasing of the candidate output and context matters.

Transparent. Reviewed. Unalterable. Sustained. Tested.