Village Governance API — User Guide

Publishable public documentation. Wire contract only — it reveals nothing about how verdicts are reached, by design.

What this is

The Village Governance API is **governance-as-a-service**. Your system sends a proposed action — optionally with the full **kōrero**, the deliberation behind it (the discussion, the human engagement, and the AI inputs that fed it) — and gets back:

- a **verdict**: `allow`, `gate` (human judgment required), or `deny`;
- a **signed, customer-owned receipt** you can verify yourself, offline;
- an immutable **sovereign record** of the whole kōrero that you can retrieve in full and dump as a portable, self-verifiable dossier.

It is sovereign by design: records live in New Zealand or the EU, never on a US-owned provider; receipts are Ed25519-signed by your own tenant key; and the record is yours — you can export it and verify it without us.

Core idea

> The durable thing is not the feature — it's whose records are *real*. You send an action and its deliberation; you get back a decision and proof. How the decision was reached never crosses the wire.

Getting started

1. **Get a key.** You receive a tenant-scoped key `vk_...` and your base URL (e.g. `https://your-tenant.mysovereignty.digital/api/v1`). Keys are issued once and shown once.
2. **Authenticate.** Send `Authorization: Bearer vk_...` on every request.
3. **Ask for a verdict:**

```
curl -X POST "$BASE/api/v1/governance/verdicts" \
  -H "Authorization: Bearer $KEY" -H "Content-Type: application/json" \
  -d '{ "action": { "type": "report.read", "summary": "Read the Q2 report", "reversible": true
```

Response:

```
{ "decision": "allow", "reason_codes": [], "decision_id": "gd_…", "policy_epoch": "pe_…",
  "receipt": { "receipt_version": 2, "signature": "…", "signer_did": "did:web:…#…", … } }
```

Sending the full kōrero (the rich record)

To capture a real decision — *what AI was used, what it produced, how a person engaged with it in their own words, and what they decided* — send a `korero` object alongside the action:

```
{
  "action": { "type": "email.send", "summary": "Approve the newsletter", "reversible": false },
  "korero": {
    "proposal": { "title": "Approve the newsletter", "description": "…" },
    "participants": [ { "ref": "dir-04a9", "role_label": "chair" } ],
    "ai_inputs": [ { "ref": "ai-1", "tool": "Your Copilot", "model": "your-model",
                     "prompt_sha256": "…", "output_sha256": "…", "output_summary": "…" } ],
    "engagements": [ { "participant_ref": "dir-04a9", "act": "challenged", "ai_input_ref": "ai-
                       "reasons": "I checked the figures against source and corrected two." } ]
```

```
    "outcome_claim": "Approved with amendments"
  }
}
```

The engagement `reasons` are the person's **own words** — the API keeps AI output and human reasoning structurally separate, and never fills the reasons for you. That separation is the point.

Verdicts

- **`allow`** — proceed. A receipt is issued.
- **`gate`** — a human verdict is required before you execute. The response carries `gate.status: "pending"` and a `poll` URL. While gated, you can append more deliberation (below). Expiry resolves to `deny` — silence is never consent.
- **`deny`** — do not proceed.

`reason_codes` describe *your action* (e.g. `IRREVERSIBLE_ACTION`, `FINANCIAL_THRESHOLD`, `HUMAN_REVIEW_REQUIRED`), never our internal rules. The full catalogue is at `GET /governance/policy`.

Appending to a gated decision

While a decision is `pending`, add the deliberation that continues on your side:

```
curl -X POST "$BASE/api/v1/governance/verdicts/$ID/korero" \
  -H "Authorization: Bearer $KEY" -H "Content-Type: application/json" \
  -d '{ "korero": { "engagements": [ { "participant_ref": "dir-02", "act": "accepted_with_reaso
        "ai_input_ref": "ai-1", "reasons": "On reflection I accept it with the noted caveat." }
```

Each append is an append-only segment with its own receipt; the cumulative kōrero hash is re-bound. After the decision resolves or expires the record is closed (`409`).

The full record + dump

`GET /governance/verdicts/{id}/record` returns your whole record — the kōrero decrypted for you, the verdict, every receipt, and the signed proof chain. With this dossier and the DID document, anyone can verify the receipts, the proof chain, and the kōrero hash **without contacting us**. That is the portable, sovereign dossier.

Verifying a receipt offline (no call to us)

This is the differentiator — you never have to trust our word:

1. Fetch the tenant DID document: `GET https://<tenant>/.well-known/did.json`.
2. Find the `verificationMethod` whose id fragment matches the receipt's `signer_did`; take its `publicKeyMultibase`.
3. Recompute the **canonical payload**: every receipt field *except* `signature`, as canonical JSON — keys sorted, dates ISO-8601, `null`/absent omitted.
4. `Ed25519.verify(payload, hex→bytes(signature), publicKey)`.

The JS SDK ships a `verifyReceipt(receipt, didDocument)` helper that does exactly this with zero dependency on us.

Tiers, quotas, usage

`GET /governance/usage` returns your own current-period usage, tier, and quota. Sandbox stops at its quota; paid tiers meter overage. Verdict and record endpoints are rate-limited per key; `verify` and `policy` stay available even if a subscription lapses, so your evidence never goes dark.

Errors

Uniform envelope: `{ "error": { "code": "…", "message": "…" } }`. Codes: `invalid_request`, `unauthorised`, `forbidden_scope`, `tenant_mismatch`, `payload_too_large`, `rate_limited`, `decision_not_found`, `receipt_invalid_format`, `unavailable`. Every 5xx is `unavailable` — **fail-closed: hold the action and retry**, never assume allow. A `404` is identical whether a decision isn't yours or doesn't exist.

What we do not claim

The record is tamper-evident and signed. We do not market it as tamper-proof or court-defensible — whether it satisfies a legal duty is a matter for your counsel and, where it comes to it, a court. What we give you is the strongest contemporaneous, self-verifiable record of how a decision was made. Where a customer enables the write-once toggle, the record can no longer be amended at all; that hardens the record, but not the claim — even then it is evidence of how a decision was made, not proof that any legal duty is met.