BridgeBrain SDK/API Developer Guide

Version 2.5.1 - Enforceable Usage Metering + Enterprise Trust Update
Plugin target: bridgebrain-sdk-api v2.5.x with the v2.5.1 hardening patch applied
Generated June 8, 2026

This guide replaces the older v2.3.0 PDF and documents the current SDK/API contract after the enterprise trust, canonical license store, Navigator licensing-terms handoff, and enforceable usage-metering updates.

What BridgeBrain is: the identity + rights layer for AI personas, PTP containers, PLF licensing, and an SDK/API that lets apps securely load personas, verify entitlements, enforce license terms, and report usage for royalties.

What BridgeBrain is not: an LLM provider. Applications may use BridgeBrain-hosted processing or bring their own model, while still using BridgeBrain for identity, rights, proof, permissioning, metering, and usage reporting.

Architecture rule: Navigator defines the deal. SDK/API verifies, meters, and enforces the deal. PTP carries portable persona identity, licensing metadata, constraints, and governance.

What is new since v2.3.0

• RS256/JWKS enterprise trust layer. BridgeBrain now signs public license proofs with RS256 and publishes JWKS metadata for local verification by trusted applications.

• Canonical license table. License state now lives in bb_licenses instead of relying only on ad-hoc usermeta blobs. The usermeta path remains a migration/backward-compatibility bridge.

• License proof, live status, introspection, and revoke endpoints. Apps can retrieve proof, check revocation/expiration status, introspect proofs, and support whole-license revocation.

• Enforceable usage-metering ledger. The bb_license_usage table tracks usage by license_id, meter_key, and period_key.

• Usage caps and remaining usage. Status and introspection responses now include usage_limits, usage_remaining, metering_required, and overage_behavior.

• Navigator compatibility. SDK/API accepts usage limits from constraints.usage_caps, top-level usage_limits, or terms.usage_limits, then normalizes them for enforcement.

• Hardened usage authorization. POST /usage must confirm that the API user is authorized for the license being metered before accepting a usage event.

• Authoritative ledger reporting. Usage reports should read from bb_license_usage, not only legacy bb_usage_event usermeta.

Base URL

The API is served from the WordPress site where the BridgeBrain SDK/API plugin is installed:

```
https://YOUR_DOMAIN/wp-json/bb/v1
```

Authentication

Most endpoints require a user API key. Send it as a request header:

```
X-BridgeBrain-User-Key: YOUR_48_CHARACTER_KEY
```

The plugin accepts common server/proxy header variations, but client code should use X-BridgeBrain-User-Key consistently.

Public endpoints do not require this header, but apps should still treat public license status as non-sensitive and use private or authenticated flows for payment-sensitive data.

Public endpoints

- GET /marketplace/search
- GET /keys/jwks - REST compatibility alias
- GET /.well-known/jwks.json - canonical JWKS URL served from the site root
- GET /trust/metadata
- GET /licenses/{license_id}/status - public-safe status only
- POST /licenses/introspect - proof verification + public-safe status summary

Endpoint Summary

Method	Endpoint	Description	Auth
GET	/me	Authenticated user summary: user_id, user_login, inventory_count, has_api_key.	Auth
GET	/inventory	List inventory with license status and Pro capability flags.	Auth
POST	/inventory	Import a PTP persona container into inventory. Requires active entitlement.	Auth
GET	/inventory/{id}	Fetch one inventory record by persona_id/key.	Auth
POST	/inventory/sync	Sync owned bb_persona posts into the authenticated user inventory.	Auth
GET	/inventory/{id}/facets	Export live Persona Action Panel facets for an inventory item.	Auth
GET	/personas/{id}	Get persona metadata. Requires active license. Supports include expansions.	Auth
GET	/personas/{id}/constitution	Get constitution JSON, SHA-256 hash, and identity_kernel_id.	Auth
GET	/personas/{id}/memory	Fetch memory beads if BridgeBrain Core memory module is active.	Auth
POST	/personas/{id}/memory/search	Semantic memory search with optional Passport filtering.	Auth
POST	/sessions/{id}/memory/write	Write a memory bead to a session with optional Passport tag.	Auth
GET	/personas/{id}/extended-training	Fetch sanitized Pro/Extended Training if licensed.	Auth
GET	/personas/{id}/extended-training/files/{idx}	Download an extended training file if licensed.	Auth
GET	/personas/{id}/pro-training	Fetch Pro Training blob, hash, and updated_at if licensed.	Auth
POST	/licenses	Issue canonical license record and RS256 proof if trust keys are ready.	Auth
GET	/licenses/{id}/proof	Retrieve/re-issue proof for an existing canonical license.	Auth
POST	/licenses/{id}/verify?proof=JWT	Verify BridgeBrain-issued RS256 license proof.	Auth
GET	/licenses/{id}/status	Public-safe live license status plus usage remaining.	Public
POST	/licenses/introspect	Proof verification, revocation/expiration check, and usage remaining.	Public
POST	/licenses/{id}/revoke	Revoke a license. Licensor, licensee, or admin only.	Auth
POST	/usage	Record a metered usage event. Requires license_id and enforces caps.	Auth
GET	/usage/reports	Aggregated usage totals from authoritative usage ledger.	Auth
GET	/marketplace/search?q=...	Search public marketplace index.	Public
POST	/keys/rotate	Rotate authenticated user API key.	Auth
GET	/keys/jwks	REST compatibility JWKS endpoint.	Public
GET	/trust/metadata	Trust metadata, issuer, algorithm, canonical JWKS URI.	Public

Account and API Keys

GET /me (Auth)

Return basic account details for the authenticated user.

```
{
  "user_id": 123,
  "user_login": "robert",
  "inventory_count": 17,
  "has_api_key": true
}
```

Example

```
curl -H "X-BridgeBrain-User-Key: YOUR_KEY"   https://YOUR_DOMAIN/wp-json/bb/v1/me
```

POST /keys/rotate (Auth)

Rotate the authenticated user API key. Existing applications using the old key will stop working until updated.

```
curl -X POST   -H "X-BridgeBrain-User-Key: YOUR_KEY"   https://YOUR_DOMAIN/wp-json/bb/v1/keys/rotate
```

Enterprise Trust and JWKS

GET /.well-known/jwks.json (Public)

Canonical JWKS endpoint. External apps should prefer this URL when verifying BridgeBrain-issued RS256 proofs locally.

GET /keys/jwks (Public)

REST compatibility alias for the JWKS document. It remains supported, but the root /.well-known/jwks.json URL is the canonical URI shown in proof and trust metadata responses.

```
{
  "keys": [
    {
      "kty": "RSA",
      "alg": "RS256",
      "use": "sig",
      "kid": "bb-active",
      "n": "...",
      "e": "AQAB"
    }
  ]
}
```

GET /trust/metadata (Public)

Returns issuer metadata, signing algorithm, canonical JWKS URI, active key status, and verification guidance for client apps.

Inventory

GET /inventory (Auth)

List inventory items. Each item is enriched with license status and Pro capability flags.

```
{
  "123": {
    "persona_id": 123,
    "name": "My Persona",
    "version": "1.2.0",
    "license_tier": "personal",
    "license_status": "active",
    "pro_enabled": true,
    "pro_capabilities": ["extended_training", "pro_training"]
  }
}
```

POST /inventory (Auth)

Import a PTP persona container into inventory. This is license-gated: the user must have an active entitlement for the persona_id.

Field	Type	Required	Description
persona_id	string	Yes	Persona identifier in the PTP container.
name	string	Yes	Display name/title.
version	string	No	Semantic version. Default 1.0.0.
provider	string	No	LLM provider identifier.
role	string	No	Persona role label.
purpose	string	No	Persona purpose statement.
licensing	object	No	PLF licensing metadata, stored as JSON when persona_id is numeric.
governance	object	No	Governance metadata, stored as JSON when persona_id is numeric.
pro_capabilities	object/array	No	Pro capability metadata.
continuity	object	No	PTP continuity metadata; stored as packed JSON and mirrored to meta keys.

If the user does not have an active entitlement for the persona, the API returns 403 bb_license_required or 403 bb_license_inactive.

POST /inventory/sync (Auth)

Sync all bb_persona posts owned by the authenticated user into inventory. Safe for client-side resync buttons.

GET /inventory/{id}/facets (Auth)

Return live Persona Action Panel facets for an inventory item. If the facets table does not exist, an empty facets object is returned.

Persona Metadata and Runtime Data

GET /personas/{id} (Auth)

Return persona metadata. Requires a valid license for the requesting user. Optional expansions may be requested with include=extended_training,pro_training.

Query	Type	Description
include	string	Comma-separated expansions. Supported: extended_training, pro_training.

```
curl -H "X-BridgeBrain-User-Key: YOUR_KEY"   "https://YOUR_DOMAIN/wp-json/bb/v1/personas/123?include=extended_training,pro_tra
```

GET /personas/{id}/constitution (Auth)

Return the persona constitution JSON, SHA-256 hash, and identity_kernel_id. Apps should cache by hash and refetch when changed.

Persona Memory

GET /personas/{id}/memory (Auth)

Query	Type	Description
limit	int	Max beads to return. Default 50.
marker_type	string	Filter by marker type such as evolution, boundary, goal.
context_type	string	Filter by context such as sdk, chat, system.

POST /personas/{id}/memory/search (Auth)

Field	Type	Required	Description
query	string	Yes	Search text.
limit	int	No	Max results. Default 5.
threshold	float	No	Similarity threshold. Default 0.65.
memory_key	string	No	Passport key. If present, results are filtered to the matching subject tag.
bb_passport_key	string	No	Alias of memory_key.

POST /sessions/{id}/memory/write (Auth)

Field	Type	Required	Description
content	string	Yes	Bead content.
speaker	string	Yes	Who said it: user, persona, system, etc.
persona_id	int	No	Persona numeric ID to associate.
context_type	string	No	Context. Default sdk.
tags	array	No	Array of tags.
marker_type	string	No	Marker type.

Field	Type	Required	Description
source_app	string	No	App identifier. Default sdk.
memory_key	string	No	Passport key to tag bead with subject hash.
bb_passport_key	string	No	Alias of memory_key.

Extended Training and Pro Training

GET /personas/{id}/extended-training (Auth)

Return sanitized Pro/Extended Training payload only when the authenticated user has rights to access it.

```
{
  "text": "<p>Sanitized training text...</p>",
  "urls": [
    { "url": "https://example.com", "label": "Optional label", "added_at": "2026-02-12T..." }
  ],
  "files": [
    {
      "index": 0,
      "filename": "file.pdf",
      "type": "application/pdf",
      "size": 12345,
      "sha256": "abc123...",
      "download_url": "https://YOUR_DOMAIN/wp-json/bb/v1/personas/123/extended-training/files/0"
    }
  ],
  "version": "1.1",
  "updated_at": "2026-02-12T..."
}
```

GET /personas/{id}/extended-training/files/{idx} (Auth)

Download a specific extended training file. File access is rights-checked.

GET /personas/{id}/pro-training (Auth)

Return the portable Pro Training blob with SHA-256 hash and update timestamp if the license grants access.

```
{
  "enabled": true,
  "blob": "{...large training blob...}",
  "sha256": "abc123...",
  "updated_at": "2026-02-12T..."
}
```

License Issuance

POST /licenses (Auth)

Issue a canonical license record and a BridgeBrain-signed proof. In v2.4+ proof signing is RS256. If RS256 keys are not configured, the endpoint fails closed with 503 trust_not_ready after creating the canonical license record, and the proof can be retrieved later with GET /licenses/{id}/proof once trust is configured.

Field	Type	Required	Description
persona_id	string/int	Yes	Persona identifier.
tier	string	No	License tier such as personal, professional, commercial, enterprise. Default personal.
entitlements	array/object	No	Entitlements to store on the license. For Pro access, include extended_training_access and/or pro_training_access.
pro_enabled	bool	No	Whether Pro is enabled for this license. Default false.
rights	object	No	Rights metadata: purpose, commerciality, distribution, territory, approvals, attribution.
constraints	object	No	Enforcement constraints. May include usage_caps.
usage_limits	object	No	Navigator-native usage limits. SDK/API should preserve and normalize into constraints.usage_caps.
terms	object	No	Optional terms object. terms.usage_limits is accepted for Navigator compatibility.
expires_at	string	No	License expiry timestamp/date.
approval_mode	string	No	auto, approval_required, per_use, manual, etc.
payment_state	string	No	Internal payment state. Not exposed on public status surfaces.

Navigator-compatible request example

```
curl -X POST   -H "X-BridgeBrain-User-Key: YOUR_KEY"   -H "Content-Type: application/json"   -d '{
    "persona_id": 123,
    "tier": "commercial",
    "rights": {
      "purpose": "production",
      "commercial_use": true,
      "distribution": ["streaming", "marketing"],
      "approval_mode": "auto"
    },
    "usage_limits": {
      "metering_enabled": true,
      "max_total_invocations": 30,
      "max_monthly_invocations": 10,
      "max_episodes": 30,
      "overage_behavior": "deny_until_upgraded"
    }
  }'   https://YOUR_DOMAIN/wp-json/bb/v1/licenses
```

Response shape

```
{
  "ok": true,
  "license_id": "lic_...",
  "proof": "eyJhbGciOiJSUzI1NiIsImtpZCI6ImJiLWFjdGl2ZSJ9...",
  "expires_at": "2026-07-08T00:00:00Z",
  "jwks_uri": "https://YOUR_DOMAIN/.well-known/jwks.json",
  "status_check_uri": "https://YOUR_DOMAIN/wp-json/bb/v1/licenses/lic_.../status"
}
```

License Proof, Status, Introspection, and Revocation

GET /licenses/{id}/proof (Auth)

Retrieve or re-issue a signed proof for a canonical license. Access is limited to the licensor, licensee, or admin.

```
curl -H "X-BridgeBrain-User-Key: YOUR_KEY"   https://YOUR_DOMAIN/wp-json/bb/v1/licenses/lic_123/proof
```

POST /licenses/{id}/verify?proof=JWT (Auth)

Verify a BridgeBrain-issued RS256 proof and confirm it matches the license ID being verified. HS256 proofs from the old model should no longer be accepted in the enterprise trust flow.

GET /licenses/{id}/status (Public)

Return public-safe license state. This endpoint is intended for client apps to check whether a license is active, revoked, expired, and how much usage remains. Payment-sensitive state is intentionally omitted from public response surfaces.

```
{
  "license_id": "lic_123",
  "status": "active",
  "valid": true,
  "expires": "2026-07-08T00:00:00Z",
  "policy_version": "plf-2026-06",
  "rights_version": "1.0",
  "usage_limits": {
    "metering_enabled": true,
    "max_total_invocations": 30,
    "overage_behavior": "deny_until_upgraded"
  },
  "usage_remaining": {
    "invocation:total": { "used": 3, "limit": 30, "remaining": 27 }
  },
  "metering_required": true,
  "overage_behavior": "deny_until_upgraded",
  "checked_at": "2026-06-08T00:00:00Z"
}
```

POST /licenses/introspect (Public)

One-call proof verification plus live license status and usage remaining. Apps can use this when they receive a proof and need to check revocation/expiration/cap status before using a persona.

```
curl -X POST   -H "Content-Type: application/json"   -d '{"proof":"JWT_HERE"}'   https://YOUR_DOMAIN/wp-json/bb/v1/licenses/in
```

POST /licenses/{id}/revoke (Auth)

Revoke a license. The endpoint should allow only the licensor, the licensee, or an admin to revoke. Revocation updates canonical license state and the revocation store.

Usage Metering and Enforcement

POST /usage (Auth)

Record a usage event against a specific license. This is now a license-aware enforcement endpoint, not merely a usage log. The request must include license_id. The API must confirm the license exists, is active, is not revoked, is not expired, and belongs to or is accessible by the authenticated API user.

Important: If the authenticated API user is not allowed to meter against the supplied license_id, the endpoint must return 403 bb_forbidden. This prevents one API user from consuming another user's license cap.

Field	Type	Required	Description
license_id	string	Yes	Canonical PLF license ID. Required for all metered usage.
persona_id	string/int	Recommended	Persona identifier. If supplied, it must match the license persona. If omitted, the API may derive it from the license record.
app_id	string	No	Calling application identifier.
type	string	No	Metric type. Examples: invocation, output, voice_minutes, video_minutes, tokens. Default invocation.
value	int/float	No	Metric value. Default 1 for invocation-like events.
proof	string	No	Optional JWT proof. If supplied, it must verify and match license_id.
meta	object	No	Arbitrary metadata such as project_id, episode_id, output_type, seat_id, channel.

Usage example

```
curl -X POST   -H "X-BridgeBrain-User-Key: YOUR_KEY"   -H "Content-Type: application/json"   -d '{
    "license_id": "lic_123",
    "persona_id": 123,
    "app_id": "studio-command-center",
    "type": "invocation",
    "value": 1,
    "meta": {
      "project_id": "blood-moon-s1",
      "episode_id": "ep01",
      "use_context": "script-development"
    }
  }'   https://YOUR_DOMAIN/wp-json/bb/v1/usage
```

Successful response shape

```
{
  "ok": true,
  "recorded": true,
  "license_id": "lic_123",
  "persona_id": 123,
  "meter_key": "invocation",
  "usage_remaining": {
    "invocation:total": { "used": 4, "limit": 30, "remaining": 26 }
  }
}
```

Over-cap response shape

```
HTTP 402 bb_usage_cap_exceeded
{
  "code": "bb_usage_cap_exceeded",
  "message": "Usage cap exceeded for this license.",
  "data": {
    "status": 402,
    "overage_behavior": "deny_until_upgraded",
```

```
    "usage_remaining": {
      "invocation:total": { "used": 30, "limit": 30, "remaining": 0 }
    }
  }
}
```

Supported Usage Limit Fields

SDK/API should accept caps from constraints.usage_caps, usage_limits, or terms.usage_limits. It should normalize aliases so client apps can use consistent meter keys.

Field	Meaning	Typical meter key / period
metering_enabled	If false, record reporting counters but do not deny usage.	N/A
max_total_invocations	Lifetime API/persona access count.	invocation / total
max_monthly_invocations	Monthly API/persona access count.	invocation / YYYY-MM
max_outputs_total	Lifetime generated output count across supported modes.	output / total
max_episodes	Allowed number of episode units. Requires episode_id metadata for precise unique-unit enforcement.	episode / total
max_projects	Allowed number of project units. Requires project_id metadata for precise unique-unit enforcement.	project / total
max_seats	Allowed seat count. Requires seat_id/user metadata for precise unique-seat enforcement.	seat / total
max_voice_minutes_total	Lifetime voice synthesis/voice-use minutes.	voice_minutes / total
max_video_minutes_total	Lifetime video minutes.	video_minutes / total
overage_behavior	deny_until_upgraded, require_approval, or allow_but_flag.	Policy behavior

Meter aliases

The metering layer should normalize common aliases. Examples: tokens/calls/request/API call may resolve to invocation where appropriate; images/renders/generations may resolve to output; voice/voice_seconds may resolve to voice_minutes after unit conversion; video/video_seconds may resolve to video_minutes.

Unique-unit enforcement note

Project, episode, and seat caps require distinct IDs in the usage meta object. If those IDs are not supplied, the API can count events but cannot reliably enforce unique projects, episodes, or seats. The best practice is to require project_id for project-scoped licenses, episode_id for episodic usage, and seat_id or end_user_id for seat-limited usage.

Usage ledger table

The authoritative ledger is wp_bb_license_usage or {$wpdb->prefix}bb_license_usage. It stores one append-updated row per license_id + meter_key + period_key.

Column	Purpose
license_id	Canonical license being metered.
persona_id	Persona associated with the license.
user_id	Authenticated API user / licensee context.
app_id	Calling app identifier.
meter_key	Normalized meter such as invocation, output, voice_minutes.
period_key	total or period bucket such as 2026-06.

Column	Purpose
used	Current consumed amount.
limit_value	Limit applied to this meter/period.
last_event_at	Timestamp of most recent accepted event.
created_at / updated_at	Ledger row lifecycle timestamps.

Usage Reports

GET /usage/reports (Auth)

Return usage totals for licenses accessible by the authenticated user. In the v2.5.1 hardened contract, this endpoint should read from the bb_license_usage ledger, not only the legacy bb_usage_event usermeta log.

Query	Type	Description
license_id	string	Optional. Filter to one license if the authenticated user has access.
persona_id	string/int	Optional. Filter to one persona.
meter_key	string	Optional. Filter to one meter, e.g. invocation or output.
period_key	string	Optional. Filter to total or YYYY-MM bucket.
from / to	date	Optional reporting window if event-level rollup is available.

```
{
  "ok": true,
  "rows": [
    {
      "license_id": "lic_123",
      "persona_id": 123,
      "meter_key": "invocation",
      "period_key": "total",
      "used": 4,
      "limit_value": 30,
      "remaining": 26,
      "last_event_at": "2026-06-08T..."
    }
  ]
}
```

Marketplace Search

GET /marketplace/search?q=... (Public)

Search public marketplace personas/listings. This endpoint is public and should expose only listing-safe information.

```
curl "https://YOUR_DOMAIN/wp-json/bb/v1/marketplace/search?q=producer"
```

Recommended App Flow

This is the expected path for an external app, game, production tool, or agent using BridgeBrain rights infrastructure.

• Authenticate the user/app with X-BridgeBrain-User-Key.

• Fetch /inventory or /marketplace/search to locate a persona/listing.

• Obtain or verify a license. For existing licenses, retrieve proof with GET /licenses/{id}/proof or introspect a proof passed to the app.

• Before using the persona, call GET /licenses/{id}/status or POST /licenses/introspect to check active/revoked/expired status and usage_remaining.

• Fetch persona metadata/constitution/training only after entitlement checks pass.

• For each meaningful use, call POST /usage with license_id, meter type, value, and project/episode/seat metadata where applicable.

• If /usage returns bb_usage_cap_exceeded or require_approval behavior, stop generation or move the user into upgrade/approval workflow.

• Use /usage/reports or Business Manager reporting to reconcile usage, royalties, and compliance evidence.

Fail-closed rules

• Missing or invalid API key: deny authenticated operations.

• Missing license_id on /usage: deny with 400 bb_license_required.

• License revoked, inactive, expired, or canceled: deny usage and protected persona access.

• Proof supplied but invalid/mismatched: deny with bb_invalid_proof.

• Authenticated user is not licensor, licensee, or otherwise authorized for the license: deny with 403 bb_forbidden.

• Over-cap with deny_until_upgraded or require_approval: deny with 402 bb_usage_cap_exceeded.

• Over-cap with allow_but_flag: record usage but return flagged:true for downstream review.

Error Codes and Statuses

HTTP	Code	Where	Meaning
401	bb_auth	Any authenticated endpoint	Missing or invalid API key.
403	bb_forbidden	Usage, training, license proof/revoke	Authenticated user is not allowed to access this license/resource.
400	bb_license_required	POST /usage or license-gated endpoints	Required license_id missing, or user lacks entitlement/license.
403	bb_license_inactive	POST /usage, protected endpoints	License exists but is inactive.
403	bb_license_revoked	POST /usage, status/introspection	License has been revoked.
403	bb_license_expired	POST /usage, protected endpoints	License has expired.
402	bb_usage_cap_exceeded	POST /usage	Requested usage would exceed one or more license caps.
400	bb_invalid_proof	Proof verify/introspect/usage	JWT proof is invalid, expired, wrong algorithm, or mismatched to the license.
404	bb_not_found	Any endpoint	Resource not found.
400	bb_invalid_input / bb_invalid_query / bb_ptp_invalid	Various	Invalid request body, query, or PTP container.
503	trust_not_ready	POST /licenses / proof issuance	RS256 signing not configured; license record may exist but proof was not issued.
503	bb_memory_unavailable	Memory endpoints	BridgeBrain Core memory module is not active.
403	LICENSE_REQUIRED	GET /personas/{id}	No license found for this persona.
403	LICENSE_EXPIRED	GET /personas/{id}	Persona license has expired.
403	LICENSE_PAUSED	GET /personas/{id}	License paused by owner.
403	LICENSE_CANCELLED	GET /personas/{id}	License has been cancelled.

Admin, Upgrade, and Deployment Notes

• On activation, the plugin installs/updates the canonical bb_licenses table and the bb_license_usage table.

• Existing installs should also receive bb_license_usage via idempotent admin_init upgrade logic, so a deactivate/reactivate cycle should not be required.

• After upgrading from older versions, open BridgeBrain API -> Trust Dashboard and run/re-run the legacy license migration if needed.

• Configure RS256 key paths in wp-config.php and generate/promote keys from the Trust Dashboard before relying on public proof issuance.

• Use the canonical JWKS URL /.well-known/jwks.json in enterprise app documentation. Keep /bb/v1/keys/jwks as a compatibility alias.

• Public status/introspection surfaces should not expose payment_state. Payment-sensitive details belong in future authenticated/private endpoints.

• When applying the v2.5.1 hardening patch, confirm /usage enforces license ownership/access before writing usage and /usage/reports reads the ledger.

wp-config.php RS256 constants

```
define( 'BB_KM_PRIVATE_KEY_PATH_ACTIVE', '/etc/bridgebrain/keys/bb-active.pem' );
define( 'BB_KM_PRIVATE_KEY_PATH_NEXT',   '/etc/bridgebrain/keys/bb-next.pem' );
```

Appendix: Quick cURL Header Helper

```
# bash
BB_BASE="https://YOUR_DOMAIN/wp-json/bb/v1"
BB_KEY="YOUR_48_CHARACTER_KEY"

curl -H "X-BridgeBrain-User-Key: $BB_KEY" "$BB_BASE/me"

curl -H "X-BridgeBrain-User-Key: $BB_KEY" "$BB_BASE/inventory"

curl -X POST   -H "X-BridgeBrain-User-Key: $BB_KEY"   -H "Content-Type: application/json"   -d '{"license_id":"lic_123","perso
```

Appendix: Developer Acceptance Tests

• License issuance with constraints.usage_caps returns usage_limits on status/introspection.

• License issuance with top-level usage_limits preserves those limits and enforces them on /usage.

• License issuance with terms.usage_limits preserves those limits and enforces them on /usage.

• User B cannot meter against User A license_id. Expected: 403 bb_forbidden.

• Matching licensee can meter under cap. Expected: ok true and usage_remaining decremented.

• Over-cap event with deny_until_upgraded returns 402 bb_usage_cap_exceeded and does not advance counters.

• Revoked license denies /usage and status/introspection reports valid false.

• Expired license denies protected persona access and /usage.

• /usage/reports returns rows from bb_license_usage, including license_id, meter_key, period_key, used, limit_value, remaining.

• Persona_id mismatch between request and license is rejected or the API derives persona_id from license record.