

01 — Overview Integrator Tenant EMBAN AI

Audience: Tim IT di sisi tenant CS yang punya backend sendiri dan ingin mengintegrasikannya dengan agent EMBAN AI. **Bahasa:** Indonesia. **Status dokumen:** draft, aktif untuk Phase 4+.

1. Siapa Anda di doc ini

Anda adalah developer di sisi **tenant** yang punya backend sendiri (sistem booking, CRM, ticketing internal, inventory, dsb) dan ingin agent CS EMBAN AI berinteraksi dengan backend itu lewat API.

Contoh skenario:

- Tim IT "Rent-a-Car X" yang sudah subscribe plan **cs** EMBAN AI dan mau agent CS bisa booking mobil lewat API mereka sendiri.
- Tim IT toko ritel yang mau agent CS bisa cek stok produk + menerbitkan tiket eskalasi ke help-desk internal.

Tugas teknis Anda:

- **Host ticketing webhook** yang menerima notifikasi **ticket.created** dari EMBAN saat agent CS eskalasi ke human.
- **Membalas ticket** via **POST /api/v1/tickets/:id/reply**.
- **Host action endpoint** yang menerima outbound request dari EMBAN saat customer memicu custom action (booking, cek stok, dsb).
- **Mengirim callback** via **POST /api/v1/callbacks/:submission_id** untuk action mode asinkron.

Yang Anda pegang:

- 1 secret per tenant untuk **ticket webhook** (disepakati saat onboarding tenant CS).
- 1 secret per **custom action** (disepakati saat action didaftarkan).
- Akses ke endpoint **ticket reply** dan **callback**.

Yang **tidak** Anda pegang: provisioning tenant baru, lifecycle (plan change / suspend / activate / delete), dan billing/onboarding di platform — itu jalur Upstream, di luar cakupan doc ini.

2. Aktor & boundary

Selain Anda, ada tiga aktor lain yang berinteraksi dengan EMBAN. Memahami siapa-bicara-ke-siapa membantu debug saat flow lintas-aktor.

Aktor	Siapa	Bicara ke EMBAN lewat
Tenant IT (Anda)	Developer di sisi tenant CS	HTTP API callback/reply + webhook receiver, HMAC per-tenant/per-action
Tenant owner	End-user yang bayar plan	WhatsApp / Telegram / Email — bukan API
Customer (khusus role CS)	Pelanggan akhir dari tenant CS	WhatsApp / Telegram / Email, direlay lewat tenant
EMBAN engine	Pusat orchestration	menerima request dari Anda, memanggil endpoint Anda

Catatan penting:

- Tenant owner **tidak pernah** bicara ke EMBAN lewat API. Sesi tenant hidup di channel messaging.
- Customer juga tidak lewat API. Customer bicara ke channel tenant; EMBAN bertindak sebagai agent CS yang membalas (atau eskalasi via ticketing ke Anda).

3. Base URL & versioning

- **Host canonical:** <https://api.rentalai.id>. Semua endpoint di-prefix [/api/v1/...](#)
- **Mode test/sandbox:** kalau dibutuhkan untuk integrasi awal, koordinasikan via channel onboarding. Saat ini production dan test share base URL yang sama; isolasi diatur per-tenant (mis. tenant test pakai akun terpisah dengan email khusus). Subdomain dedicated belum disediakan.
- Versi API saat ini: **v1**. Breaking change memicu versi baru ([/api/v2/...](#)); v1 tetap hidup minimal 6 bulan setelah sunset diumumkan.

Jangan hardcode host di kode. Saat onboarding, Anda akan menerima [webhook_base_url](#) dari Upstream — pakai value itu, simpan di env. Alasan:

infrastruktur di belakang `api.rentalai.id` bisa berkembang (multi-region/routing), kode yang hardcode akan broken saat migration terjadi.

4. Model autentikasi (ringkasan)

Anda akan memegang **dua jenis secret** (jumlah bisa lebih banyak kalau Anda register multiple custom actions):

Secret	Skup	Dipakai di
<code>ticket_webhook_secret</code> (per-tenant)	tenant-scoped, 1 per tenant	outbound <code>ticket.created</code> webhook + inbound <code>/api/v1/tickets/:id/reply</code>
<code>action.secret</code> (per-action, per-tenant)	action-scoped, N per tenant	outbound action endpoint + inbound <code>/api/v1/callbacks/:sub_mission_id</code>

Semua autentikasi memakai pola **HMAC-SHA256**. Ada **dua skema** yang berbeda:

1. **Ts-prefixed scheme** (default — dipakai oleh ticket webhook + callback + reply). Window 5 menit.
2. **Body-only scheme** (Action Engine outbound saja — engine kontrol retry sendiri, no replay risk).

Detail skema + contoh sign lintas bahasa ada di **#2 Authentication**.

5. Standard response envelope

Semua response JSON mengikuti skema berikut.

Sukses:

```
{  
  
  "ok": true,
```

```
"tenant_id": "01J...",  
"...": "field-domain-specific"  
}
```

Error:

```
{  
  
  "ok": false,  
  
  "error_code": "schema_invalid",  
  
  "message": "Human-readable reason (EN)",  
  
  "request_id": "req_01J...",  
  
  "details": { "...": "opsional, berisi breakdown validasi" }  
}
```

`error_code` adalah string stabil yang aman di-switch di kode Anda. `message` boleh berubah. `request_id` selalu ada — sertakan di ticket support kalau ada masalah.

Daftar lengkap `error_code` di **#8 Error codes**.

6. Rate limit & idempotency (ringkasan)

- **Ticket reply + callback** memakai kuota standar (default 60 req/menit, `API_RATE_LIMIT_STANDARD`).
- Semua endpoint *write* menerima header `Idempotency-Key` (UUID/ULID pilihan caller). Permintaan ulang dengan key yang sama dalam 24 jam mengembalikan response cached, tidak membuat resource duplikat.
- Retry dari sisi caller: **eksponensial** (2s, 4s, 8s, 16s) maksimal 4 kali, lalu gagal permanen.

Detail di **#8 Error codes** (§4 Rate limits + §5 Idempotency).

7. Hello world — cek health

Endpoint `GET /api/v1/health` adalah **public**, tanpa HMAC. Pakai untuk smoke-test koneksi sebelum Anda mulai sign request.

curl

```
curl -sS https://api.rentalai.id/api/v1/health
```

Node.js (fetch)

```
const res = await fetch("https://api.rentalai.id/api/v1/health");

const body = await res.json();

console.log(body); // { ok: true, status: "healthy", version: "1.0.0", ... }
```

Python (requests)

```
import requests

resp = requests.get("https://api.rentalai.id/api/v1/health", timeout=5)

resp.raise_for_status()

print(resp.json()) # {'ok': True, 'status': 'healthy', ...}
```

Kalau 3 bahasa di atas jalan, lanjut ke #2 untuk skema HMAC.

8. Urutan membaca dokumen

Saran urutan: **#1** → **#2** → **#5** → **#12 (untuk read-only integrasi)** → **#3** → **#6** → **#4** → **#7 (skim)** → **#8** → **#9** → **#11**.

Daftar lengkap:

#	File	Ringkasan
1	<code>01-overview.md</code>	File ini.

#	File	Ringkasan
2	02-authentication.md	Skema HMAC (ts-prefixed + body-only) + contoh sign lintas bahasa.
3	03-callbacks-api.md	Callback async action (Anda → EMBAN).
4	04-ticketing-api.md	ticket.created webhook (EMBAN → Anda) + endpoint reply (Anda → EMBAN).
5	05-action-engine.md	Outbound request EMBAN → endpoint action Anda.
6	06-file-delivery.md	Tenant → customer file delivery via files[] di callback.
7	07-working-hours.md	Awareness — apa yang terjadi saat customer masuk di luar jam kerja.
8	08-error-codes.md	Konsolidasi error_code + rate limit + idempotency detail.
9	09-examples.md	Flow end-to-end yang merangkai endpoint-endpoint.
10	10-glossary.md	Istilah: tenant, agent, role, plan, channel, customer, ticket, action, submission, dsb.
11	11-reference-implementation.md	Cross-link ke kode receiver nyata di repo emban-ai (ticket receiver, admin tickets list, product query stub).
12	12-template-library.md	Template library — 3 action template generic (business_metric , business_list ,

#	File	Ringkasan
		<code>business_search</code>) untuk read-only query data backend. Satu endpoint per template, internal dispatch berdasarkan field. Cocok kalau owner butuh expose metric/list/search dari ERP/CRM/POS tenant tanpa custom-define action per-query.

9. Kontak & dukungan

- Bug report & pertanyaan teknis: channel komunikasi yang disepakati pada onboarding (biasanya Slack shared channel atau email ke tim EMBAN).
- Security issue (suspected leak HMAC secret, anomaly): segera rotate via prosedur di **#2 §10**, lalu escalate.
- SLA, jadwal maintenance, dan runbook insiden: di luar cakupan doc publik — kontrak terpisah.