

12 — Template Library (Business-Data Trio)

Audience: Tenant IT. **Prasyarat:** [01-overview.md](#), [02-authentication.md](#), [05-action-engine.md](#) (§2 action JSON skeleton + §3 outbound request signing).

Doc ini adalah **library** dari 3 generic action template yang sudah disediakan EMBAN engine. Owner cukup register action JSON yang sudah disediakan EMBAN ke tenant mereka (lewat [register_tenant_action](#) di percakapan dengan agent CS/SA), lalu Tenant IT implement satu endpoint backend per template — dispatch internal berdasarkan field [metric_name](#) / [resource](#) / [query](#). Tidak perlu custom-define action per-query.

Kenapa template library? Praktek lapangan: tenant register 1 custom action per query ([metric_orders_today](#), [metric_revenue_mtd](#), ...) skalanya jelek. Lebih bersih: 1 endpoint backend menerima dispatch key, internal switch ke handler per-metric. Template library trio ([business_metric](#), [business_list](#), [business_search](#)) cover 80% read-only query use case.

Daftar isi

1. [Kapan pakai template library](#)
2. [Pola umum \(HMAC, structured mode, register flow\)](#)
3. [Template 1 — business_metric](#)
4. [Template 2 — business_list](#)
5. [Template 3 — business_search](#)
6. [Error code mapping](#)
7. [FAQ + anti-patterns](#)

1. Kapan pakai template library

Use case	Template	Contoh consumer call
Single-number metric (count/sum/avg/persen/durasi)	business_metric	"Berapa order hari ini?" → query_business_metric (metric_name ='orders_to day')

Use case	Template	Contoh consumer call
Paginated list dari resource	<code>business_list</code>	"List 20 client terbaru" → <code>query_business_list(resource='clients')</code>
Free-text search di resource	<code>business_search</code>	"Cari produk kemeja batik" → <code>query_business_search(resource='products', query='kemeja batik')</code>

Jangan pakai template library untuk:

- Side-effect write (create/update/delete) — pakai custom action dengan `response.mode='sync'` atau `'async'`
- Multi-step workflow yang butuh field collection conversational — pakai custom action
- Sensitive data yang tidak boleh masuk hint_llm — pakai custom action dengan response template yang strip sensitive fields

2. Pola umum

2.1 Structured response mode

Ketiga template memakai `response.mode: "structured"` di action JSON. Konsekuensi:

- EMBAN engine TIDAK render `success_template` — consumer tool (EMBAN internal) compose user-facing message dari JSON response mentah
- Backend tenant WAJIB return JSON yang strict match contract per template (lihat per-section di bawah)
- Field yang tidak di-spec di contract akan diabaikan oleh consumer tool — boleh ada di response untuk debugging, tapi jangan andalkan untuk muncul ke owner

2.2 HMAC signing

Sama dengan `05-action-engine.md §3`:

X-ASPRI-Signature: sha256=<hex(HMAC(secret, body))>

Secret di-store sebagai env var (atau lewat EMBAN claim-token flow ke DB ciphertext — lihat `05-action-engine.md §6`). Convention: 1 secret per template

(ASPRI_BUSINESS_METRIC_SECRET, ASPRI_BUSINESS_LIST_SECRET, ASPRI_BUSINESS_SEARCH_SECRET).

2.3 Register flow (owner-side)

Owner panggil `register_tenant_action` di chat dengan agent CS/SA, paste JSON template:

`register_tenant_action(definition_json = <JSON dari §3/§4/§5>)`

Engine validate schema → simpan ke DB → return success. Tool tersedia langsung untuk LLM consumer (`query_business_metric` / `query_business_list` / `query_business_search`).

2.4 Authentication context

Endpoint backend dapat header tambahan dari EMBAN:

- `X-ASPRI-Tenant-Id` — tenant ID
- `X-ASPRI-Submission-Id` — ULID submission, idempotency key
- `X-ASPRI-Schema-Version: 1.0`

Body shape:

```
{  
  
  "submission_id": "01J1ZXK7ABCDE",  
  
  "action": "business_metric",  
  
  "version": 1,  
  
  "data": { ... fields per template ... },  
  
  "context": {  
  
    "tenant_id": "...",  
  
    "channel": "owner-internal"  
  
  }  
}
```

Backend ekstrak `data.<field>` untuk dispatch.

3. Template 1 — `business_metric`

Single-number query. Backend dispatch berdasarkan `metric_name` string ke per-metric handler.

3.1 Action JSON (paste ke `register_tenant_action`)

```
{  
  
  "name": "business_metric",  
  
  "description": "Generic count/sum/avg/percentage/duration metric query.",  
  
  "schema_version": "1.0",  
  
  "fields": {  
  
    "metric_name": {  
  
      "type": "string",  
  
      "required": true,  
  
      "ask": { "id": "Metric apa yang mau dilihat?" },  
  
      "validation": { "pattern": "[a-z][a-z0-9_]{0,62}$" }  
  
    },  
  
    "filter_period": {  
  
      "type": "string",  
  
      "required": false,  
  
      "validation": { "max_length": 32 }  
  
    }  
  
  }  
}
```

```
},  
  
"endpoint": {  
  
  "method": "POST",  
  
  "url": "https://backend.tenant.example/aspri/business-metric",  
  
  "auth": {  
  
    "type": "hmac_sha256",  
  
    "secret_env": "ASPRI_BUSINESS_METRIC_SECRET"  
  
  },  
  
  "timeout_ms": 8000,  
  
  "retry": { "max": 2, "backoff": "exponential" }  
  
},  
  
"response": {  
  
  "mode": "structured"  
  
},  
  
"rate_limit": {  
  
  "global_per_minute": 60  
  
}  
  
}
```

3.2 Backend response contract

Backend WAJIB return:

```
{  
  
  "metric_name": "orders_today",
```

```
"value": 42,  
  
"value_type": "count",  
  
"label": "Total order hari ini",  
  
"filter_period": "today",  
  
"as_of": "2026-05-13T10:00:00+07:00"  
  
}
```

Field	Type	Required	Catatan
<code>metric_name</code>	string	optional	Echo back <code>data.metric_name</code> . Kalau missing, consumer pakai value yang dikirim
<code>value</code>	number	wajib	Finite number. Integer untuk count, decimal untuk currency/percentage/duration
<code>value_type</code>	enum	wajib	<code>"count"</code> <code>"currency"</code> <code>"percentage"</code> <code>"duration_seconds"</code> . Tool format berbeda per tipe
<code>label</code>	string	wajib	Non-empty. Dipakai consumer untuk human-readable framing ("Total order hari ini")
<code>filter_period</code>	string	optional	Echo back; kalau missing, consumer pakai input

Field	Type	Required	Catatan
as_of	ISO 8601	optional	Timestamp data backend. Kalau ada, di-include di hint_ilm

value_type semantics:

- **count** — integer, formatted dengan locale separator (id-ID: "1.234")
- **currency** — IDR, formatted **Rp 12.500.000**
- **percentage** — 0-100 ratio (bukan 0-1), formatted **12.50%**
- **duration_seconds** — detik, formatted otomatis ke detik/menit/jam/hari

3.3 Reference implementation (Node.js / TypeScript)

```
// src/api/routes/aspri-business-metric.ts
```

```
import type { FastifyInstance } from "fastify";
```

```
import { createHmac, timingSafeEqual } from "node:crypto";
```

```
interface MetricRequestBody {
```

```
  submission_id: string;
```

```
  action: string;
```

```
  version: number;
```

```
  data: {
```

```
    metric_name: string;
```

```
    filter_period?: string;
```

```
  };
```

```
  context: {
```

```
    tenant_id: string;
```

```
    channel: string;
```

```

};

}

interface MetricResponse {

    metric_name: string;

    value: number;

    value_type: "count" | "currency" | "percentage" | "duration_seconds";

    label: string;

    filter_period?: string | null;

    as_of?: string;

}

const SECRET = process.env.ASPRI_BUSINESS_METRIC_SECRET;

if (!SECRET) {

    throw new Error("ASPRI_BUSINESS_METRIC_SECRET env var required");

}

function verifySignature(rawBody: string, header: string | undefined): boolean {

    if (!header || !header.startsWith("sha256=")) return false;

    const provided = Buffer.from(header.slice(7), "hex");

    const expected = createHmac("sha256", SECRET!).update(rawBody).digest();

    if (provided.length !== expected.length) return false;

    return timingSafeEqual(provided, expected);

}

// Per-metric dispatcher. Add entries here as your business needs grow.

```



```
async function dispatchMetric(

  metricName: string,

  filterPeriod: string | undefined,

): Promise<MetricResponse | { error_code: string; error_message: string }> {

  switch (metricName) {

    case "orders_today": {

      const count = await queryOrderCountToday();

      return {

        metric_name: "orders_today",

        value: count,

        value_type: "count",

        label: "Total order hari ini",

        filter_period: filterPeriod ?? "today",

        as_of: new Date().toISOString(),

      };

    }

    case "revenue_mtd": {

      const totalldr = await queryRevenueMonthToDate();

      return {

        metric_name: "revenue_mtd",

        value: totalldr,
```

```

    value_type: "currency",

    label: "Pendapatan bulan ini",

    filter_period: filterPeriod ?? "mtd",

    as_of: new Date().toISOString(),

  };
}

case "avg_resolution_time": {

  const seconds = await queryAvgResolutionSeconds(filterPeriod);

  return {

    metric_name: "avg_resolution_time",

    value: seconds,

    value_type: "duration_seconds",

    label: "Rata-rata waktu resolusi ticket",

    filter_period: filterPeriod ?? "last_30d",

    as_of: new Date().toISOString(),

  };
}

default:

  return {

    error_code: "unknown_metric",

    error_message: `Metric '${metricName}' belum di-implement`,

  };
}

```

```

    }
}

// Backend stub functions — replace with your real query layer

async function queryOrderCountToday(): Promise<number> {

    // SELECT COUNT(*) FROM orders WHERE created_at::date = CURRENT_DATE

    return 0;

}

async function queryRevenueMonthToDate(): Promise<number> {

    // SELECT COALESCE(SUM(total_idr), 0) FROM orders

    // WHERE date_trunc('month', created_at) = date_trunc('month', CURRENT_DATE)

    return 0;

}

async function queryAvgResolutionSeconds(_period?: string): Promise<number> {

    return 0;

}

export function registerMetricRoute(app: FastifyInstance) {

    app.post("/aspri/business-metric", async (req, reply) => {

        const rawBody = req.body as string;

        const signature = req.headers["x-aspri-signature"] as string | undefined;

        if (!verifySignature(rawBody, signature)) {

            return reply.code(401).send({

```

```

        error_code: "signature_invalid",

        error_message: "HMAC signature verification failed",

    });

}

const body = JSON.parse(rawBody) as MetricRequestBody;

const result = await dispatchMetric(

    body.data.metric_name,

    body.data.filter_period,

);

if ("error_code" in result) {

    return reply.code(400).send(result);

}

return reply.send(result);

});

}

```

Note: handler harus menerima `rawBody` sebagai string sebelum `JSON.parse` — HMAC dihitung atas byte body persis (raw). Di Fastify, pakai `rawBody` plugin atau preserve via `addContentTypeParser`. Lihat [11-reference-implementation.md §5](#) untuk pola complete.

3.4 Consumer tool (EMBAN internal — owner tidak panggil langsung)

LLM agent CS/SA panggil tool `query_business_metric`:

```

query_business_metric(

    metric_name: "orders_today", // required, lowercase_alnum

```

```
filter_period: "today",    // optional
```

```
action_name: "business_metric" // optional override, default 'business_metric'
```

```
)
```

Tool wrap call ke action engine + parse structured response + compose `hint_llm` natural Indonesian sentence dengan `label` dari backend + `value_formatted` (locale-aware).

4. Template 2 — `business_list`

Paginated resource list. Backend dispatch berdasarkan `resource` string.

4.1 Action JSON

```
{  
  
  "name": "business_list",  
  
  "description": "Generic paginated resource list.",  
  
  "schema_version": "1.0",  
  
  "fields": {  
  
    "resource": {  
  
      "type": "string",  
  
      "required": true,  
  
      "ask": { "id": "Resource apa yang mau di-list?" },  
  
      "validation": { "pattern": "^[a-z][a-z0-9_]{0,62}$" }  
  
    },  
  
    "limit": {  
  
      "type": "number",
```

```
    "required": false,

    "validation": { "min": 1, "max": 100 }

  },

  "cursor": {

    "type": "string",

    "required": false,

    "validation": { "max_length": 256 }

  }

},

"endpoint": {

  "method": "POST",

  "url": "https://backend.tenant.example/aspri/business-list",

  "auth": {

    "type": "hmac_sha256",

    "secret_env": "ASPRI_BUSINESS_LIST_SECRET"

  },

  "timeout_ms": 8000

},

"response": {

  "mode": "structured"

},

"rate_limit": {
```

```
"global_per_minute": 60
}
}
```

4.2 Backend response contract

```
{
  "resource": "orders",
  "items": [
    {
      "id": "ORD-001",
      "title": "Order #001",
      "summary": "Rp 250.000 — Andi (WA)",
      "as_of": "2026-05-13T10:00:00+07:00"
    }
  ],
  "next_cursor": "eyJhZnRlcil6ljAwMSJ9",
  "total": 1234,
  "limit": 20,
  "as_of": "2026-05-13T10:00:00+07:00"
}
```

Field	Type	Required	Catatan
resource	string	optional	Echo back

Field	Type	Required	Catatan
<code>items</code>	array	wajib	Bisa empty array <code>[]</code> (no results)
<code>items[i].id</code>	string	wajib per item	Non-empty
<code>items[i].title</code>	string	wajib per item	Non-empty. Dipakai consumer untuk display
<code>items[i].summary</code>	string	optional	One-line description. Embedded di <code>hint_ilm</code>
<code>items[i].as_of</code>	string	optional	Per-item timestamp
<code>next_cursor</code>	string null	optional	Cursor opaque untuk halaman berikutnya. <code>null</code> atau missing = end
<code>total</code>	integer	optional	Total count keseluruhan (estimasi OK). Untuk progress indicator di <code>hint_ilm</code>
<code>limit</code>	integer	optional	Echo limit yang di-apply backend
<code>as_of</code>	string	optional	Timestamp top-level

Cursor convention: opaque string yang backend interpret sendiri. Bisa base64-encoded JSON, SQL row offset, etc. Consumer pass-through ke parameter `cursor` panggilan berikutnya.

4.3 Reference implementation (Node.js / TypeScript)

```
// src/api/routes/aspri-business-list.ts
```

```
interface ListRequestBody {
```

```
  data: {
```

```
    resource: string;
```



```
    limit?: number;

    cursor?: string | null;

};

}

interface ListItem {

    id: string;

    title: string;

    summary?: string;

    as_of?: string;

}

interface ListResponse {

    resource: string;

    items: ListItem[];

    next_cursor?: string | null;

    total?: number;

    limit: number;

    as_of: string;

}

async function dispatchList(

    resource: string,

    limit: number,
```

```

    cursor: string | null,

): Promise<ListResponse | { error_code: string; error_message: string }> {

    switch (resource) {

        case "orders": {

            const { rows, nextCursor, total } = await queryRecentOrders(limit, cursor);

            return {

                resource: "orders",

                items: rows.map((r) => ({

                    id: r.id,

                    title: `Order #${r.short_id}`,

                    summary: `${formatIdr(r.total_idr)} — ${r.customer_name} (${r.channel})`,

                })),

                next_cursor: nextCursor,

                total,

                limit,

                as_of: new Date().toISOString(),

            };

        }

        case "clients": {

            const { rows, nextCursor, total } = await queryClients(limit, cursor);

            return {

                resource: "clients",

```

```

items: rows.map((r) => ({
  id: r.id,
  title: r.name,
  summary: r.last_interaction
  ? `Terakhir kontak ${r.last_interaction}`
  : "Belum ada interaksi",
})),
next_cursor: nextCursor,
total,
limit,
as_of: new Date().toISOString(),
};
}
default:
return {
  error_code: "unknown_resource",
  error_message: `Resource '${resource}' belum di-support`,
};
}
}

// Cursor helper: base64-encoded JSON { after: lastId, sort: "created_desc" }

```

```

function decodeCursor(cursor: string | null): { after: string } | null {

    if (!cursor) return null;

    try {

        return JSON.parse(Buffer.from(cursor, "base64").toString());

    } catch {

        return null;

    }

}

function encodeCursor(payload: { after: string }): string {

    return Buffer.from(JSON.stringify(payload)).toString("base64");

}

async function queryRecentOrders(

    limit: number,

    cursorRaw: string | null,

): Promise<{

    rows: Array<{

        id: string;

        short_id: string;

        total_idr: number;

        customer_name: string;

        channel: string;

    }>;

```

```

    nextCursor: string | null;

    total: number;

}> {

    const cursor = decodeCursor(cursorRaw);

    // SELECT * FROM orders

    // WHERE ($cursor IS NULL OR id < $cursor)

    // ORDER BY id DESC LIMIT $limit + 1

    // → if returned > limit, last row's id = nextCursor.after

    // Replace stub below with real query layer

    const rows: typeof queryRecentOrders extends never ? never : Array<{

        id: string;

        short_id: string;

        total_idr: number;

        customer_name: string;

        channel: string;

    }> = [];

    const nextCursor =

        rows.length > limit ? encodeCursor({ after: rows[limit - 1]!.id }) : null;

    return { rows: rows.slice(0, limit), nextCursor, total: 0 };

}

async function queryClients(_limit: number, _cursor: string | null) {

```

```

    return { rows: [] as Array<{ id: string; name: string; last_interaction: string | null }>, nextCursor:
null, total: 0 };

}

function formatIdr(amount: number): string {

    return `Rp ${amount.toLocaleString("id-ID")}`;

}

```

4.4 Consumer tool

```

query_business_list(

    resource: "orders",    // required, lowercase_alnum

    limit: 20,             // optional, default 20, max 100

    cursor: "eyJhZnRlcil...", // optional, dari next_cursor sebelumnya

    action_name: "business_list" // optional override

)

```

Consumer tool clamping limit di range 1-100. Cursor passthrough ke backend.

5. Template 3 — **business_search**

Free-text search. Backend dispatch berdasarkan **resource** + apply **query** string.

5.1 Action JSON

```

{

    "name": "business_search",

    "description": "Generic free-text search on a resource.",

    "schema_version": "1.0",

```

```
"fields": {  
  
  "resource": {  
  
    "type": "string",  
  
    "required": true,  
  
    "validation": { "pattern": "[a-z][a-z0-9_]{0,62}$" }  
  
  },  
  
  "query": {  
  
    "type": "string",  
  
    "required": true,  
  
    "validation": { "min_length": 1, "max_length": 200 }  
  
  },  
  
  "limit": {  
  
    "type": "number",  
  
    "required": false,  
  
    "validation": { "min": 1, "max": 50 }  
  
  },  
  
  "endpoint": {  
  
    "method": "POST",  
  
    "url": "https://backend.tenant.example/aspri/business-search",  
  
    "auth": {  
  
      "type": "hmac_sha256",
```

```
    "secret_env": "ASPRI_BUSINESS_SEARCH_SECRET"

  },

  "timeout_ms": 8000

},

"response": {

  "mode": "structured"

},

"rate_limit": {

  "global_per_minute": 60

}

}
```

5.2 Backend response contract

```
{

  "resource": "products",

  "query": "kemeja batik",

  "items": [

    {

      "id": "P-001",

      "title": "Kemeja Batik Premium",

      "summary": "Size M-XL, Rp 175.000",

      "score": 0.95,
```



```

    "as_of": "2026-05-13T10:00:00+07:00"

  },

  "total_matched": 12,

  "limit": 10,

  "as_of": "2026-05-13T10:00:00+07:00"
}

```

Field	Type	Required	Catatan
resource	string	optional	Echo back
query	string	optional	Echo back
items	array	wajib	Bisa empty <code>[]</code> (no matches)
items[i].id	string	wajib per item	Non-empty
items[i].title	string	wajib per item	Non-empty
items[i].summary	string	optional	One-line description
items[i].score	number	optional	Relevance score (0-1 atau range backend). Embedded di hint_ilm
items[i].as_of	string	optional	Per-item timestamp
total_matched	integer	optional	Total matches before limit
limit	integer	optional	Echo limit yang di-apply
as_of	string	optional	Top-level timestamp

5.3 Reference implementation (Node.js / TypeScript)

```
// src/api/routes/aspri-business-search.ts
```

```
interface SearchRequestBody {
```

```
  data: {
```

```
    resource: string;
```

```
    query: string;
```

```
    limit?: number;
```

```
  };
```

```
}
```

```
async function dispatchSearch(
```

```
  resource: string,
```

```
  query: string,
```

```
  limit: number,
```

```
): Promise<unknown> {
```

```
  switch (resource) {
```

```
    case "products": {
```

```
      const { rows, totalMatched } = await searchProducts(query, limit);
```

```
      return {
```

```
        resource: "products",
```

```
        query,
```

```
        items: rows.map((r) => ({
```

```

        id: r.id,

        title: r.name,

        summary: `${r.size_range}, Rp ${r.price_idr.toLocaleString("id-ID")}`,

        score: r.relevance,

    })),

    total_matched: totalMatched,

    limit,

    as_of: new Date().toISOString(),

};

}

case "customers": {

    const { rows, totalMatched } = await searchCustomers(query, limit);

    return {

        resource: "customers",

        query,

        items: rows.map((r) => ({

            id: r.id,

            title: r.name,

            summary: r.phone ?? r.email ?? "Kontak: -",

            score: r.relevance,

        })),

        total_matched: totalMatched,

```

```
        limit,

        as_of: new Date().toISOString(),

    };

}

default:

    return {

        error_code: "unknown_resource",

        error_message: `Resource '${resource}' belum di-support untuk search`,

    };

}

}

async function searchProducts(

    _query: string,

    _limit: number,

): Promise<{

    rows: Array<{

        id: string;

        name: string;

        size_range: string;

        price_idr: number;

        relevance: number;
```

```

    }>;

    totalMatched: number;

}> {

    // SELECT id, name, size_range, price_idr, ts_rank(...) AS relevance

    // FROM products

    // WHERE search_tsv @@ plainto_tsquery('indonesian', $query)

    // ORDER BY relevance DESC LIMIT $limit

    return { rows: [], totalMatched: 0 };

}

async function searchCustomers(

    _query: string,

    _limit: number,

): Promise<{

    rows: Array<{

        id: string;

        name: string;

        phone: string | null;

        email: string | null;

        relevance: number;

    }>;

    totalMatched: number;

}> {

```

```
return { rows: [], totalMatched: 0 };  
  
}
```

5.4 Consumer tool

```
query_business_search(  
  
    resource: "products", // required, lowercase_alnum  
  
    query: "kemeja batik", // required, 1-200 char  
  
    limit: 10,           // optional, default 10, max 50 (lebih kecil dari list)  
  
    action_name: "business_search" // optional override  
  
)
```

6. Error code mapping

Consumer tool surface error code ke owner via **error_code** field:

Error code	Asal	Owner-facing meaning	Action
business_only	EMBAN gate	Tool dipanggil dari role non-business	LLM otomatis pakai tool lain
validation_error	EMBAN tool	metric_name / resource / query format invalid	LLM minta clarification
action_not_registered	EMBAN tool	Owner belum register action template ini	Owner panggil register_tenant_action dengan JSON dari doc ini
action_not_structured	EMBAN tool	Action terdaftar tapi response.mode != "structured"	Owner re-register dengan mode benar

Error code	Asal	Owner-facing meaning	Action
<code>action_definition_corrupt</code>	EMBAN tool	DB row JSON parse fail	Owner re-register
<code>endpoint_error</code>	Action engine	Backend tenant return 4xx/5xx atau network failure	Tenant IT cek backend log
<code>unexpected_async</code>	EMBAN tool	Engine return <code>awaiting_async</code> (tidak seharusnya untuk structured)	Owner re-register dengan structured murni
<code>backend_response_invalid</code>	EMBAN tool	Response JSON shape salah (lihat contract per template)	Tenant IT fix backend
Custom code (mis. <code>unknown_metric</code> , <code>db_down</code>)	Tenant backend	Domain-specific error	Tenant IT log + propagate ke owner

Backend tenant boleh return custom error_code dalam 4xx/5xx response body:

```
{
  "error_code": "unknown_metric",
  "error_message": "Metric 'orders_thisyear' belum di-implement di backend"
}
```

Consumer tool surface `error_code` value ini ke owner — owner dapat info actionable tanpa harus debug EMBAN engine.

7. FAQ + anti-patterns

7.1 "Boleh skip HMAC?"

Tidak. Setiap endpoint backend WAJIB verify HMAC walau resource read-only — kalau attacker temukan endpoint URL, mereka bisa enumerate metric_name / resource secara brute force. HMAC stops that. Selama development OK pakai `auth.type = "none"` untuk smoke test (lihat smoke script), tapi production WAJIB HMAC.

7.2 "Backend saya beda language (PHP/Python/Go) — masih bisa?"

Bisa. Skema doc ini language-agnostic — Node.js reference impl di §3.3, §4.3, §5.3 cuma example. Pola HMAC + JSON contract sama persis untuk semua bahasa. Lihat `02-authentication.md` untuk pola HMAC di language lain (PHP openssl_, Python hmac module, etc).

7.3 "Boleh return field tambahan di response?"

Boleh untuk debugging (`request_id`, `cache_hit`, `query_duration_ms`). Tapi consumer tool ignore field yang tidak di-spec di contract — jangan andalkan untuk muncul ke owner. Kalau butuh field ke owner, propose extension ke contract via Github issue.

7.4 "Backend saya lambat (>30 detik) — boleh async?"

Tidak. Template library trio strict structured mode (sync). Kalau backend lambat:

1. Optimize query (index, materialized view, cache layer)
2. Atau bikin custom action mode='async' dengan `immediate_ack` + callback (lihat `03-callbacks-api.md`). Custom action TIDAK pakai template library — perlu custom consumer tool.

7.5 "Multiple tenant beda endpoint per template?"

Tidak masalah. Setiap tenant register action JSON sendiri dengan `endpoint.url` mengarah ke backend mereka. Action engine resolve per-tenant — tidak ada konflik nama.

7.6 Anti-pattern: paraphrase response di backend

Jangan format string di backend (mis. return `"Total order hari ini: 42"`). Backend WAJIB return raw structured field (`value: 42, label: "...", value_type: "count"`). Consumer tool yang compose user-facing message — kalau backend pre-format, tool format Rupiah/persen/dll tidak akan jalan.

7.7 Anti-pattern: 1 action per metric

// JANGAN seperti ini

```
register_tenant_action({ name: "metric_orders_today", ... });
```

```
register_tenant_action({ name: "metric_revenue_mtd", ... });
```

```
register_tenant_action({ name: "metric_avg_order_value", ... });
```

// (20 action terpisah, 20 endpoint backend, 20 secret env var)

// PAKAI INI (template library pattern)

```
register_tenant_action({ name: "business_metric", ... }); // sekali register
```

// Backend dispatch internal berdasarkan metric_name

// → 1 endpoint, 1 secret, 1 action

Ratusan metric → 1 action, 1 endpoint, internal switch case di backend.

Cross-references

- [05-action-engine.md](#) — action JSON skeleton, HMAC outbound, structured mode hingga sini diadopsi
- [02-authentication.md](#) — HMAC implementation patterns (TS, PHP, Python, Go)
- [08-error-codes.md](#) — error envelope shape umum
- [11-reference-implementation.md](#) — RAI backend sebagai pola receiver standar (Node.js + Fastify + rawBody)