

09 — Contoh End-to-End

Audience: Tenant IT. **Prasyarat:** minimal [01-overview.md](#) dan [02-authentication.md](#).
Setiap contoh menandai doc yang spesifik dibutuhkan.

Doc ini merangkai endpoint-endpoint individual jadi flow lengkap. Kode contoh pakai Node.js untuk brevity — Python/curl setara ada di doc endpoint masing-masing.

Contoh 1 — Customer triggers custom async action (booking)

Audience: Tenant IT. **Doc terkait:** #3, #5.

Skenario

Tenant "Rent-a-Car Bagus" sudah register action [book_car](#) lewat chat dengan agent EMBAN.
Endpoint backend Tenant IT di <https://backend.rentbagus.com/aspri/book>.

Customer ngetik di WA: "Mau booking Avanza untuk hari Rabu." Agent CS mengumpulkan field yang kurang, lalu fire action.

Step 1 — EMBAN POST ke Tenant IT

EMBAN mengirim:

POST <https://backend.rentbagus.com/aspri/book>

Content-Type: application/json

Idempotency-Key: 01J1ZXK7SUB...

X-ASPRI-Tenant-Id: 01J1TENANT...

X-ASPRI-Submission-Id: 01J1ZXK7SUB...

X-ASPRI-Schema-Version: 1.0

X-ASPRI-Signature: sha256=abc123... (HMAC(secret, body) — TANPA timestamp)

{

```
"submission_id": "01J1ZXK7SUB...",  
  
"action": "book_car",  
  
"version": 3,  
  
"data": {  
  
  "car_model": "avanza",  
  
  "pickup_date": "2026-04-22",  
  
  "customer_phone": "+628123456789"  
  
},  
  
"callback_url": "https://api.rentalai.id/api/v1/callbacks/01J1ZXK7SUB...",  
  
"customer": {  
  
  "channel": "whatsapp",  
  
  "platform_id": "+628123456789",  
  
  "display_name": "Budi Santoso"  
  
},  
  
"tenant_id": "01J1TENANT...",  
  
"timestamp": "2026-04-20T10:30:00.000Z"  
  
}
```

Step 2 — Tenant IT handler

```
app.post("/aspri/book", express.raw({ type: "application/json" })), async (req, res) => {  
  
  const raw = req.body.toString("utf8");  
  
  // Verify (HMAC(secret, body) — no timestamp prefix)  
  
  const expected = "sha256=" +
```

```

    createHmac("sha256",
process.env.ASPRI_BOOK_CAR_SECRET).update(raw).digest("hex");

    if (!timingSafeEqual(Buffer.from(req.header("X-ASPRI-Signature") ?? ""),
Buffer.from(expected))) {

        return res.status(401).json({ error_code: "bad_signature" });

    }

    const payload = JSON.parse(raw);

    if (await bookingStore.exists(payload.submission_id)) {

        return res.status(200).json({ queued: true, duplicate: true });

    }

    // Async mode — queue dan return 202

    await bookingQueue.enqueue({

        submissionId: payload.submission_id,

        callbackUrl: payload.callback_url,

        data: payload.data,

        customer: payload.customer,

    });

    return res.status(202).json({ queued: true });

});

```

Step 3 — customer sementara terima **immediate_ack**

"Booking diproses, mohon tunggu..."

Step 4 — worker Tenant IT proses booking, lalu callback

async function processBookingJob(job) {

```
const result = await createBookingInLegacySystem(job.data);
```

```
const callbackBody = result.success
```

```
? {
```

```
  status: "success",
```

```
  template_key: "booking_confirmed",
```

```
  data: {
```

```
    booking_id: result.bookingId,
```

```
    car_model: job.data.car_model,
```

```
    pickup_date: job.data.pickup_date,
```

```
    price_idr: result.totalPrice,
```

```
  },
```

```
  files: result.invoicePdfUrl ? [{
```

```
    url: result.invoicePdfUrl,
```

```
    filename: `invoice-${result.bookingId}.pdf`,
```

```
    mime_type: "application/pdf",
```

```
  ] : [],
```

```
}
```

```
: {
```

```
  status: "error",
```

```
  error_code: "out_of_stock",
```

```
  error_message: "Unit sudah disewa tanggal tersebut",
```

```
    data: { alternative_date: result.nextAvailable },

  };

const raw = JSON.stringify(callbackBody);

const ts = Date.now();

const sig = "sha256=" + createHmac("sha256",

  process.env.ASPRI_BOOK_CAR_SECRET).update(`${ts}.${raw}`).digest("hex");

await fetch(job.callbackUrl, {

  method: "POST",

  headers: {

    "Content-Type": "application/json",

    "X-ASPRI-Timestamp": String(ts),

    "X-ASPRI-Signature": sig,

    "Idempotency-Key": job.submissionId,

  },

  body: raw,

});

}
```

Step 5 — customer terima rendered template + file

[Agent EMBAN]

Booking BK-2026-0412 untuk Toyota Avanza pada 22 April 2026 sudah

dikonfirmasi. Total: Rp850.000.

[attachment: invoice-BK-2026-0412.pdf]

Contoh 2 — Ticket escalation + human reply

Audience: Tenant IT. **Doc terkait:** #4.

Skenario

Customer tanya hal teknis kompleks. Agent CS EMBAN tidak bisa jawab dengan SOP yang ada → escalate. Petugas di sistem ticketing Tenant IT balas, customer terima reply via WA.

Step 1 — EMBAN POST ticket.created ke Tenant IT

POST <https://ticket.rentbagus.com/aspri/webhook>

Content-Type: application/json

X-ASPRI-Signature: sha256=... (HMAC(secret, ts + "." + body))

X-ASPRI-Timestamp: 1745145600123

X-ASPRI-Tenant-Id: 01J1TENANT...

Idempotency-Key: 01J1TICKET001...

```
{  
  "event": "ticket.created",  
  "ticket_id": "01J1TICKET001...",  
  "ticket_number": "RAI-20260420-000007",  
  "tenant_id": "01J1TENANT...",  
  "customer": {  
    "channel": "whatsapp",  
    "platform_id": "+628123456789",  
    "display_name": "Budi Santoso"
```

```
},  
  
"subject": "Tidak bisa login ke dashboard",  
  
"initial_message": "Halo min, saya sudah reset password tapi tetap error...",  
  
"priority": "normal",  
  
"sla_due_at": "2026-04-20T14:30:00.000Z",  
  
"reply_callback_url": "https://api.rentalai.id/api/v1/tickets/01J1TICKET001.../reply",  
  
"timestamp": "2026-04-20T10:30:00.000Z"  
  
}
```

Step 2 — Tenant IT verify + simpan

Seperti handler di doc #4 §3.

Step 3 — petugas balas di UI Tenant IT

Petugas "Andi" ketik balasan di dashboard internal. Backend Tenant IT memanggil reply endpoint:

```
async function postReply(ticketId, message, senderName, closeTicket = false) {  
  
  const body = JSON.stringify({  
  
    message,  
  
    close_ticket: closeTicket,  
  
    sender_label: `${senderName} dari Admin`,  
  
  });  
  
  const ts = Date.now();  
  
  const sig = "sha256=" + createHmac("sha256",  
  
    process.env.ASPRI_TICKET_WEBHOOK_SECRET).update(`${ts}.${body}`).digest("hex");
```

```

const res = await fetch(

  `https://api.rentalai.id/api/v1/tickets/${ticketId}/reply`,

  {

    method: "POST",

    headers: {

      "Content-Type": "application/json",

      "X-ASPRI-Timestamp": String(ts),

      "X-ASPRI-Signature": sig,

      "Idempotency-Key": crypto.randomUUID(),

    },

    body,

  },

);

return res.json();

}

await postReply(

  "01J1TICKET001...",

  "Halo Pak Budi, password sudah kami reset manual. Silakan coba login lagi dengan email terdaftar + password 'temp123'. Mohon ganti password setelah login pertama.",

  "Andi",

  true, // close ticket setelah ini

);

```


Step 4 — customer terima di WA

[Andi dari Admin] Halo Pak Budi, password sudah kami reset manual.

Silakan coba login lagi dengan email terdaftar + password 'temp123'.

Mohon ganti password setelah login pertama.

Status ticket → **closed**.

Contoh 3 — File delivery happy path + partial failure

Audience: Tenant IT. **Doc terkait:** #3 §6, #6.

Skenario

Tenant IT kirim callback sukses dengan 3 file attached: invoice PDF, foto unit, dan satu file yang hostname-nya belum di allowlist.

Request

```
await fetch("https://api.rentalai.id/api/v1/callbacks/01J1SUB...", {  
  
  method: "POST",  
  
  headers: { /* ... HMAC headers ... */ },  
  
  body: JSON.stringify({  
  
    status: "success",  
  
    template_key: "booking_confirmed",  
  
    data: { booking_id: "BK-2026-0412", car_model: "Toyota Avanza" },  
  
    files: [  
  
      {  
  
        url: "https://files.rentbagus.com/invoice/BK-2026-0412.pdf",
```

```
    filename: "invoice.pdf",

    mime_type: "application/pdf",

    expected_size_bytes: 524288,

  },

  {

    url: "https://files.rentbagus.com/photo/avanza-2023.jpg",

    filename: "unit.jpg",

    mime_type: "image/jpeg",

  },

  {

    url: "https://new-cdn.rentbagus.com/terms.pdf", // hostname belum di allowlist

    filename: "terms.pdf",

    mime_type: "application/pdf",

  },

],

}},

});
```

Response (200)

```
{

  "ok": true,

  "delivered": true,

  "customer_channel": "whatsapp",
```

```
"submission_id": "01J1SUB...",

"files": {

  "delivered": 2,

  "rejected": 1,

  "failed": 0,

  "details": [

    { "url": "https://files.rentbagus.com/invoice/BK-2026-0412.pdf", "status": "delivered" },

    { "url": "https://files.rentbagus.com/photo/avanza-2023.jpg", "status": "delivered" },

    {

      "url": "https://new-cdn.rentbagus.com/terms.pdf",

      "status": "rejected",

      "error": "host_not_allowed"

    }

  ]

}

}
```

Yang customer terima

1. Text template "Booking confirmed..." (dari [template_key](#)).
 2. Invoice PDF (di-deliver).
 3. Foto unit JPG (di-deliver).
 4. **Tidak** terima terms.pdf — itu rejected di sisi EMBAN. Tenant IT bisa escalate tambah [new-cdn.rentbagus.com](#) ke allowlist, lalu kirim callback baru dengan hanya file itu saja.
-

Contoh 4 — Working hours guard fires

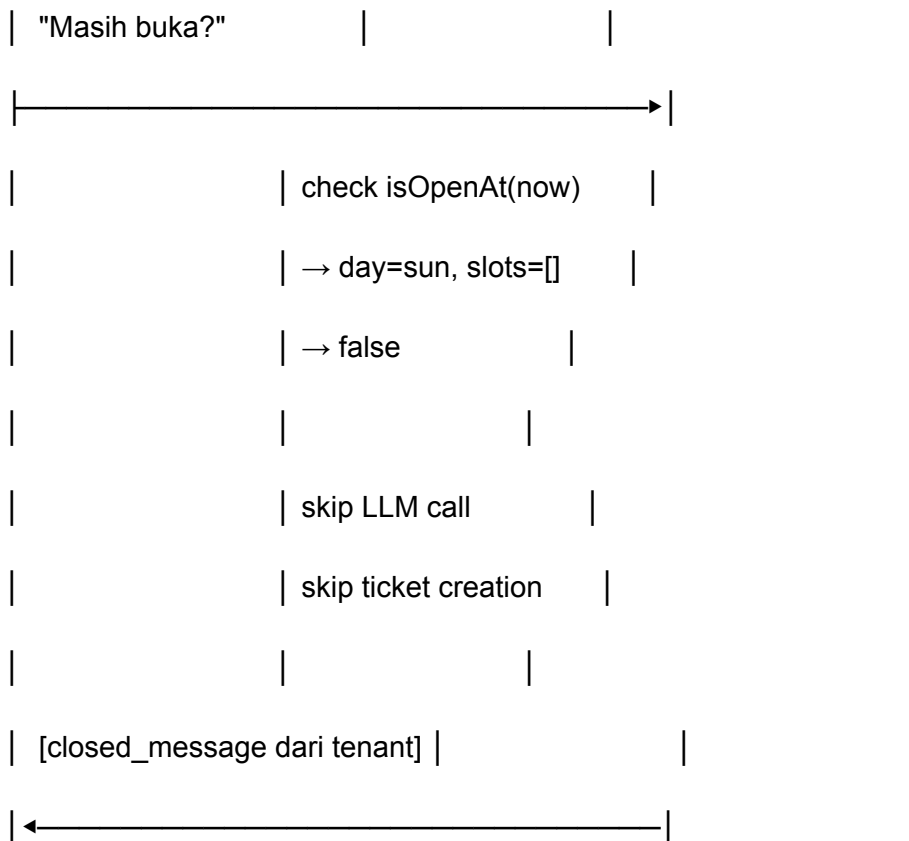
Audience: Tenant IT (awareness). **Doc terkait:** #7.

Skenario

Tenant "Warung Mak Ida" jam operasional Senin-Jumat 09:00-17:00, Sabtu 09:00-13:00, Minggu tutup. Customer kirim WA Minggu jam 10 pagi: "Masih buka nggak?"

Flow

Customer WA EMBAN Engine LLM / Ticketing



Yang customer terima

Halo! Warung Mak Ida libur hari Minggu. Kami buka lagi Senin 09:00.

Terima kasih! 🙏

Tenant IT perspective

- Tidak ada webhook `ticket.created` — karena tidak ada ticket yang dibuat.
- Tidak ada callback API yang terpicu — tidak ada action fire.
- Petugas di Tenant IT dashboard tidak melihat activity.

Kalau Tenant IT perlu tracking "inbound outside hours" untuk insight, harus pull dari log sendiri (channel-level) atau request feature ke tim EMBAN (roadmap `customer.message_dropped` event).

Contoh 5 — Product query (sync action) — RAI reference stub

Audience: Tenant IT. **Doc terkait:** #5.

Skenario

Tenant Rental AI Indonesia (RAI) ingin agent CS bisa jawab pertanyaan customer "Stok TV LED 55 inci tersedia berapa unit?" tanpa harus eskalasi ke petugas. Tenant register action `query_product_info` yang nge-call backend RAI sendiri.

Implementasi reference live di repo:

`rai-backend/src/api/routes/aspri-product-query.ts` + `rai-backend/src/services/product-query-stub.ts`. Stub mengembalikan katalog hardcoded; di production, ganti `findProducts()` dengan lookup ke inventory system. Wrapper HMAC + idempotency + response shape tetap identik.

Step 1 — Action definition

Tenant register action lewat percakapan dengan agent EMBAN master-bot. Definisi (sesuai schema `src/services/action-schema.ts`):

```
{  
  
  "name": "query_product_info",  
  
  "description": "Lookup ketersediaan + harga produk yang disewakan RAI berdasarkan keyword  
+ category.",  
  
  "schema_version": "1.0",  
  
  "trigger_hints": ["stok", "ketersediaan", "harga sewa", "rental"],  
}
```

```
"fields": {  
  
  "query": {  
  
    "type": "free_text",  
  
    "required": true,  
  
    "ask": { "id": "Mau cari produk apa? (sebut keyword, mis. 'tv 55', 'projector')" }  
  
  },  
  
  "category": {  
  
    "type": "enum_single",  
  
    "required": false,  
  
    "default": "all",  
  
    "ask": { "id": "Kategori spesifik? (kalau tidak yakin, ketik 'semua')" },  
  
    "options": [  
  
      { "value": "electronics",    "label": { "id": "Elektronik" } },  
  
      { "value": "office_equipment", "label": { "id": "Peralatan Kantor" } },  
  
      { "value": "all",           "label": { "id": "Semua kategori" } }  
  
    ]  
  
  }  
  
},  
  
"endpoint": {  
  
  "url": "https://api.rentalai.id/webhooks/aspri-product-query",  
  
  "auth": {
```

```

    "scheme": "hmac_body_only",

    "secret_env": "ASPRI_PRODUCT_QUERY_SECRET"

  },

  "response": {

    "mode": "sync",

    "template": {

      "id": "Hasil pencarian \"{{query}}\" ({{count}} produk):\n{{#items}}- {{name}} ({{sku}}):  
Rp{{daily_price_idr}}/hari, stok {{available_units}} unit\n{{/items}}"

    }

  }

}

```

Step 2 — EMBAN POST ke RAI

Customer ngetik di WA: "Mau sewa TV LED, ada yang 55 inci?". Agent CS mengisi `query="tv 55"` + `category="electronics"` (auto-derived dari hint), lalu fire action.

POST <https://api.rentalai.id/webhooks/aspri-product-query>

Content-Type: application/json

Idempotency-Key: 01J1ZXK7SUB...

X-ASPRI-Tenant-Id: 01J1TENANT...

X-ASPRI-Submission-Id: 01J1ZXK7SUB...

X-ASPRI-Schema-Version: 1.0

X-ASPRI-Signature: sha256=abc123... (HMAC(secret, body) — body-only)

```
{
```

```
"submission_id": "01J1ZXK7SUB...",  
  
"action": "query_product_info",  
  
"version": 1,  
  
"data": {  
  
  "query": "tv 55",  
  
  "category": "electronics"  
  
},  
  
"callback_url": "https://api.rentalai.id/api/v1/callbacks/01J1ZXK7SUB...",  
  
"customer": {  
  
  "channel": "whatsapp",  
  
  "platform_id": "+628123456789",  
  
  "display_name": "Budi Santoso"  
  
},  
  
"tenant_id": "01J1TENANT...",  
  
"timestamp": "2026-04-20T10:30:00.000Z"  
  
}
```

Step 3 — RAI handler verifies + responds (SYNC)

Snippet lengkap di [rai-backend/src/api/routes/aspri-product-query.ts](#):

```
app.post(  
  
  "/webhooks/aspri-product-query",  
  
  {
```



```

preHandler: [

  hmacVerifyPreHandler({

    getSecret: () => env.ASPRI_PRODUCT_QUERY_SECRET || null,

    bodyOnly: true,      // Action Engine: HMAC(secret, body), no ts prefix

  }),

],

},

async (req, reply) => {

  const parsed = ActionPayloadSchema.safeParse(req.body);

  if (!parsed.success) return reply.code(400).send({

    ok: false, error_code: "schema_invalid",

    details: parsed.error.flatten(),

  });

  const products = findProducts(parsed.data.data.query, parsed.data.data.category ?? "all");

  return reply.code(200).send({

    submission_id: parsed.data.submission_id,

    query: parsed.data.data.query,

    category: parsed.data.data.category ?? "all",

    count: products.length,

    items: products,

  });

},

```

);

Step 4 — Response body (sync — di-render langsung ke customer)

```
{  
  "submission_id": "01J1ZXK7SUB...",  
  "query": "tv 55",  
  "category": "electronics",  
  "count": 1,  
  "items": [  
    {  
      "sku": "ELEC-TV-LED-55",  
      "name": "TV LED 55 inci",  
      "category": "electronics",  
      "daily_price_idr": 250000,  
      "available_units": 6,  
      "description": "Smart TV LED 55 inci 4K. Bundle dengan stand mobile + HDMI cable."  
    }  
  ]  
}
```

Step 5 — Customer terima rendered template

[Agent EMBAN]

Hasil pencarian "tv 55" (1 produk):

- TV LED 55 inci (ELEC-TV-LED-55): Rp250000/hari, stok 6 unit

Kenapa stub, bukan real DB?

Tujuan G3 di `docs/integrator/` adalah **kontrak integrasi**, bukan implementasi inventory RAI. Live tenant ganti `findProducts()` di `product-query-stub.ts` dengan call ke ERP / SQL lookup / GraphQL — wrapper HMAC + Zod validate + response shape **tidak perlu diubah**. Pisahnya jelas: route handler = transport, service module = domain logic.

Body-only HMAC bukan ts-prefixed

Action Engine pakai skema HMAC tanpa timestamp prefix (`HMAC(secret, body)`) karena engine kontrol retry sendiri — tidak ada risk replay dari pihak ketiga. Lihat doc #5 §3 catatan dan code di `rai-backend/src/api/middleware/hmac.ts` (option `bodyOnly: true`).

Ringkasan pola umum

Dari 5 contoh di atas:

Pola	Terlihat di
HMAC sign body + timestamp (ts-prefixed)	Contoh 1 Step 4 (callback), Contoh 2
HMAC sign body only (action outbound)	Contoh 1 Step 1, Contoh 5
<code>Idempotency-Key</code> untuk retry safety	Contoh 1, 5
<code>submission_id</code> / <code>ticket_id</code> sebagai natural dedup	Contoh 1, 2, 5
Partial success di response	Contoh 3
No-op di engine saat guard fire	Contoh 4
Sync mode response (template render direct)	Contoh 5

Pola-pola ini konsisten lintas semua endpoint. Kalau Anda mengikuti satu endpoint, endpoint lain akan terasa familiar.

Lanjut: **#10 Glossary** untuk referensi istilah kalau ada yang asing, atau **#11 Reference implementation** untuk lihat code yang menjalankan Contoh 2 + 5 di production.