```
module: 34
title: Public API & Webhooks
phase: 1
status: detail-spec-draft
last_updated: 2026-05-08
session_source: brainstorming-session-2026-05-07-2206.md
related_lps: [LP-6, LP-9, LP-10, LP-11, LP-13]
spec_style: requirements-pur
```

Modul 34 — Public API & Webhooks

> ***Detail-Spec gemäß LP-9.*** *Phase-1-Basic. Read-API (GET) für Hauptentitäten + eingehende Webhooks für Lead-Push + ausgehende Webhooks als Automation-Engine-Action (M17). API-Key-Verwaltung kommt aus M27 Block E. OpenAPI-Doku auto-generiert. Voll-CRUD-API erst Phase 2 (bewusste Scope-Beschränkung).*

Themenblöcke

```
A. Datenmodell (Logging und Webhook-Subscriptions)
B. Authentifizierung & Scopes (Verweis auf M27 Block E)
C. Read-API (GET-Endpoints für Hauptentitäten)
D. Eingehende Webhooks (Lead-Push)
E. Ausgehende Webhooks (Subscription-Modell)
F. Allgemeine Event-Ingestion
G. OpenAPI-Doku
H. Rate-Limiting
I. Versionierung & Stabilität
J. Permissions / Scope-Mapping
K. Phase 2
```

Block A — Datenmodell

IntegrationApiRequest

Audit-Log aller eingehenden API-Requests. Tenant-scoped.

- `id`, `tenant_id`, `created_at`
- `api_key_id` (FK auf ApiKey aus M27 — nullable bei unauth Requests)
- `route_pattern` (z.B. `/api/v1/contacts/:id`)
- `http_method` (GET / POST / PUT / DELETE)
- `query_params` (JSONB — sanitized)
- `request_body_size_bytes`
- `response_status_code` (Integer)
- `response_size_bytes`
- `duration_ms`
- `client_ip`
- `user_agent`
- `error_message` (nullable, Klartext)

Indizes:
- `(tenant_id, api_key_id, created_at DESC)` — pro-Key-Logs
- `(tenant_id, route_pattern, created_at DESC)` — pro-Endpoint-Stats
- `(tenant_id, response_status_code, created_at DESC)` — Error-Monitoring

Retention: 90 Tage, danach gelöscht (außer 4xx/5xx → 180 Tage).

WebhookSubscription

Definition eines vom Tenant abonnierten ausgehenden Webhooks. Tenant-scoped.

- `id`, `tenant_id`, `created_at`, `updated_at`, `created_by_user_id`
- `name` (interner Name)
- `target_url` (URL, an die ausgehende Events gesendet werden)
- `secret_encrypted` (für HMAC-Signatur)
- `event_types[]` (Liste der abonnierten Events, z.B. `contact.created`, `deal.stage_changed`, `project.completed`)
- `filter_definition` (optional — JSONB für event-spezifische Filter)
- `is_active` (Boolean)
- `failure_count_consecutive` (Integer — Auto-Disable nach N Fehlern)
- `last_success_at`, `last_failure_at`, `last_failure_reason`
- `archived_at`

Indizes: `(tenant_id, is_active, event_types)` für Event-Dispatch.

WebhookDelivery

Pro Event-Auslösung pro Subscription ein Eintrag.

- `id`, `tenant_id`, `subscription_id`, `created_at`
- `event_type`, `event_id` (für Idempotenz beim Empfänger)
- `payload` (JSONB — finaler Body)
- `status` (Enum: `pending | delivered | failed | exhausted | discarded`)
- `attempt_count` (Integer, max. 5)
- `last_attempt_at`, `delivered_at`
- `last_response_status_code`, `last_response_body_excerpt` (max. 500 Zeichen)

Indizes: `(tenant_id, subscription_id, status)`, `(status, last_attempt_at)` für Retry-Worker.

WebhookDeliveryAttempt

Pro Versuch ein Audit-Eintrag.

- `id`, `tenant_id`, `delivery_id`, `attempted_at`
- `attempt_number` (1..5)
- `request_signature` (HMAC-Hex, gespeichert für Diagnose)
- `response_status_code` (Integer, nullable bei Network-Error)
- `response_body_excerpt` (max. 500 Zeichen)
- `response_time_ms`
- `error_text` (Klartext bei Network/Timeout-Fehler)

Retention: 30 Tage.

IncomingWebhookIngestion

Logs der eingehenden Webhooks (Lead-Push, Form-Submissions, etc.). Tenant-scoped.

- `id`, `tenant_id`, `received_at`

- `slug` (z.B. `website-form-kontakt` — vom Tenant definiert)
- `source_kind` (Enum: `lead | form_submission | generic_event` — Phase 1 only Lead-fokussiert)
- `auth_method` (Enum: `api_key | hmac_signature | unauthenticated_with_token`)
- `payload` (JSONB)
- `processing_status` (Enum: `received | processing | done | failed | rejected_invalid_payload | rejected_auth | rejected_rate_limit`)
- `processed_at`, `error_text`
- `derived_lead_id` / `derived_contact_id` (FK, nullable — was wurde erzeugt)
- `client_ip`, `user_agent`

Retention: 90 Tage erfolgreiche, 180 Tage fehlerhafte (für Diagnose).

Block B — Authentifizierung & Scopes

Authentifizierung-Methoden

Methode 1: API-Key-Bearer (Standard)
- Header: `Authorization: Bearer hg_<random_64_chars>`
- API-Key-Verwaltung in M27 Block E
- Pro Request: Loader prüft Key-Existenz, Scope, Tenant-Zuordnung, Aktiv-Status

Methode 2: HMAC-Signatur (für eingehende Webhooks)
- Header: `X-HiveGrid-Signature: sha256=<hex>`
- Body wird mit Webhook-Slug-Secret HMAC-SHA256 signiert
- Verifikation mit Constant-Time-Compare gegen Slug-spezifisches Secret

Methode 3: URL-Token (Fallback für Form-Webhooks)
- URL-Path enthält Token: `/api/v1/webhooks/incoming/{tenant}/{slug}?token=...`
- Mit Klartext-Warnung im UI: *„URL-Token ist weniger sicher als API-Key — nur für Form-Anbieter ohne Header-Support nutzen"*

Scope-Modell (kommt aus M27 Block E, hier konkretisiert)

Scope	Erlaubt
`read_only`	Alle GET-Endpoints außer sensible Felder (Mails, HR, finanzielle Daten)
`read_full`	Alle GET-Endpoints inkl. sensible Daten
`webhook_inbound`	Eingehende Lead-/Form-Webhook-Endpoints — kein Read-Zugriff
`automation_trigger`	Auslösen von Automation-Triggern via API

Scope-Mapping zu Endpoints

- Default Phase 1: 1 Key = 1 Scope
- Phase 2: Multi-Scope-Keys + granulare Scopes pro Modul

Klartext-Fehler

- 401 unauthorized: *„API-Key fehlt oder ungültig. Header* `Authorization: Bearer hg_...` *setzen."*
- 403 forbidden: *„Dieser API-Key hat nicht die Berechtigung für diese Aktion. Erforderlicher Scope:* `read_full` *."*
- 429 rate limit: *„Rate-Limit erreicht: 1.000 Anfragen/Stunde. Reset um [timestamp]. Tenant-Admin kann Limit erhöhen."*

Block C — Read-API (GET-Endpoints)

Endpoint-Übersicht

```
GET  /api/v1/contacts                  → Liste Kontakte
GET  /api/v1/contacts/:id              → Einzelner Kontakt
GET  /api/v1/companies                 → Liste Firmen
GET  /api/v1/companies/:id             → Einzelne Firma
GET  /api/v1/deals                     → Liste Deals
GET  /api/v1/deals/:id                 → Einzelner Deal
GET  /api/v1/projects                  → Liste Projekte
GET  /api/v1/projects/:id              → Einzelnes Projekt
GET  /api/v1/tasks                     → Liste Tasks
GET  /api/v1/tasks/:id                 → Einzelner Task
GET  /api/v1/leads                     → Liste Leads
GET  /api/v1/leads/:id                 → Einzelner Lead
```

Gemeinsame Konventionen

Pagination (Cursor-basiert):
- Query-Param `cursor` (opaque, base64-encoded)
- Query-Param `limit` (Default 50, max. 200)
- Response enthält `meta.next_cursor` (null wenn keine weiteren)

Filter:
- Per Query-Param, z.B. `?tag=newsletter&assignee_id=...`
- Multi-Value: `?tag=a&tag=b` (OR-Logik)
- Date-Range: `?created_at_gte=2026-01-01&created_at_lte=2026-12-31`
- Custom-Felder: `?custom.heizungsanlage_modell=Vaillant`

Sortierung:
- Query-Param `sort` (z.B. `sort=-updated_at` für absteigend, `sort=last_name` für aufsteigend)
- Default pro Endpoint dokumentiert

Field-Selection (Sparse Fieldsets):
- Query-Param `fields` (z.B. `?fields=id,first_name,last_name,emails`)
- Reduziert Payload bei großen Listen

Embed Relations:
- Query-Param `embed` (z.B. `?embed=primary_company,assigned_to`)
- Default: nur IDs der Verknüpfungen
- Mit Embed: vollständige Sub-Objekte
- Tiefe max. 1 (kein verschachteltes Embedding in Phase 1)

Response-Format

Erfolg:

```
{
  "data": { ... } oder [...],
  "meta": {
    "next_cursor": "...",
    "total_count_estimate": 1234
  }
}
```

Fehler:

```
{
  "error": {
    "code": "validation_failed",
    "message": "Klartext-Fehler-Beschreibung",
    "details": { "field": "explanation" }
  }
}
```

Endpoint-Details (Beispiel: GET /api/v1/contacts)

Filter:
- `tag` (Multi)
- `assignee_id`
- `lead_source`
- `is_endkunde`
- `language`
- `created_at_gte`, `created_at_lte`, `updated_at_gte`, `updated_at_lte`
- `q` (Volltext über Name/Email/Phone/Notes)
- `archived` (Boolean — Default false)
- `custom.{field_key}` (für Custom-Felder, dynamisch validiert)

Sortierung:
- `last_name`, `first_name`, `created_at`, `updated_at`, `last_activity_at`
- Default: `-last_activity_at`

Embed-Optionen:
- `primary_company`, `assigned_to`, `tags`, `emails`, `phones`, `custom_fields`

Sensible Felder (nur mit `read_full`-Scope):
- `notes`
- `custom_fields` mit `is_sensitive=true` (z.B. HR-Personalnummer in Phase 2)
- Activities

Endpoint-Details (Beispiel: GET /api/v1/deals)

Filter: `pipeline_id`, `stage_id`, `assignee_id`, `tag`, `value_gte`, `value_lte`, `state` (`open|won|lost`), `contact_id`, `company_id`, `created_at_*`, `updated_at_*`

Sortierung: `created_at`, `updated_at`, `value`, `expected_close_date`, `last_activity_at` — Default: `-last_activity_at`

Embed: `pipeline`, `stage`, `primary_contact`, `primary_company`, `assigned_to`, `tags`

Permissions-aware-Default (AV #5)

- Endpoints filtern automatisch nach Scope
- Sensible Felder werden bei `read_only` aus Response entfernt (nicht 403, sondern Feld-Auslassung mit Hinweis im OpenAPI-Schema)

Block D — Eingehende Webhooks (Lead-Push)

Endpoint

```
POST /api/v1/webhooks/incoming/{tenant_slug}/{webhook_slug}
```

Webhook-Slug-Verwaltung

Verwaltung in `/{tenant}/admin/api-keys/incoming-webhooks/` (gemeinsam mit API-Keys aus M27).

Felder pro Slug:
- `slug` (Tenant-eindeutig, z.B. `website-kontaktformular`)
- `purpose` (Enum: `lead | form_submission | generic_event` — Phase 1: meist `lead`)
- `auth_method` (Enum: `api_key | hmac_signature | url_token`)
- `secret_or_token_encrypted`
- `payload_mapping` (JSONB — Mapping von Webhook-Field zu HiveGrid-Field)
- `default_assignee_id` (FK, optional)
- `default_tags[]`
- `auto_promote_to_contact` (Boolean — sofort als Contact statt Lead anlegen, Default false)
- `is_active`, `archived_at`

Standard-Payload-Format (Lead-Push)

Tenant kann eigenes Mapping definieren, aber Default-Schema für direkten POST:

```
{
  "first_name": "Max",
  "last_name": "Mustermann",
  "email": "max@example.de",
  "phone": "+491234567890",
  "message": "Anfrage zur Wartung",
  "source_label": "Website-Form",
  "metadata": { ...beliebige Felder... }
}
```

Verarbeitung

1. Auth verifizieren (Methode aus Slug-Config)
2. Rate-Limit prüfen (siehe Block H)
3. Payload validieren (gegen `payload_mapping`)
4. `IncomingWebhookIngestion` mit `processing_status=processing` anlegen
5. Lead anlegen (Modul 2) mit Mapped-Werten + Source = Slug
6. Auto-Match-Versuch: existiert Contact mit selber Email? → falls ja, Activity am Contact statt neuem Lead (oder konfigurierbar pro Slug)
7. Auto-Tag aus `default_tags`
8. `processing_status=done`, `derived_lead_id` gesetzt
9. Response: `201 Created` mit `{ "id": "...", "kind": "lead" }`

Fehler-Pfade

- Ungültiges Payload: `400 Bad Request` mit Klartext-Validation-Errors
- Auth-Fehler: `401 Unauthorized` mit Klartext
- Slug nicht gefunden / inactive: `404 Not Found`
- Rate-Limit: `429`
- Tenant in `frozen`-Status: `403 Forbidden` mit Klartext: *„Workspace ist pausiert — eingehende Webhooks werden aktuell verworfen."*
- Server-Fehler: `500` + Eintrag in `IncomingWebhookIngestion.processing_status=failed` für Debugging

Klartext-Hinweise im Tenant-UI

- Slug-Detail-View zeigt: letzte 100 Ingestion-Events mit Status, Fehler-Klartext, Replay-Button (Phase 2)
- *„Webhook-Slug* `website-form` *hat in den letzten 24h: 47 Erfolge, 2 Fehler. [Logs anzeigen]"*

Block E — Ausgehende Webhooks

Subscription-Verwaltung in `/{tenant}/admin/webhooks/`

Felder pro Subscription:
- `name` (interner Name)
- `target_url`
- `secret` (HMAC-Sign — System-generiert, einmalig sichtbar bei Anlage)
- `event_types[]` (Multi-Select aus verfügbaren Events)
- `filter_definition` (optional)

Verfügbare Event-Types (Phase 1)

```
contact.created
contact.updated
contact.archived
contact.deleted
contact.tag_added
contact.tag_removed

company.created
company.updated
company.archived

lead.created
lead.qualified
lead.converted
lead.lost
lead.spam

deal.created
deal.stage_changed
deal.won
deal.lost
deal.archived

project.created
project.stage_changed
project.completed
project.archived

task.created
task.completed
task.assigned

email.received
email.sent
email.bounced

call.completed
call.missed
```

Event-Payload-Format

```
{
  "event_id": "evt_...",         // unique pro Event-Auslösung, für Idempotenz
  "event_type": "contact.created",
  "tenant_id": "...",
  "tenant_slug": "geerkens",
  "occurred_at": "2026-05-08T10:23:00Z",
  "data": { /* vollständiges Entity-Objekt analog zu GET /api/v1/<entity>/:id */ },
  "previous_data": { /* nur bei *.updated/*_changed – vorheriger Zustand */ }
}
```

Signatur

- Header `X-HiveGrid-Signature: sha256=<hex>` mit HMAC-SHA256(secret, body)
- Header `X-HiveGrid-Event: contact.created`
- Header `X-HiveGrid-Event-Id: evt_...` (für Idempotenz)
- Empfänger soll Signatur verifizieren + bei Idempotenz-Konflikt 200 OK ohne Re-Processing antworten

Retry-Strategie

- Bei Response 2xx: `delivered`
- Bei 4xx (außer 408/429): `failed`, kein Retry — Klartext: *„Endpoint hat 4xx geantwortet — möglicherweise falsch konfiguriert. Subscription bleibt aktiv."*
- Bei 5xx, 408, 429, Network/Timeout: Retry mit exponentiellem Backoff (1min, 5min, 30min, 2h, 12h) — max. 5 Versuche
- Nach 5 Versuchen: `exhausted`, `failure_count_consecutive` inkrementiert
- Bei `failure_count_consecutive >= 10`: Subscription auto-deaktiviert (`is_active=false`) mit Klartext-Notification: *„Webhook ‚[Name]' wurde nach 10 fehlgeschlagenen Auslieferungen deaktiviert. [Reaktivieren]"*

Delivery-Stats in Subscription-Detail-View

- Letzte 100 Deliveries mit Status, Response-Code, Response-Excerpt
- Aggregat: *„Letzte 7 Tage: 234 Deliveries, 230 erfolgreich (98.3%)"*
- Replay-Button pro fehlgeschlagener Delivery (manueller Re-Versuch)
- Test-Funktion: *„Test-Event senden"* — sendet Sample-Payload mit `event_type=test.ping`

Permission

- Subscription-Verwaltung: `tenant.admin` ODER `integrations.admin`
- Auslösung: System-Actor

Block F — Allgemeine Event-Ingestion

Endpoint

```
POST /api/v1/integrations/events
```

Generischer Ingest für externe Systeme, die Custom-Events pushen wollen (z.B. eigenes ERP pushed Status-Update).

Auth

- API-Key mit Scope `automation_trigger` (oder `webhook_inbound`)

Payload

```
{
  "event_type": "external.deal_status_changed",
  "external_id": "deal-12345",
  "data": { ... }
}
```

Verarbeitung

- Event landet in Automation-Engine (M17) als Trigger `external_event`
- Event-Filter pro Automation: `event_type` Match
- Phase 1: nur als Trigger (keine direkte Datenmanipulation)
- Phase 2: Event-Mapping zu CRM-Updates

Block G — OpenAPI-Doku

Auto-Generierung

- OpenAPI-3.1-Spec generiert aus Code (Zod-Schemas oder ähnlich)
- Endpoint: `/api/v1/openapi.json`
- Includiert: alle Endpoints, Request-/Response-Schemas, Auth-Methoden, Beispiele

Swagger-UI

- Endpoint: `/swagger`
- Öffentlich zugänglich (nicht Tenant-spezifisch)
- *„Try it out"*-Funktion mit API-Key-Eingabe-Feld
- Beispiel-Requests für alle Endpoints

Klartext-Beschreibungen

- Jeder Endpoint hat Klartext-Description in OpenAPI
- Field-Level-Descriptions für nicht-offensichtliche Felder
- Pflicht-Felder vs. Optional klar markiert

Tenant-spezifische Doku-Hinweise

- In `/{tenant}/admin/api-keys/` Verlinkung zu Swagger mit Hinweis: *„Verwende deinen API-Key oben rechts in Swagger zum Testen"*

Block H — Rate-Limiting

Limits (Phase 1)

- **Default pro API-Key:** 1.000 Requests/Stunde (rolling)
- **Burst:** kurzfristig 50 Requests/Sekunde tolerabel
- **Pro Tenant aggregiert:** 10.000 Requests/Stunde
- **Eingehende Webhooks pro Slug:** 100 Requests/Minute
- **Tenant-Admin kann Limit pro Key erhöhen** (max. 10.000/h ohne Plattform-Admin-Approval)

Implementation

- Token-Bucket-Algorithmus
- Counter in Redis pro Key + pro Tenant
- Headers in jeder Response:
- `X-RateLimit-Limit: 1000`
- `X-RateLimit-Remaining: 873`
- `X-RateLimit-Reset: 1715169600` (Unix-Timestamp)
- Bei Überschreitung: 429 + `Retry-After`-Header

Klartext-429-Body

```
{
  "error": {
    "code": "rate_limit_exceeded",
    "message": "Rate-Limit erreicht: 1.000 Anfragen/Stunde. Limit setzt sich um [timestamp]
zurück.",
    "details": {
      "limit": 1000,
      "remaining": 0,
      "reset_at": "2026-05-08T11:00:00Z",
      "tenant_admin_can_increase": true
    }
  }
}
```

Metering für Phase 2

- Aggregat-Stats in `/{tenant}/admin/api-keys/{id}/usage/`: Tagesverbrauch, Top-Endpoints, Fehler-Rate
- Phase 2: Pricing-relevante Über-Limits (Cloud+ erlaubt mehr Requests)

Block I — Versionierung & Stabilität

URL-basierte Versionierung

- Aktuelle Version: `v1`
- Pfad: `/api/v1/...`
- Major-Bump (v2) nur bei Breaking Changes

Backwards-Compatibility-Regeln (Phase 1)

- Neue Felder hinzufügen: erlaubt
- Felder entfernen: nur mit Deprecation-Phase + neuer Version
- Field-Type ändern: nur mit neuer Version
- Pflicht-Filter hinzufügen: nur mit neuer Version

Deprecation-Header

- Endpoints im Sunset-Pfad senden `Sunset: <date>` und `Deprecation: true` Header
- Klartext-Hinweis in OpenAPI-Description

Stability-Tags pro Endpoint

- `stable` (Phase 1: alle Read-Endpoints)

- `experimental` (Phase 2-Vorboten — Kein SLA)
- `deprecated` (Sunset angekündigt)

Block J — Permissions / Scope-Mapping

Scope-Mapping zu Endpoints

Endpoint	Min. Scope
`GET /api/v1/contacts*`	`read_only` (sensible Felder ausgeblendet) oder `read_full`
`GET /api/v1/companies*`	`read_only` oder `read_full`
`GET /api/v1/deals*`	`read_only` oder `read_full`
`GET /api/v1/projects*`	`read_only` oder `read_full`
`GET /api/v1/tasks*`	`read_only` oder `read_full`
`GET /api/v1/leads*`	`read_only` oder `read_full`
`POST /api/v1/webhooks/incoming/*`	`webhook_inbound`
`POST /api/v1/integrations/events`	`automation_trigger`
`GET /api/v1/openapi.json`	unauth
`/swagger`	unauth

Sensible Felder bei `read_only`

Werden in Response weggelassen (nicht 403, da Endpoint selbst erlaubt ist):
- `Contact.notes`
- `Contact.custom_fields[where is_sensitive=true]`
- `Deal.notes`
- `Project.notes`
- `Lead.message`

Tenant-Status-Wirkung

- Tenant `frozen` / `past_due` (read-only): GET erlaubt, alle POST/PUT/DELETE → 403 mit Klartext: *„Workspace ist im Read-Only-Modus."*
- Tenant `suspended` / `deleted`: alle Requests → 403 / 410
- Tenant `deletion_pending`: GET erlaubt, POST/PUT/DELETE blockiert

API-Key-Owner-Wirkung

- API-Keys gehören Tenants, nicht Usern (M27)
- Bei Anfrage werden RLS-Policies mit `app.tenant_id = key.tenant_id` gesetzt
- Kein User-Owner-Check (Keys sehen alle Tenant-Daten ihres Scopes)

Activity-Stream-Integration

Activity-Typen die M34 produziert (Audit)

- `api_key_used` (nicht pro Request, sondern bei sicherheitsrelevanten Aktionen — Phase 2)
- `webhook_subscription_created`
- `webhook_subscription_disabled` (auto bei Failure-Threshold)
- `incoming_webhook_received` (nur für `processing_status=done` mit `derived_lead_id` → Activity am Lead)

Verknüpfungen

- Eingehende Webhooks die Leads erzeugen: Activity am Lead mit `summary_text`: *„Lead erzeugt aus Webhook ‚[Slug]' am [Datum]"*

Permission-Wirkung

- Webhook-Subscription-Audits sichtbar mit `audit.read`
- API-Request-Logs sichtbar mit `tenant.admin` oder `integrations.admin` (in `/{tenant}/admin/api-keys/{id}/logs/`)

Klartext-Hinweise/-Fehler (Querschnitt)

Auth-Fehler

- 401 fehlender Header: *„API-Key fehlt. Setze Header:* `Authorization: Bearer hg_<dein_key>`"
- 401 ungültiger Key: *„API-Key ist ungültig oder widerrufen."*
- 403 falscher Scope: *„Dieser API-Key hat Scope* `[X]`*, benötigt wird* `[Y]`*. Tenant-Admin kann Scope ändern."*

Webhook-Fehler

- Eingehender Webhook mit ungültigem Payload: *„Pflichtfeld* `email` *fehlt im Payload. Erwartetes Schema: [Link zur Mapping-Doku]"*
- Ausgehender Webhook deaktiviert: *„Webhook ‚[Name]' ist nach 10 fehlgeschlagenen Auslieferungen deaktiviert. Letzter Fehler: HTTP 502 vom Endpoint. [Reaktivieren]"*

Rate-Limit

- *„Rate-Limit erreicht. Reset um [Zeit]. Tenant-Admin kann Limit erhöhen unter /admin/api-keys/."*

Validation

- *„Filter* `created_at_gte` *muss ISO-8601 sein, z.B.* `2026-01-01` *oder* `2026-01-01T00:00:00Z`*. Erhalten:* `[wert]`*."*

Phase 2 / spätere Features

- **Voll-CRUD-API:** POST/PUT/PATCH/DELETE für alle Hauptentitäten
- **GraphQL-Endpoint** als zweite Schnittstelle (`/api/graphql`)
- **OAuth2-Provider-Modus:** Drittanbieter-Apps können HiveGrid-Daten im Namen des Users abfragen (3-Legged-OAuth)
- **Granulare Scopes pro Modul-Aktion:** z.B. `contacts:read`, `contacts:write`, `deals:read_all`

- **API-Versioning v2** mit Deprecation-Plan für v1
- **Webhook-Replay-Tool** in UI (für `IncomingWebhookIngestion` und `WebhookDelivery`)
- **Bulk-Export-Endpoint** (`POST /api/v1/exports` → asynchroner Job → S3-Pre-Signed-URL)
- **Filter-Engine als URL-Param** (komplexere Filter-Trees per `?filter=...` mit JSON)
- **Rate-Limit-Tier-Boost** über Pricing-Plan (Cloud+ = 5.000/h)
- **API-Key-Rotation-Workflow** (Übergangsfenster mit zwei aktiven Keys)
- **Plattform-Admin-API** (`admin.hivegrid.de/api/admin/...`) für Tenant-Management

Architektur-Vorschriften (Querschnitt)

- **LP-6 Anti-Agentur:** API soll für Noah und „dummen Handwerker" pflegbar sein → klare Klartext-Fehler, Swagger-UI mit Try-it-out
- **LP-9:** Requirements-pur, kein Layout
- **LP-10:** Klartext-Diagnose bei jedem 4xx/5xx
- **LP-11:** Webhook-Subscriptions sind selbst Adapter-konform — externe Tools können sich anschließen ohne Code-Änderung in HiveGrid
- **LP-13:** API-Endpoints laufen auf EU-Servern, Logs in EU-DB
- **AV #5 Permissions-aware-Default:** Sensible Felder werden bei `read_only` automatisch weggelassen
- **AV #6 Audit-Trail:** API-Key-Anlage, Webhook-Setup, alle Lead-Inbound-Events geloggt

---

Detail-Spec-Status: DRAFT

Erste vollständige Detail-Spec für Modul 34. Phase-1-Basic, bewusst Read-only + Webhooks. **Vor Bau:** OpenAPI-Tooling-Auswahl (Zod-zu-OpenAPI vs. tRPC-OpenAPI vs. manuell), Rate-Limiter-Implementierung mit Redis testen, HMAC-Signatur-Format dokumentieren.

Offene Detail-Punkte (nicht-blockierend):
- Konkrete Field-Mapping-DSL für Incoming-Webhooks (Phase 1 möglicherweise nur 1:1, Phase 2 mit Transform-Functions)
- Konkretes Default-Tenant-API-Aggregat-Limit (10.000/h vorgeschlagen — final mit Performance-Tests)
- OpenAPI-Hosting-URL (eigene Subdomain `developers.hivegrid.de` vs. `/swagger` auf app-Domain)
- Subframe-UI-Designs für API-Key-Verwaltung und Webhook-Subscription-View (separat — kommt aus M27)
- Verknüpfung mit Modul 17 Automation-Engine für `external_event`-Trigger-Pfad