immoGenio

No. 01
März 2026

OPEN-BANKING

Open Banking in der Hausverwaltung: PSD2-AIS, finAPI und die 3-stufige Matching-Pipeline

Wie Open Banking nach PSD2 die Mietverwaltung verändert: automatischer Sync, OAuth2-Consent-Lebenszyklus, HMAC-Webhooks und 3-stufiges Auto-Matching von Mieteingängen.

AUTOR
ImmoGenio

VERÖFFENTLICHT
14. März 2026

ONLINE
www.immogenio.de/blog

— AUF EINEN BLICK

Inhalt

01 Was PSD2 ändert — und was nicht

02 Warum der Provider zaehlt

03 Der OAuth2-Consent-Flow im Detail

04 Sync-Architektur: Cron, Locks, Token-Refresh

05 Webhook-Push: Echtzeit statt Tagesversatz

06 Die dreistufige Matching-Pipeline

07 Consent-Renewal: 14 Tage vor Ablauf

08 DSGVO und Datensparsamkeit

09 Praxisbeispiel: vom Mieteingang zum „bezahlt”-Status in vier Sekunden

10 Was Open Banking nicht kann — und was bleibt

11 Wo wir stehen

12 Abschluss

Stellen Sie sich einen Verwalter mit 47 Mietkonten vor, verteilt auf neun WEG, drei Sondereigentumsverwaltungen und einem Pool fremdverwalteter Mietobjekte. Jeden Monatsanfang öffnet der Buchhalter morgens das Online-Banking, klickt sich durch jedes Konto, lädt PDF-Auszüge herunter und übertraegt Mieteingänge in ein Excel mit Spalten für Vertrag, Sollbetrag, Eingangsdatum und Differenz. Drei Stunden Tipparbeit, ein verschmierter Highlighter, fuenf Tippfehler. Am Nachmittag ruft der erste Beirat an: Warum steht die Mieterin im April auf dem Mahnstapel, sie habe doch am 28.03. überwiesen? Der Buchhalter sucht, findet die Buchung, korrigiert das Excel — und der Mahnlauf hat das Schreiben längst rausgeschickt.

Dieses Szenario ist kein Strohmann. Es war über ein Jahrzehnt der Standard in deutschen Hausverwaltungen, und es ist auch heute noch in vielen Büros gelebte Realität. Mit PSD2 und Open Banking gibt es seit 2018 einen rechtlichen Rahmen, der diesen Workflow vollständig automatisierbar macht — wenn er sauber implementiert wird. Dieser Artikel beschreibt, wie ImmoGenio das tut: welche Provider-Entscheidung wir getroffen haben, wie der OAuth2-Consent-Flow technisch ablaeuft, warum Webhook-Push den Unterschied zwischen „Daten von gestern" und „Daten von vor zwei Minuten" macht, und wie eine dreistufige Matching-Pipeline Mieteingänge deterministisch und nachvollziehbar auf Mietverträge zuordnet.

Was PSD2 ändert — und was nicht

Die Zweite Zahlungsdiensterichtlinie (Richtlinie (EU) 2015/2366, kurz PSD2) verpflichtet Kreditinstitute, lizenzierten Drittanbietern (Third Party Provider, TPP) standardisierten programmatischen Zugriff auf Kontodaten zu gewaehren. Die technischen Spielregeln stehen in den Regulatory Technical Standards (RTS), insbesondere Article 32 zur Schnittstellenbereitstellung. In Deutschland ist die Umsetzung im Zahlungsdiensteaufsichtsgesetz (ZAG) verankert: § 1 Abs. 34 ZAG definiert den Kontoinformationsdienst (Account Information Service, AIS), § 51 ZAG regelt die Erlaubnispflicht, die zustaendige Aufsicht ist die BaFin.

Wichtig fuer das Verstaendnis der eigenen Compliance-Lage: AIS ist ein reiner Lesedienst. Ein AIS-lizenzierter Anbieter darf Salden, Umsaetze, Kontostammdaten lesen — aber keine Zahlung initiieren. Letzteres waere PIS (Payment Initiation Service) und braucht eine separate Erlaubnis. Fuer eine Hausverwaltung, die Mieteingaenge automatisch abgleichen will, reicht AIS vollstaendig aus. Wer aktiv Lastschriften einziehen oder Sammelueberweisungen ausloesen will, bleibt im klassischen SEPA-Bereich; dazu mehr in unserem Beitrag zur SEPA-Basislastschrift mit pain.008-Sammelläufen.

Ein zweites zentrales PSD2-Element ist die Strong Customer Authentication (SCA, § 55 ZAG). Fuer AIS-Zugriff schreibt die RTS einen 90-Tage-Refresh vor: Spaetestens alle 90 Tage muss der Kontoinhaber den Zugriff erneut mit Zwei-Faktor-Authentifizierung bestaetigen.

Diese Frist ist keine Empfehlung, sondern eine harte Vorgabe — und sie ist die haeufigste Ursache fuer „warum kommen ploetzlich keine Daten mehr?". Jede saubere Implementierung muss den Consent-Lebenszyklus beobachten und proaktiv auf den Ablauf hinweisen.

Warum der Provider zaehlt

Theoretisch koennten wir uns selbst eine TPP-Lizenz holen und direkt mit den Banken-APIs sprechen. Praktisch waere das fuer eine Verwaltungssoftware grober Unfug. Es gibt mehrere hundert deutsche Banken mit jeweils eigener API-Implementierung, eigenen Edge Cases, eigenen Outage-Mustern. Ein spezialisierter Provider abstrahiert diese Heterogenitaet hinter einer einheitlichen Schnittstelle — das ist der Kern seiner Wertschoepfung.

Wir haben uns fuer finAPI als deutschen Marktteilnehmer entschieden. Die Auswahlkriterien:

- **BaFin-Lizenz fuer AIS und PIS.** Die Lizenz liegt vor, der Anbieter steht unter deutscher Aufsicht.
- **EU-Hosting.** Verarbeitung in Frankfurt, kein Drittlandtransfer, keine Schrems-II-Kopfschmerzen.
- **Auftragsverarbeitungsvertrag nach Art. 28 DSGVO.** Schriftlich abgeschlossen, mit allen Pflichtinhalten, vor produktiver Anbindung.
- **Coverage.** Ueber 3.000 angebundene Banken in Deutschland und der DACH-Region. Fuer eine Hausverwaltung relevant, weil Mandantenkonten quer durch die Sparkassen-, Volksbanken- und Privatbankenlandschaft gestreut sind.
- **Servicequalitaet.** Webhook-Support, dokumentierte SLAs, technischer Ansprechpartner — kein Ticket-only-Support.

Diese Provider-Frage ist der erste, oft unterschaetzte Hebel fuer Datenqualitaet. Eine guenstige API hilft nichts, wenn die Volksbank Mittelhessen-Sparkonten nicht zuverlaessig synchronisiert.

Der OAuth2-Consent-Flow im Detail

Der Zugriff auf ein konkretes Bankkonto laeuft ueber den OAuth 2.0 Authorization Code Flow nach RFC 6749. Wer das Muster aus dem klassischen „Login mit Google" kennt, erkennt die Schritte wieder — nur dass am Ende statt eines Profil-Tokens ein AIS-Token mit 90 Tagen Lebensdauer steht.

Der Ablauf in ImmoGenio:

1. **Initiale Anbindung im Mandantenkontext.** Der Mandanten-Admin (nicht der Verwalter im Namen des Mandanten) navigiert in den Einstellungen auf „Bankkonto verbinden", waehlt seine Bank aus der Liste der angebundenen Institute.
2. **State-Generierung.** ImmoGenio erzeugt einen kryptografischen `state`-Parameter, persistiert ihn in `bank_oauth_states` mit Mandant-ID, Bank-ID und Ablaufzeit (10 Minuten), und leitet den Browser auf die Authorisierungs-URL des Providers um.
3. **SCA bei der Bank.** Der Provider leitet weiter zur Bank-Login-Seite. Hier authentifiziert sich der Mandanten-Admin mit seinen Bank-Credentials und absolviert die SCA — typischerweise per pushTAN, photoTAN oder ChipTAN. Das ist der entscheidende Punkt: Der Verwalter sieht die Credentials nie, ImmoGenio sieht sie nie, der Provider sieht sie nie. Die Authentifizierung passiert ausschliesslich zwischen Mandant und Bank.
4. **Redirect mit Authorization Code.** Nach erfolgreichem SCA leitet die Bank zurueck an den Provider, der wiederum an die ImmoGenio-Callback-URL umleitet. Im Query-String steht der `code` und der zuvor erzeugte `state`.
5. **State-Validierung und Token-Tausch.** ImmoGenio prueft den `state` gegen `bank_oauth_states` (existiert, nicht abgelaufen, nicht bereits eingeloest), tauscht den `code` ueber den Backchannel beim Provider gegen Access- und Refresh-Token.
6. **Verschluesselte Speicherung.** Beide Token werden mit AES-256-GCM verschluesselt und in `mandanten_bankkonten` abgelegt. Der Verschluesselungsschluessel liegt im Server-Secret-Store, nicht in der Datenbank. Selbst bei einem Datenbank-Leak waeren die Tokens ohne den Schluessel wertlos.
7. **Initialer Sync.** Direkt im Anschluss laeuft der erste Sync: Salden, Stammdaten, die letzten 90 Tage Umsaetze. Das passiert synchron, weil der Mandanten-Admin in der UI sehen will, dass es funktioniert hat.

Die Trennung zwischen Authentifizierung (immer beim Mandanten) und Datennutzung (durch den Verwalter) ist nicht nur technisch sauber, sondern entlastet auch haftungsrechtlich. Der Verwalter besitzt keine Bank-Credentials und kann sie nicht missbrauchen — eine Forderung, die jeder Wirtschaftspruefer und jeder DSGVO-Auditor begruessen wird.

Sync-Architektur: Cron, Locks, Token-Refresh

Der Dauerbetrieb des Bankkonten-Sync laeuft nach einem dreigliedrigen Muster: zeitgesteuerter Pull, ereignisgesteuerter Push und manueller Trigger.

Zeitgesteuerter Pull. Ein Cron-Job um 03:30 UTC iteriert ueber alle aktiven `mandanten_bankkonten` und triggert pro Konto einen Sync. Der Zeitpunkt ist kein Zufall: Die meisten deutschen Banken liefern die Tagessalden nach Mitternacht in einem Wartungsfens-

ter. Um 03:30 UTC sind die Daten frisch, und das ist 04:30 oder 05:30 Berliner Zeit — vor dem morgendlichen User-Verkehr.

Race-Conditions vermeiden. Damit ein Konto nicht gleichzeitig von Cron, Webhook und manuellem Trigger gepullt wird, holt sich jeder Sync-Worker per `pg_try_advisory_lock(hashtextextended('bankkonto_sync', kontoid))` ein PostgreSQL-Advisory-Lock. Wer den Lock nicht bekommt, beendet sich kommentarlos — das laeuft sicher der konkurrierende Sync schon. Advisory Locks haben den Charme, dass sie session-gebunden sind und bei Connection-Abbruch automatisch freigegeben werden; eine Stale-Lock-Bereinigung ist nicht noetig.

Token-Refresh. Bei einem 401-Response auf einen API-Call versucht der Worker einmalig, den Access-Token mit dem Refresh-Token zu erneuern, das Ergebnis verschluesselt zurueckzuschreiben und den Call zu wiederholen. Erst wenn der Refresh selbst fehlschlaegt — typischerweise weil der 90-Tage-Consent abgelaufen ist — wird das Konto auf `status = 'consent_expired'` gesetzt und der Renewal-Reminder-Workflow ausgeloest.

Retry mit Exponential Backoff. Transiente Fehler (5xx, Timeouts, Rate-Limits) werden mit drei Versuchen und Backoff 1s, 4s, 16s wiederholt. Persistente Fehler werden im `sync_error`-Feld gespeichert und im Mandanten-Cockpit als roter Indikator sichtbar — keine stille Schluckspecht-Logik.

Webhook-Push: Echtzeit statt Tagesversatz

Der Cron-basierte Pull liefert Daten von vor wenigen Stunden. Fuer einen Mahnlauf, der morgens um 09:00 startet, reicht das. Fuer den Buchhalter, der nachmittags um 14:00 wissen will, ob die Mieterin Schmidt heute Vormittag ueberwiesen hat, nicht mehr.

Der Provider liefert deshalb Webhook-Events. Beim `NEW_TRANSACTIONS`-Event wird ein HTTPS-POST an unseren Webhook-Endpunkt geschickt, mit Header `X-Signature` (Format `sha256=<hex>`). ImmoGenio verarbeitet das Event nach folgender Disziplin:

- **HMAC-Verify.** Der Body wird mit dem geteilten Webhook-Secret per HMAC-SHA256 hashbar gemacht und gegen die Signatur verglichen. Der `sha256=`-Prefix wird toleriert (mit und ohne), weil verschiedene Provider-Versionen mal mit, mal ohne Prefix senden — das ist eine Falle, in die man genau einmal tappt.
- **Timestamp-Pruefung.** Der `X-Timestamp`-Header darf maximal fuenf Minuten in der Vergangenheit liegen. Aeltere Events werden verworfen, das schuetzt gegen Replay-Angriffe.
- **Idempotenz.** Jedes Event hat eine `event_id`. Vor der Verarbeitung wird `bank_webhook_events` per `INSERT ... ON CONFLICT (provider, event_id) DO NOTHING` gepruefte; bei Konflikt antwortet der Endpunkt mit 200 und tut nichts weiter. Provider

duerfen Events doppelt zustellen, das ist Standardverhalten — die Empfaengerseite muss damit umgehen.

- **Schnelle Antwort.** Der Webhook-Handler antwortet binnen 200 ms mit 200 OK und schiebt die eigentliche Sync-Arbeit in eine Job-Queue. Wer den Provider warten laesst, riskiert Retries und Schluesselrotation auf der Provider-Seite.

In der Praxis bedeutet das: Eine Mieteingangs-Buchung, die morgens um 09:14 in der Bank verbucht wird, ist binnen Sekunden in ImmoGenio sichtbar. Der Buchhalter muss nicht refreshen, das Cockpit wird per Realtime-Push aktualisiert. Fuer Cashflow-Transparenz ist das ein qualitativer Sprung.

Die dreistufige Matching-Pipeline

Daten in der Datenbank zu haben ist die halbe Miete. Die andere Haelfte ist: einen eingehenden Betrag deterministisch dem richtigen Mietvertrag zuordnen, ohne menschliche Klickarbeit, aber mit nachvollziehbarer Audit-Spur. ImmoGenio nutzt dafuer eine dreistufige Matcher-Pipeline. Jeder eingehende Datensatz aus `bank_transaktionen` durchlaeuft die Stufen in fester Reihenfolge; sobald eine Stufe matched, wird abgebrochen.

Stufe 1: Verwendungszweck-Nummer (Konfidenz 0.95)

Wenn ein Mietvertrag eine eindeutige Referenznummer traegt, die der Mieter im Verwendungszweck mitschickt, ist das Match deterministisch. Die Pipeline parst den Verwendungszweck per Regex auf bekannte Praefixe (`MV-2024-0042`, `MNR-12345`, `Vertrag 7711`), schlaegt in `mietvertraege` nach und vergibt Konfidenz 0.95. Das ist der Goldstandard, weil er gegen alle uebrigen Stoerquellen — falsche IBAN, geringfuegig anderer Betrag, geaenderter Mietername durch Eheschliessung — robust ist. Voraussetzung ist allerdings, dass die Mieter die Referenznummer wirklich angeben. In Bestandsverwaltungen ist das selten flaechig der Fall; bei Neuvermietungen sollte die Referenz im Mietvertrag und im Begruessungsschreiben prominent stehen.

Stufe 2: IBAN-Exakt-Match (Konfidenz 0.92)

Wenn Stufe 1 nicht greift, prueft die Pipeline die Absender-IBAN gegen die in `mieter_bankkonten` hinterlegten Mieter-IBAN. Bei Treffer und passender Vertragszuordnung des Mieters wird Konfidenz 0.92 vergeben. Diese Stufe ist robust gegenueber Tippfehlern im Verwendungszweck und gegen vergessene Referenznummern, scheitert aber bei Mietern, die ihr Bankkonto wechseln, oder wenn Ehepartner ueber unterschiedliche Konten zahlen. Die `mieter_bankkonten`-Tabelle erlaubt deshalb mehrere IBAN pro Mieter und pflegt eine Aktivitaets-Markierung, sodass alte IBAN ohne Vertrags-Bezug nicht mehr matchen.

Stufe 3: Betrag±1 Euro plus Jaccard-Name (Konfidenz 0.6 bis 0.9)

Die schwierigste Stufe ist die Fuzzy-Logik fuer alles, was die ersten beiden Stufen nicht gefangen haben. Die Pipeline filtert zunaechst alle aktiven Mietvertraege, deren naechste Sollstellung im Toleranzfenster `Betrag ± 1 EUR` liegt — der Euro Toleranz faengt Cent-Differenzen aus Indexmietanpassungen und Ueberzahlungen ab. Auf der verbleibenden Kandidatenliste laeuft ein Jaccard-Index zwischen dem Buchungstext (Absendername aus dem Bankdatensatz) und dem Mietername aus dem Vertrag.

Der Jaccard-Index zerlegt beide Strings in normalisierte Tokens (lowercase, Umlaut-Replacement, Stopwort-Filter) und berechnet `|A ∩ B| / |A ∪ B|`. Ein Score von 1.0 bedeutet Token-Identitaet, 0.0 keine gemeinsamen Tokens. Die Konfidenz-Berechnung in Stufe 3 kombiniert Betrags-Naehe und Jaccard zu einem Wert zwischen 0.6 und 0.9.

Die Anwendung der Schwellenwerte:

- **≥ 0.9: Auto-Match.** Das System bucht den Eingang automatisch auf den Vertrag.
- **0.6 bis 0.9: Vorschlag.** Das System hinterlegt den Match als Vorschlag, der Buchhalter bestaetigt mit einem Klick.
- **< 0.6: kein Match.** Der Eingang bleibt im „nicht zugeordnet”-Stapel.

Eine zusaetzliche Konfliktregel verhindert Fehlbuchungen: Wenn zwei oder mehr Mietvertraege mit weniger als 0.05 Konfidenz-Abstand auf denselben Eingang matchen, gibt es kein Auto-Match — die Pipeline meldet einen Konflikt und legt den Fall dem Buchhalter vor. Das ist der typische Fall, wenn zwei Mieter mit aehnlichem Namen denselben Mietbetrag in derselben WEG zahlen.

Inkrementelles Matching

Der Matcher laeuft nicht jede Nacht ueber die Vollmenge aller Buchungen, sondern bekommt vom Sync-Worker explizit nur die IDs der gerade neu importierten Transaktionen. Bei einem Sync, der 12 neue Buchungen liefert, laeuft der Matcher genau 12 Mal — nicht 12.000 Mal. Das hat zwei Vorteile: Performance bleibt linear in der Tagesmenge, und ein einmal manuell korrigiertes Match wird nicht beim naechsten Lauf wieder „verschlimmbessert”. Wer das Muster fuer andere Datenstroeme adaptieren will, findet eine breitere Diskussion zur API-Strategie in unserem Beitrag zum offenen API-Schnittstellen-Ecosystem.

Consent-Renewal: 14 Tage vor Ablauf

Die 90-Tage-SCA-Pflicht ist die haeufigste operative Bruchstelle. ImmoGenio schreibt deshalb in `mandanten_bankkonten` das Feld `consent_expires_at` und einen separaten Reminder-Tracker `consent_reminder_sent_at`.

Ein taeglicher Job prueft alle aktiven Konten:

- **14 Tage vor Ablauf**: Erste E-Mail an den Mandanten-Admin, Betreff „Bankzugriff laeuft in 14 Tagen ab — Verlaengerung erforderlich". Der Reminder enthaelt einen direkten Link in den OAuth-Start-Flow, sodass der Admin ohne Umweg ueber das Cockpit re-authentifizieren kann.
- **Maximal ein Reminder pro 7 Tage**: `consent_reminder_sent_at` verhindert, dass der Admin taeglich angeschrieben wird. Das Feld wird beim erfolgreichen Re-Consent geleert.
- **Bei Ablauf**: Konto-Status wechselt auf `consent_expired`, im Cockpit erscheint ein roter Banner, der Sync ist pausiert. Der Admin kann jederzeit ueber den OAuth-Start-Flow den Consent erneuern; der naechste erfolgreiche Sync setzt den Status auf `active` zurueck.

Wichtig: Wer den 14-Tage-Vorlauf verkuerzt, riskiert Wochenend-Ausfaelle. Wer ihn auf 30 Tage verlaengert, generiert Reminder-Fatigue. Vierzehn Tage haben sich als praktikable Balance erwiesen.

DSGVO und Datensparsamkeit

PSD2 erlaubt den Zugriff — die DSGVO regelt, was wir damit duerfen. ImmoGenio speichert pro Buchung ausschliesslich die Felder, die fuer den Zahlungsabgleich, die Buchhaltung und die handelsrechtliche Aufbewahrung erforderlich sind:

- Buchungstag und Valuta
- Betrag und Waehrung
- Verwendungszweck (vollstaendig)
- Absender-IBAN und Absender-Name
- Bank-Referenz fuer ISO 20022 camt.052/053-Kompatibilitaet

Bewusst nicht gespeichert wird, was der Provider technisch zwar liefern koennte, aber fuer den Verwendungszweck nicht erforderlich ist:

- Geo-Daten und Geraete-Fingerprints
- Branchenklassifikationen und Kategorisierungen, die der Provider mit eigenen ML-Modellen anreichert
- Kontostammdaten anderer Konten desselben Inhabers

Diese Datensparsamkeit ist nicht nur Art. 5 Abs. 1 lit. c DSGVO geschuldet. Sie reduziert auch die Angriffsflaeche im Falle eines Sicherheitsvorfalls und macht die Datenschutz-Folgenabschaetzung wesentlich entspannter. Der Auftragsverarbeitungsvertrag mit dem Pro-

vider regelt parallel, dass der Provider eigene Anreicherungsdaten nicht ohne separaten Auftrag persistiert.

Praxisbeispiel: vom Mieteingang zum „bezahlt"-Status in vier Sekunden

Mieterin Schmidt zahlt am 1. Mai um 09:14 Uhr per Sofortueberweisung 1.250,00 EUR an das WEG-Mietkonto. Der Ablauf:

1. **09:14:31** Bank verbucht den Eingang.
2. **09:14:42** Provider erhaelt Push von der Bank, generiert `NEW_TRANSACTIONS`-Event, sendet Webhook.
3. **09:14:43** ImmoGenio empfaengt den Webhook, validiert HMAC-Signatur und Timestamp, persistiert Event in `bank_webhook_events`, schiebt Sync-Job in die Queue, antwortet 200.
4. **09:14:44** Sync-Worker zieht Job, holt Advisory-Lock auf das Konto, ruft die Tx-Liste vom Provider ab, persistiert die neue Buchung in `bank_transaktionen`, gibt Lock frei, ruft Matcher mit der Tx-ID auf.
5. **09:14:45** Matcher-Stufe 1: Verwendungszweck enthaelt `MV-2024-0042` — Treffer mit Konfidenz 0.95. Auto-Match ueber Schwellenwert, Sollstellung wird auf `bezahlt` gesetzt.
6. **09:14:46** Realtime-Push aktualisiert das Cockpit des Buchhalters; das Mietkonto Schmidt zeigt gruen.

Vier Sekunden vom Banking-System bis zum gruenen Haken. Wer das einmal live gesehen hat, geht nicht mehr zurueck zum Excel.

Diese Echtzeit-Sicht greift mit dem dreistufigen Mahnwesen und den Verzugszinsen nach § 288 BGB sowie dem WEG-Wirtschaftsplan und der Sollstellung nach § 28 WEG ineinander: Wer in Echtzeit sieht, was hereinkommt, kann das Mahnwesen ehrlich fahren und den Wirtschaftsplan-Status fuer die Eigentuemerversammlung tagesaktuell darstellen.

Was Open Banking nicht kann — und was bleibt

Damit die Erwartungen kalibriert sind: AIS deckt das Lesen ab, nichts darueber hinaus. Konkret nicht enthalten:

- **PIS — Payment Initiation.** Wir loesen keine Zahlungen aus. Sammellaeufe, Lastschriftrueckgaben, Ueberweisungen laufen weiterhin ueber SEPA-XML im klassischen pain.001/pain.008-Format.

- **Multi-Banking-Aggregation als verkaufter Service.** Wir aggregieren Konten innerhalb eines Mandanten, vertreiben aber kein Multi-Banking-Frontend an Endkunden.
- **SCHUFA- oder Bonitaetsabfragen.** Open Banking liefert keine Bonitaetsauskunft; das ist regulatorisch und vertraglich ein eigener Dienst.
- **Steuerliche Klassifizierung.** Der Provider kategorisiert Buchungen mit eigenen Modellen — ImmoGenio nutzt diese Klassifikation bewusst nicht. Steuerliche Zuordnung passiert auf Buchungskreis-Ebene in den Buchhaltungs-Workflows, nicht auf der Banking-Schicht.
- **Ersatz fuer Bank-Belege.** Eine Banking-API-Buchung ist kein Originalbeleg im Sinne der GoBD. Fuer eingehende Rechnungen bleibt der strukturierte XML-Beleg die Grundlage; siehe dazu unseren Beitrag zum E-Rechnungs-In-Flight-Swap zwischen ZUGFeRD und XRechnung.

Wo wir stehen

Die finAPI-Anbindung ist produktiv. Der OAuth2-Consent-Flow, der naechtliche Sync, der Webhook-Empfang, die dreistufige Matcher-Pipeline und der Consent-Renewal-Reminder laufen seit mehreren Monaten in der Produktion. Die zugrundeliegenden Datenbankmigrationen 088 bis 092 (Tabellen `mandanten_bankkonten`, `bank_oauth_states`, `bank_transaktionen`, `bank_webhook_events`, Feld `consent_reminder_sent_at`) sind ausgerollt, AES-256-GCM-Verschluesselung der Tokens ist aktiv, der Cron-Lauf um 03:30 UTC ist stabil. Beobachtet werden weiterhin die Auto-Match-Quote pro Mandant, die Konflikt-Rate in Stufe 3 und die Consent-Renewal-Erfolgsquote — mit dem Ziel, Auto-Match dauerhaft jenseits der 80-Prozent-Marke zu halten.

Abschluss

Fragen, Rueckmeldungen oder Hinweise zu eigenen Banking-Edge-Cases gehen am besten direkt an kontakt@immogenio.de. Besonders interessant sind Faelle mit unueblichen Verwendungszweck-Konventionen, ausgefallenen IBAN-Wechsel-Situationen und Banken, deren PSD2-Schnittstelle in der Praxis Eigenheiten zeigt. Jede gemeldete Schwachstelle landet in der naechsten Iteration der Matcher-Pipeline.