

REST API

Billing REST API Handbook

A practical cookbook for the current local Rails implementation - authentication, safe writes, core billing flows and explicit limits.

154 / 159

OpenAPI operations exercised through successful local response wiring

1091

tests in the captured local suite

0

live provider, bank, SMTP or production calls in this guide

Scope

This handbook documents the current local Rails API and selected cookbook flows. It is not a Frisbii manual and does not claim complete semantics for all 159 source operations.

1 - CONTRACT AND EVIDENCE

Read this before the first request

The API is a merchant-scoped local implementation that validates requests and responses against the checked-in Frisbii Transform OpenAPI snapshot with separately documented, provenance-backed response corrections. The source snapshot contains 121 paths, 159 HTTP operations and 231 schemas. Route presence alone is not implementation proof.

Current runtime evidence	154 exact operations completed ApiSpec response wiring in the captured suite; 5 did not.
Evidence meaning	validated_response_exercised is a local lower bound. It does not establish complete endpoint semantics or vendor parity.
Canonical field style	JSON request and response properties use the OpenAPI PascalCase names, for example RequirePayment, CompanyName and DateRange.
Tenant boundary	The authenticated client selects one merchant. Cross-merchant resource identifiers return 404.
External effects	Payment providers, SMTP delivery and live webhook destinations remain disabled until separately configured and rehearsed.

Use the live evidence

Open </docs> for the searchable operation inventory, </openapi.json> for the checked-in schema, and [docs/compatibility/runtime-coverage.md](/docs/compatibility/runtime-coverage.md) for the exact exercised and unexercised operation list.

Base URL used in examples

Examples use <https://billing.example.invalid>, a non-routable documentation host. Replace it with the operator-provided base URL. Placeholder identifiers such as CUSTOMER_ID are synthetic and must be replaced with IDs returned inside the same merchant scope.

2 - ACCESS

Authentication and authorization

Client credentials are implemented locally

POST /oauth/token accepts the client_credentials grant. Client credentials can be sent as form fields or HTTP Basic authentication. A successful local-mode response returns a bearer token with a 3600-second lifetime and Cache-Control: no-store.

TOKEN REQUEST

```
curl -sS -X POST 'https://billing.example.invalid/oauth/token' \
-H 'Content-Type: application/x-www-form-urlencoded' \
--data-urlencode 'grant_type=client_credentials' \
--data-urlencode 'client_id=YOUR_CLIENT_ID' \
--data-urlencode 'client_secret=YOUR_CLIENT_SECRET'
```

LOCAL-MODE RESPONSE SHAPE

```
{
  "access_token": "ACCESS_TOKEN",
  "token_type": "bearer",
  "expires_in": 3600
}
```

In Keycloak API mode, the same local endpoint validates that the client is active, restricts requested scopes to the local grant, and exchanges credentials with the configured Keycloak token endpoint. Admin browser sessions use a separate OIDC authorization-code flow.

Scopes

- GET requests require billing.read, except GET selfservicetoken, which creates a bearer capability and requires billing.write.
- POST, PUT, PATCH and DELETE requests require billing.write.
- Keycloak identities with more than one merchant grant must select exactly one authorized merchant with X-Merchant-Id.

Credential handling

Do not log access tokens, client secrets, self-service tokens or signed download URLs. A missing or invalid token returns 401 with WWW-Authenticate: Bearer realm="billing".

3 - HTTP RULES

Request conventions

Authorization	Authorization: Bearer ACCESS_TOKEN on every /api/v1 request.
JSON bodies	Send Content-Type: application/json when a body is present. Other body media types return 415.
Body limit	1 MiB, enforced before Rails JSON parsing. Oversized bodies return 413.
Query limit	16 KiB for the raw query string. Oversized queries return 414.
Authorization limit	16 KiB. Oversized authorization input is rejected as unauthorized.
Money	JSON numbers are parsed through BigDecimal and stored canonically with explicit currency.
Dates	Use the exact schema format. DateRange.ExclusiveEnd is excluded from the interval.
Unknown fields	Strict DTOs reject properties that the OpenAPI schema marks as additionalProperties: false.

Idempotency-Key

Mutations accept an optional printable ASCII Idempotency-Key of 1 to 128 characters. Use one stable key for each logical write. Replaying the same actor, method, path, body and query returns the saved result; reusing the key with different content returns 409 IdempotencyConflict.

If-Match

If-Match checks PUT/PATCH operations whose path parameter is {id}, plus POST /api/v1/emails/{id}/resend. Use the quoted ETag; a changed version returns 412. For resend, matching idempotent replay precedes the version check; fresh writes lock the email through mutation. Other POST operations and parameters such as {contractId} or {orderId} do not gain this guard.

GUARDED {ID} MUTATION HEADERS

```
PUT /api/v1/customers/CUSTOMER_ID
Authorization: Bearer ACCESS_TOKEN
Content-Type: application/json
Idempotency-Key: customer-update-2026-09-06-001
If-Match: "3"
```

4 - MERCHANT DATA

Customer cookbook

The following create/read flow is exercised through the authenticated HTTP controller, response schema wiring and tenant isolation checks.

CREATE CUSTOMER - 201

```
curl -sS -X POST 'https://billing.example.invalid/api/v1/customers' \
-H 'Authorization: Bearer ACCESS_TOKEN' \
-H 'Content-Type: application/json' \
-H 'Idempotency-Key: customer-acme-001' \
--data '{
  "ExternalCustomerId": "crm-acme-001",
  "CompanyName": "Example Customer GmbH",
  "EmailAddress": "billing@example.invalid",
  "Address": {
    "Street": "Example Street", "HouseNumber": "1",
    "PostalCode": "10115", "City": "Berlin",
    "Country": "DE"
  }
}'
```

READ CUSTOMER - 200 OR 404

```
curl -sS \
-H 'Authorization: Bearer ACCESS_TOKEN' \
'https://billing.example.invalid/api/v1/customers/CUSTOMER_ID'
```

Lists and filters

GET /api/v1/customers exposes the query fields declared by its operation, including search, statusFilter, externalId, from, skip, take and CustomFields. The current customer/contract/order implementation defaults take to 200 and caps it at 500. Catalog lists use their own limits, generally take up to 1000. Consult the exact operation before sending a filter.

Deletion is guarded

A customer with retained contracts cannot be deleted. Imported read-only merchants and records reject mutation until an explicit audited activation.

5 - SUBSCRIPTION FLOW

Order preview, create and commit

Preview is non-persistent. Create reserves an order and returns 201. Commit is a distinct transition and must use a separate stable idempotency key. The Cart must reference catalog records from the same merchant.

PREVIEW - 200, NO DURABLE ORDER

```
POST /api/v1/orders/preview
Authorization: Bearer ACCESS_TOKEN
Content-Type: application/json

{
  "Cart": { "PlanVariantId": "PLAN_VARIANT_ID" },
  "Customer": { "CompanyName": "Example Customer GmbH" }
}
```

CREATE - 201

```
POST /api/v1/orders
Authorization: Bearer ACCESS_TOKEN
Content-Type: application/json
Idempotency-Key: order-create-001

{
  "Cart": { "PlanVariantId": "PLAN_VARIANT_ID" },
  "CustomerId": "CUSTOMER_ID"
}
```

COMMIT - 200

```
POST /api/v1/orders/ORDER_ID/commit
Authorization: Bearer ACCESS_TOKEN
Content-Type: application/json
Idempotency-Key: order-commit-001

{ "RequirePayment": false }
```

RequirePayment is PascalCase exactly as defined by OrderCommitDTO. If RequirePayment is true and payment data is required, the local implementation can return 202 with PaymentPending and does not create the contract. Raw PAN, CVV and card-number fields are rejected.

No blind retry

Retry a commit only with the same Idempotency-Key and identical content. A transport error or unknown provider outcome never authorizes a second logical payment or commit.

6 - BILLING RECORDS

Contracts and invoice documents

Contract lifecycle

GET /api/v1/contracts/{id} is exercised locally. Pause, resume and end are separate operations with preview or effective-time rules. Financial documents, ledger entries and finalized snapshots are immutable; corrections create linked records rather than overwriting history.

CONTRACT READ

```
GET /api/v1/contracts/CONTRACT_ID
Authorization: Bearer ACCESS_TOKEN
Accept: application/json
```

Build and finalize a manual invoice

CREATE INVOICE DRAFT - 200

```
POST /api/v1/contracts/CONTRACT_ID/invoicebuilder
Authorization: Bearer ACCESS_TOKEN
Content-Type: application/json
Idempotency-Key: invoice-draft-001
```

```
{
  "BillingDate": "2026-09-06T10:00:00Z",
  "ItemList": [{
    "Description": "Synthetic service",
    "Quantity": 3,
    "PricePerUnit": 0.10,
    "TaxPolicyId": "TAX_POLICY_ID"
  }]
}
```

FINALIZE - 200

```
POST /api/v1/invoicedrafts/DRAFT_ID
Authorization: Bearer ACCESS_TOKEN
Content-Type: application/json
Idempotency-Key: invoice-finalize-001
```

```
{}
```

Finalizing the same draft is repeatable and returns the same invoice. GET /api/v1/invoices/{id}/download returns the stored PDF bytes. Download-link endpoints issue short-lived signed local URLs; treat them as bearer capabilities.

Dedicated interim billing

POST /api/v1/contracts/{contractId}/bill bills only eligible unbilled metered or rated usage and eligible frozen component-change fees. It excludes the base recurring fee, marks only selected rows as billed, and does not advance NextBillingDate or the recurring cadence. An empty selection returns 422 nothing_to_bill. See docs/guides/contract-compatibility.md for trial, opt-in and remaining proration limits.

7 - IMMUTABLE HISTORY

Contract change history

New local contracts capture immutable typed lifecycle state beginning with Signup; Signup has an After state and no Before state. GET /api/v1/contractchanges returns the core chronology; GET /api/v1/contractchanges/{contractChangeId} projects the stored snapshot. Historical imports and legacy entries cannot be reconstructed and return 422 contract_change_history_unsupported whenever a selected list page or detail contains incomplete history.

LIST CORE HISTORY

```
GET /api/v1/contractchanges?contractId=CONTRACT_ID&take=50
Authorization: Bearer ACCESS_TOKEN
Accept: application/json
```

Results are ordered by Timestamp descending, then ID descending. The optional from parameter is a raw 24-character change ID and includes its anchor row. Drop that duplicate when continuing a page. These are read-committed pages, not a frozen multi-page snapshot; there is no proprietary cursor header.

EXPANDED DETAIL (JOIN QUERY ONTO ONE URL)

```
GET /api/v1/contractchanges/CHANGE_ID?includeContract=true
&includeComponentSubscriptions=All
&includeDiscountSubscriptions=All&includePauses=All
Authorization: Bearer ACCESS_TOKEN
```

includeContract defaults to false for lists and true for detail, and controls both Contract and PricePhases. Subscription and pause expansions accept None, Changed or All. Limits: take 1-500 (default 200), stored envelope 4 MiB, expanded response 8 MiB. Legacy coverage is explicit; the server never invents earlier state from today's records.

Scheduled changes

Scheduled phase and pause changes activate atomically with their history entry. Due transitions are drained before later visible mutations, with a 100-transition foreground cap; an independent bounded job handles idle contracts. A failed activation rolls back domain state and typed history together while retaining its bounded failure claim and audit result. Trial, pause and terminal statuses have explicit local guards.

Compatibility boundary

These local invariants have independent tests and review. They do not prove every Transform lifecycle scenario. See docs/guides/contract-history.md for exact filters, family limits and operational boundaries.

8 - SNAPSHOTS

Reports and accounting exports

Reports

Implemented report profiles are Contracts, Invoices, Orders and TransactionsFiltered. Formats are Json, Csv and JsonLines. Other official ReportType values return ReportUnsupported until their business meaning exists locally.

CREATE IMMUTABLE REPORT - 200

```
POST /api/v1/reports/Invoices
Authorization: Bearer ACCESS_TOKEN
Content-Type: application/json
Idempotency-Key: report-invoices-september
```

```
{
  "Format": "JsonLines",
  "DateRange": {
    "Start": "2026-09-01",
    "ExclusiveEnd": "2026-10-01"
  }
}
```

A report stores the projected source rows and exact output bytes before returning Finished. Limits fail with 422 before persistence: at most 3,660 days for an explicit range, 50,000 scanned source rows, 10,000 matched rows and 10 MiB output. Narrow or split an oversized report; results are never silently truncated.

Admin CSV snapshots

Browser administrators can export filtered customer, order, payment and refund lists as immutable CSV reports. Customer exports support de-DE/en-US; order, payment and refund exports use the observed German 23-, 19- and 4-column formats. Limits are 10,000 matching rows (5,000 refunds) and 10 MiB; pagination does not limit the export.

For log CSV, apply both dates and filters before exporting. Selection uses merchant-local creation dates, or the refund TimeStamp; exported timestamps use UTC. Text is formula-protected. These reports require authenticated Admin downloads. API-client credentials and bearer links cannot download them. Repeating an idempotency key returns the same file.

The Admin document list also downloads original PDFs as a ZIP. Apply document dates and search first. Local limits: 1,000 PDFs and 64 MiB. Missing or corrupt originals abort the download. The application host requires /usr/bin/zip.

Accounting exports

The source enum names Datev, Lexware, BillwerkSap, DatevExtf, Ewe and Pons. Only DatevExtf is implemented. It requires explicit consultant, client, fiscal-year, account-length and mapping settings. Output remains a local compatibility implementation until a licensed DATEV import test is authorized and passes.

Snapshot rule

Direct report and accounting-export downloads return stored immutable bytes. Later invoices or postings do not change an existing snapshot.

9 - EXTERNAL EDGES

Portal, webhooks and payment boundaries

Self-service

GET /api/v1/contracts/{contractId}/selfservicetoken requires billing.write because it creates a bearer capability. The customer portal uses its own SelfService token scheme and requires Idempotency-Key for mutations. Do not place portal or signed-download tokens in browser storage, analytics, screenshots or logs.

Webhooks

Webhook administration stores merchant-scoped local configuration. Delivery is opt-in and disabled by default. The worker signs bounded payloads, applies destination and network checks, records attempts, and uses retries, but no live receiver success is claimed by this handbook.

Payment transaction reads

GET /api/v1/paymenttransactions returns safe transaction DTOs. Filters: status, inclusive dateFrom, exclusive dateTo and excludePreauth. Pagination: take=200 by default, maximum 500; skip up to 1000. Search and from remain explicit 422 boundaries. POST /api/v1/paymenttransactions/downloadlink creates a separate public CSV snapshot with required dates and returns 201 with a short-lived link. Nonempty search returns 422. This local CSV profile excludes private PSP errors; it does not expose Admin CSV reports. See /docs for limits and compatibility boundaries.

Payments and email

- No example contains PAN, CVV, IBAN, mandate, payment bearer or provider token data.
- POST /api/v1/contracts/{contractId}/payment records an external payment or refund in the local ledger; it never initiates a bank or PSP transaction and needs no provider adapter.
- Provider-executing payment operations require an explicitly configured adapter. Missing configuration fails honestly instead of returning paid or successful state.
- SMTP delivery is disabled until configured and rehearsed. Local email records do not prove external delivery.
- Refunds, external subscriptions and provider-specific report profiles remain bounded by their explicit local implementation status.

Production activation

Provider credentials, Keycloak client registrations, SMTP, DATEV profiles and live webhook destinations require separate operator authorization and environment-specific rehearsal.

10 - OPERATIONS

Errors, retries and implementation limits

400	Malformed JSON, invalid request schema, required parameter or idempotency key.
401 / 403	Invalid API identity, or an authenticated identity lacks the required scope.
404	Operation or merchant-scoped resource is not visible.
409 / 412	State or idempotency conflict, or an If-Match revision no longer matches.
413 / 414 / 415	Body too large, query too large or unsupported body media type.
422	A domain rule, configuration requirement or report/export limit failed.
501	The operation exists in the source contract but local semantics are not implemented.
5xx	Internal failure. Preserve the request identity and investigate before retrying a side effect.

ERROR ENVELOPE

```
{
  "ErrorCode": "IdempotencyConflict",
  "ErrorMessage": "Key was used for a different request"
}
```

Retry checklist

- Reuse the original Idempotency-Key only for the identical logical request.
- Refresh a resource after 412 and decide again; do not overwrite blindly.
- After 5xx, back off and confirm current resource state before another mutation.
- Treat 202 PaymentPending as unfinished. It is not a contract, payment or invoice success.

11 - HONEST STATUS

What is implemented - and what is not

The current runtime artifact was produced by 1091 local tests with 16177 assertions and no failures or errors. It observed 1088 successful response-wiring events across 154 unique OpenAPI operations. 5 exact operations were not observed in that run.

Implemented core	Merchant-scoped catalog, customer, contract, order, subscription, usage, price-list, invoice, report, selected accounting and administration flows.
Deliberately bounded	Recurring and interim billing, webhook delivery, signed downloads, report/export sizes, request sizes and imported read-only data.
Still incomplete	Write-off allocation and period rules; advanced dunning and live delivery; endpoint-specific semantics, full GUI coverage and provider activation.
Not inferred	Frisbii OAuth parity, PSP behavior, legal invoice coverage for every jurisdiction, licensed DATEV compatibility or external delivery success.
Evidence files	docs/compatibility/runtime-coverage.json and runtime-coverage.md preserve exact operation IDs, statuses and named local tests without request payloads or business identifiers.

Decision rule

Use only operations whose current local behavior, validation and side effects meet your workflow. An OpenAPI row, routed handler family or visible screen is never enough by itself.

Local verification

REBUILD EVIDENCE AND HANDBOOK

```
ruby script/api_test_coverage.rb --run
bundle exec ruby script/build_api_handbook.rb
```

This document was generated from the checked-in Rails code, vendored OpenAPI snapshot and local runtime-coverage artifact. It makes no production or external-provider call.