

API Reference

Complete endpoint reference for the LedgerOne NestJS backend — every module, every route, with request and response samples, for frontend integration.

BASE URL (PRODUCTION)

<https://api.aarthalabs.com/api>

AUTH

Bearer JWT + rotating session nonce (see Conventions)

MODULES DOCUMENTED

20

BASE URL (LOCAL DEV)

<http://localhost:3051/api>

FORMAT

JSON over HTTPS, envelope response shape

PREPARED

29 July 2026

Contents

Conventions & Authentication

01 · Auth & Sessions (13 endpoints)

POST /auth/register
POST /auth/login
GET /auth/verify-email
POST /auth/refresh
POST /auth/logout
POST /auth/forgot-password
POST /auth/reset-password
GET /auth/social/:provider/url
POST /auth/social/:provider/callback
GET /auth/me
GET /auth/me/privacy-summary
GET /auth/me/data-export
PATCH /auth/profile
DELETE /auth/me

02 · Two-Factor Auth (3 endpoints)

POST /auth/2fa/generate
POST /auth/2fa/enable
DELETE /auth/2fa/disable

03 · Accounts (5 endpoints)

GET /accounts
POST /accounts/link
POST /accounts/manual
POST /accounts/balances
PATCH /accounts/:id/ownership

04 · Transactions (12 endpoints)

GET /transactions
POST /transactions/import
POST /transactions/import/preview
POST /transactions/import/batch
POST /transactions/sync
POST /transactions/manual
GET /transactions/summary
GET /transactions/cashflow
GET /transactions/merchants
PATCH /transactions/:id/derived
GET /transactions/:id/comments
POST /transactions/:id/comments

05 · Rules Engine (5 endpoints)

GET /rules
POST /rules
PUT /rules/:id
POST /rules/:id/execute
GET /rules/suggestions

06 · Exports (6 endpoints)

GET /exports/csv
GET /exports/json
GET /exports/xlsx
GET /exports/pdf
GET /exports/download/:token
POST /exports/sheets

07 · Plaid (5 endpoints)

POST /plaid/link-token
POST /plaid/exchange
POST /plaid/update-link-token
POST /plaid/repair-complete
POST /plaid/webhook

08 • Teller (3 endpoints)

GET /teller/config
POST /teller/enrollment
POST /teller/sync

09 • Households (Couples) (19 endpoints)

GET /households
POST /households
GET /households/:id
GET /households/:id/dashboard
PATCH /households/:id/privacy
GET /households/:id/members
PATCH /households/:id/members/:memberId
GET /households/:id/invites
POST /households/:id/invites
POST /households/invites/accept
GET /households/:id/goals
POST /households/:id/goals
PATCH /households/:id/goals/goalId
POST /households/:id/fair-split
POST /households/:id/debt-payoff
POST /households/:id/future-scenarios
GET /households/:id/money-date-prompts
GET /households/:id/accountant-tasks
POST /households/:id/accountant-tasks

10 • Planning (11 endpoints)

GET /planning/budgets
POST /planning/budgets
PATCH /planning/budgets/:id
GET /planning/goals
POST /planning/goals
PATCH /planning/goals/:id
GET /planning/investments
POST /planning/investments
GET /planning/net-worth
POST /planning/net-worth/snapshots
GET /planning/recurring
POST /planning/recurring/detect

11 • Bill Pay (8 endpoints)

GET /bill-pay/summary
GET /bill-pay/payees
POST /bill-pay/payees
GET /bill-pay/bills
POST /bill-pay/bills
PATCH /bill-pay/bills/:id
POST /bill-pay/bills/:id/pay
POST /bill-pay/bills/:id/initiate-payment

12 • Credit Score (4 endpoints)

GET /credit-score/summary
GET /credit-score/entries
POST /credit-score/entries
POST /credit-score/pull

13 • Tax (9 endpoints)

GET /tax/returns
POST /tax/returns
PATCH /tax/returns/:id
POST /tax/returns/:id/documents
GET /tax/returns/:id/intake
PUT /tax/returns/:id/intake
POST /tax/returns/:id/export
POST /tax/returns/:id/efile
GET /tax/returns/:id/efile

14 • Billing (5 endpoints)

GET /billing/subscription
GET /billing/production-readiness
POST /billing/checkout
POST /billing/portal
POST /billing/webhook

15 • Notifications (8 endpoints)

GET /notifications
GET /notifications/preferences
PATCH /notifications/preferences
GET /notifications/vapid-public-key
POST /notifications/push-subscriptions
PATCH /notifications/:id/read
POST /notifications/read-all
POST /notifications/test

16 · Google Sheets (5 endpoints)

GET /google/connect

POST /google/exchange

DELETE /google/disconnect

GET /google/status

POST /google/data-system-mode

17 · Security, Compliance & Admin (4 endpoints)

GET /security/risk

GET /compliance/soc2/readiness

GET /compliance/soc2/operations

GET /admin/users

18 · View (Presentation API) (2 endpoints)

GET /view/accounts

GET /view/transactions

19 · API Keys (3 endpoints)

GET /api-keys

POST /api-keys

DELETE /api-keys/:id

20 · Public API (external, key-authed) (4 endpoints)

GET /public/v1/transactions

GET /public/v1/transactions/summary

GET /public/v1/transactions/cashflow

GET /public/v1/transactions/merchants

Conventions & Authentication

Read this once — every module below follows these exact rules.

Base URL & prefix

Every route is mounted under `/api`. Paths in this document omit the base URL — prefix them with `https://api.aarthalabs.com/api` in production or `http://localhost:3051/api` locally.

Response envelope

Every successful response — regardless of endpoint — is wrapped the same way:

```
{
  "data": <endpoint-specific payload>,
  "meta": { "timestamp": "2026-07-29T02:04:28.980Z", "version": "v1" },
  "error": null
}
```

Errors use the same envelope with `data: null` and a populated error:

```
{
  "data": null,
  "error": { "statusCode": 400, "message": "Invalid credentials." },
  "meta": { "timestamp": "2026-07-29T02:04:28.980Z", "version": "v1" }
}
```

Validation failures return an array of messages in `error.message`, one per failed field/rule.

Authentication — Bearer token + session nonce

Every endpoint requires authentication **except** those explicitly marked `PUBLIC` in this document (registration, login, password reset, email verification, social-auth start/callback, Plaid/Stripe webhooks, and signed export downloads).

Authenticated requests need two headers:

HEADER	VALUE
Authorization	<code>Bearer <accessToken></code> — issued by <code>/auth/login</code> or <code>/auth/register</code> . Expires in 60 seconds; use <code>/auth/refresh</code> to renew.
X-LedgerOne-Session-Nonce	A single-use, rotating value. Read it from <code>data.sessionNonce</code> on login/register for your first call, then on every response after that, read the <code>X-LedgerOne-Next-Nonce</code> response header and send that value as the nonce on your next request. Reusing a stale nonce returns 401.

This binds every access token to a specific session, IP, and user-agent, and prevents token replay even if a token is intercepted.

Plan-gated endpoints

Endpoints tagged `PLAN: PRO` require an active Pro or Elite subscription and return 403 with "This feature requires a pro plan." otherwise. Endpoints tagged `ROLE: ADMIN` require the caller's account to have the admin role and return 403 with "Insufficient role." otherwise.

Pagination

List endpoints accept `page` (default 1) and `limit` query params. For UI-facing list/view endpoints, `limit` is server-clamped to a maximum of **25** regardless of what's requested.

Opaque resource IDs

Account and transaction `id` values returned by browser-facing endpoints are encrypted, opaque tokens — not raw database IDs. Treat them as strings to pass back verbatim in path params or edit calls; do not attempt to parse or store them as UUIDs. They may rotate.

Public API (external integrations)

The `/public/v1/*` module uses a separate auth scheme: an API key created via `/api-keys`, sent as the `X-LedgerOne-API-Key` header instead of a bearer token. No session nonce is required for this module.

Auth & Sessions

Registration, login, password reset, social sign-in, profile, and GDPR self-service (view, export, delete your data). Base path /auth.

POST /auth/register **PUBLIC**

Create a new account. Returns an access token, refresh token, and session nonce immediately — no separate login call needed after registering.

REQUEST BODY

FIELD	TYPE	NOTES
email	string	required, valid email
password	string	required, min 8 characters

```
{ "email": "jane@example.com", "password": "TestPass123!" }
```

RESPONSE - 201

```
{
  "data": {
    "user": { "id": "ec92652a-...", "email": "jane@example.com",
      "fullName": null, "emailVerified": false
    },
    "accessToken": "eyJhbGciOi...",
    "refreshToken": "092f2658f5...",
    "sessionNonce": "0gkyGPXH2c...",
    "message": "Registration successful. Please verify your email."
  }, "error": null
}
```

POST /auth/login **PUBLIC**

Authenticate with email and password. Optionally supply a TOTP code if 2FA is enabled on the account.

REQUEST BODY

FIELD	TYPE	NOTES
email	string	required
password	string	required
totpToken	string	optional, required if 2FA enabled

```
{ "email": "jane@example.com", "password":  
  "TestPass123!" }
```

RESPONSE · 200

```
{  
  "data": {  
    "user": { "id": "ec92652a-...", "email":  
      "jane@example.com",  
      "fullName": null, "emailVerified": false  
    },  
    "accessToken": "eyJhbGciOi...",  
    "refreshToken": "2967d90b35...",  
    "sessionNonce": "GrMdxHLh00..."  
  }, "error": null  
}
```

GET /auth/verify-email **PUBLIC**

Confirms an email address using the token emailed at registration. Token expires after 24 hours.

QUERY PARAMS

FIELD	TYPE	NOTES
token	string	required, from the verification email link

RESPONSE · 200

```
{ "data": { "message": "Email verified." },  
  "error": null }
```

POST /auth/refresh **PUBLIC**

Exchanges a valid refresh token for a new access token, refresh token, and session nonce. The old refresh token is revoked (rotation).

REQUEST BODY

```
{ "refreshToken": "092f2658f5..." }
```

RESPONSE · 200

```
{ "data": { "accessToken": "eyJhbGciOi...",  
  "refreshToken": "a1b2c3d4...",  
  "sessionNonce": "xY9k..." }, "error": null }
```

POST /auth/logout

Revokes the given refresh token, ending that session.

REQUEST BODY

```
{ "refreshToken": "092f2658f5..." }
```

RESPONSE · 200

```
{ "data": { "message": "Logged out." },  
  "error": null }
```

POST /auth/forgot-password **PUBLIC**

Sends a password-reset email if the address exists. Always returns the same message either way (does not reveal whether an account exists).

REQUEST BODY

```
{ "email": "jane@example.com" }
```

RESPONSE · 200

```
{ "data": { "message": "If that email exists, a  
reset link has been sent." }, "error": null }
```

POST /auth/reset-password **PUBLIC**

Sets a new password using the token from the reset email (expires after 1 hour). Revokes all existing sessions.

REQUEST BODY

FIELD	TYPE	NOTES
token	string	required
password	string	required, min 8 chars

RESPONSE · 200

```
{ "data": { "message": "Password reset." },  
  "error": null }
```

GET /auth/social/:provider/url **PUBLIC**

Returns the authorization URL to redirect the user to for social login.

PARAMS

FIELD	TYPE	NOTES
:provider	path	google or apple
next	query	optional post-login redirect path

RESPONSE · 200

```
{ "data": { "url":  
  "https://accounts.google.com/o/oauth2/v2/auth?..  
  " }, "error": null }
```

POST `/auth/social/:provider/callback` **PUBLIC**

Completes social login: exchanges the provider's authorization code for a LedgerOne session.

REQUEST BODY

```
{ "code": "4/0AY0e-g7...", "state": "a1b2c3..."
}
```

RESPONSE · 200

Same shape as /auth/Login — user, accessToken, refreshToken, sessionNonce.

GET `/auth/me`

Returns the current authenticated user's profile.

REQUEST

No body or params.

RESPONSE · 200

```
{ "data": { "user": {
  "id": "ec92652a-...", "email":
  "jane@example.com",
  "fullName": null, "phone": null,
  "companyName": null,
  "emailVerified": false, "twoFactorEnabled":
  false,
  "createdAt": "2026-07-29T02:04:26.973Z" }
}, "error": null }
```

GET /auth/me/privacy-summary

GDPR privacy inventory — a per-category count of the personal data LedgerOne holds for this user, plus which controls are available.

REQUEST

No body or params.

RESPONSE · 200

```
{ "data": {
  "subject": { "id": "ec92652a-...", "email":
    "jane@example.com" },
  "dataCategories": { "accounts": 2,
    "transactions": 148, "rules": 3,
    "taxReturns": 0, "exports": 1,
    "notifications": 5 },
  "controls": { "exportEndpoint":
    "/api/auth/me/data-export",
    "deletionEndpoint": "/api/auth/me",
    "excludedFromExport": ["passwordHash",
    "bank access tokens", "raw bank payloads" ] }
}, "error": null }
```

GET /auth/me/data-export

Returns a minimized JSON export of this user's personal data — accounts, transactions, rules, exports, API keys, tax returns. Excludes secrets, tokens, and stable internal IDs.

REQUEST

No body or params.

RESPONSE · 200

```
{ "data": {
  "generatedAt": "2026-07-29T02:04:32.362Z",
  "subject": { "id": "ec92652a-...", "email":
    "jane@example.com" },
  "data": { "accounts": [], "transactions":
    [], "rules": [],
    "exports": [], "apiKeys": [],
    "taxReturns": [] }
}, "error": null }
```

PATCH /auth/profile

Updates profile fields. All fields optional — send only what changed.

REQUEST BODY

FIELD	TYPE	MAX
fullName	string?	200
phone	string?	30
companyName	string?	200
addressLine1 / addressLine2	string?	300
city / state / country	string?	100
postalCode	string?	20

```
{ "fullName": "Jane Doe" }
```

RESPONSE · 200

```
{ "data": { "user": { "id": "ec92652a-...",  
  "email": "jane@example.com", "fullName":  
  "Jane Doe", ... } }, "error": null }
```

DELETE /auth/me

Permanently deletes the account and all associated application data in one transaction.
Irreversible.

REQUEST

No body or params.

RESPONSE · 200

```
{ "data": { "message": "Account and associated  
personal data deleted." }, "error": null }
```

MODULE 02

Two-Factor Authentication

TOTP-based 2FA enrollment. Base path /auth/2fa.

POST /auth/2fa/generate

Generates a new TOTP secret and returns a scannable QR code to enroll in an authenticator app. Does not enable 2FA yet.

REQUEST

No body.

RESPONSE · 200

```
{ "data": { "qrCode":  
  "...",  
  "secret": "JBSWY3DPEHPK3PXP" }, "error": null  
}
```

POST /auth/2fa/enable

Confirms enrollment by verifying a code from the authenticator app, then turns 2FA on for the account.

REQUEST BODY

```
{ "token": "482913" }
```

RESPONSE · 200

```
{ "data": { "message": "Two-factor  
authentication enabled." }, "error": null }
```

DELETE /auth/2fa/disable

Turns off 2FA after verifying a current TOTP code.

REQUEST BODY

```
{ "token": "482913" }
```

RESPONSE · 200

```
{ "data": { "message": "Two-factor  
authentication disabled." }, "error": null }
```

Accounts

Bank and manual accounts. Account IDs returned here are opaque tokens, not database IDs. Base path /accounts.

GET

/accounts

Lists the caller's accounts (Plaid, Teller, and manual).

QUERY PARAMS

FIELD	TYPE	NOTES
page	number	default 1
limit	number	default 20, clamped to 25

RESPONSE · 200

```
{ "data": { "accounts": [{
  "id": "SnJuTFVMMXNBaUIv...",
  "institutionName": "Chase Checking",
  "accountType": "checking", "mask": null,
  "currentBalance": 1000.00, "ownershipType":
  "mine",
  "syncStatus": "idle", "isActive": true }],
  "total": 1, "page": 1, "limit": 20 },
  "error": null }
```

POST

/accounts/link

Creates a Plaid Link token to open the Plaid Link widget for connecting a bank.

REQUEST

No body.

RESPONSE · 200

```
{ "data": { "linkToken": "link-sandbox-8f2..." },
  "error": null }
```

POST /accounts/manual

Adds an account by hand (cash, an unsupported bank, etc.).

REQUEST BODY

FIELD	TYPE	NOTES
institutionName	string	required, max 120 chars
accountType	string	required — checking, savings, credit, investment, loan, cash, other
mask	string?	optional, max 12 chars (e.g. last 4 digits)

```
{ "institutionName": "Chase Checking",  
  "accountType": "checking" }
```

RESPONSE · 201

```
{ "data": { "id": "M1NkQ2F3Z3l1...",  
            "institutionName": "Chase Checking",  
            "accountType": "checking", "mask": null,  
            "ownershipType": "mine",  
            "isActive": true, "createdAt": "2026-07-  
29T02:04:37.504Z" }, "error": null }
```

POST /accounts/balances

Refreshes balances for all of the caller's connected (Plaid/Teller) accounts.

REQUEST

No body.

RESPONSE · 200

```
{ "data": { "updated": 2 }, "error": null }
```

PATCH /accounts/:id/ownership

Sets whose the account is within a household — yours, theirs, or joint.

REQUEST BODY

FIELD	TYPE	NOTES
ownershipType	string	required — mine, theirs, joint
householdId	string?	required if not "mine"
ownerUserId	string?	required if "theirs"

```
{ "ownershipType": "joint", "householdId":  
  "d0cdfdda-..." }
```

RESPONSE · 200

```
{ "data": { "id": "SnJuTFVM...",  
            "ownershipType": "joint" }, "error": null }
```

Transactions

The ledger — listing, importing, editing, and summarizing transactions. Base path `/transactions`.

GET

/transactions

Lists transactions with filters.

QUERY PARAMS

FIELD	NOTES
start_date, end_date	ISO date bounds
account_id	opaque account id
min_amount, max_amount	numeric bounds
category, source, search	text filters
include_hidden	"true"/"false"
page, limit	default 1 / 25, clamped to 25

RESPONSE · 200

```
{ "data": { "transactions": [{
  "id": "OWFEQktsMXZ...", "name": "Coffee
Shop", "amount": "-4.50",
  "category": "Dining", "note": "",
  "attribution": "mine",
  "split": { "mode": "none", "minePercent":
100, "mineAmount": -4.5 },
  "status": "user", "hidden": false, "date":
"2026-07-15",
  "source": "manual", "accountId":
"L3kxK0VGSHVU..." } ] },
  "total": 1, "page": 1, "limit": 25 },
  "error": null }
```

POST

/transactions/import

Imports a single CSV file (Chase, Bank of America, Wells Fargo, or generic format auto-detected). Multipart upload, max 5 MB.

REQUEST

FIELD	TYPE	NOTES
file	multipart	required, form field name "file"
mapping	string?	optional JSON column-mapping override

RESPONSE · 201

```
{ "data": { "imported": 2, "skipped": 0,
"total": 2 }, "error": null }
```

POST /transactions/import/preview

Previews a CSV before importing: detects headers, row count, and column mapping (including any mapping remembered from a prior import), without writing rows or exposing raw sample data.

REQUEST

Multipart, field name "file", max 5 MB.

RESPONSE · 200

```
{ "data": { "fileName": "test.csv",
"headerSignature": "9bf87c92...",
"headers": ["Date", "Description", "Amount"],
"rowCount": 2,
"mapping": { "date": "Date", "description":
"Description", "amount": "Amount" },
"remembered": false }, "error": null }
```

POST /transactions/import/batch

Imports up to 20 CSV files in one request, continuing on a per-file error basis.

REQUEST

Multipart, field name "files" (up to 20, 5 MB each). Optional mappings JSON.

RESPONSE · 201

```
{ "data": { "results": [
{ "fileName": "chase.csv", "imported": 40,
"skipped": 1 },
{ "fileName": "boa.csv", "imported": 12,
"skipped": 0 }
] }, "error": null }
```

POST /transactions/sync

Triggers an on-demand Plaid transaction sync for the caller's connected accounts.

REQUEST BODY

```
{ "startDate": "2026-07-01", "endDate": "2026-
07-29" }
```

RESPONSE · 200

```
{ "data": { "synced": 2, "newTransactions": 14
}, "error": null }
```

POST /transactions/manual

Adds a transaction by hand. Requires the caller to already have at least one account.

REQUEST BODY

FIELD	TYPE	NOTES
date	string	required, ISO date
description	string	required
amount	number	required, negative = expense
accountId	string?	optional, defaults to first account
category, note	string?	optional
attribution	string?	mine, yours, ours
splitMode	string?	none, equal, custom
splitMinePercent / splitYoursPercent	number?	0–100, required if splitMode is custom
hidden	boolean?	default false

RESPONSE · 201

```
{ "data": { "id": "ZGQ5V1MzMV..." }, "error":  
null }
```

```
{ "description": "Coffee Shop", "amount": -4.5,  
  "date": "2026-07-15", "category": "Dining" }
```

GET /transactions/summary

Income, expense, and net totals over an optional date range.

QUERY PARAMS

FIELD	NOTES
start_date, end_date	optional ISO bounds

RESPONSE · 200

```
{ "data": { "total": "1995.50", "count": 2,
  "income": "2000.00", "expense": "4.50",
  "net": "1995.50" }, "error": null }
```

GET /transactions/cashflow

Monthly income/expense/net for a rolling window.

QUERY PARAMS

FIELD	NOTES
months	default 6

RESPONSE · 200

```
{ "data": [
  { "month": "2026-06", "income": "0.00",
    "expense": "0.00", "net": "0.00" },
  { "month": "2026-07", "income": "2000.00",
    "expense": "4.50", "net": "1995.50" }
], "error": null }
```

GET /transactions/merchants

Top merchants by spend.

QUERY PARAMS

FIELD	NOTES
limit	default 6

RESPONSE · 200

```
{ "data": [ { "merchant": "Starbucks", "total":
  "5.75", "count": 1 } ], "error": null }
```

PATCH /transactions/:id/derived

Edits the user-editable layer of a transaction (category, notes, attribution, split, hidden) without touching the original imported record.

REQUEST BODY

FIELD	TYPE
userCategory	string?
userNotes	string?
attribution	"mine" "yours" "ours"
splitMode	"none" "equal" "custom"
splitMinePercent / splitYoursPercent	number?
isHidden	boolean?

```
{ "userCategory": "Groceries", "userNotes": "edited" }
```

RESPONSE · 200

```
{ "data": { "id": "34b1fd1c-...",  
  "userCategory": "Groceries",  
  "userNotes": "edited", "attribution":  
    "mine", "splitMode": "none",  
    "isHidden": false, "modifiedAt":  
      "2026-07-29T02:04:39.756Z",  
    "modifiedBy": "user" }, "error": null  
}
```

GET /transactions/:id/comments

Lists comments on a transaction (for household discussion threads).

REQUEST

No body.

RESPONSE · 200

```
{ "data": [ { "id": "c1a2...", "body": "was  
this the one from the trip?",  
  "author": { "id": "ec92...", "email":  
    "jane@example.com" },  
  "createdAt": "2026-07-29T03:10:00.000Z" } ],  
  "error": null }
```

POST /transactions/:id/comments

Adds a comment to a transaction.

REQUEST BODY

```
{ "body": "was this the one from the trip?" }
```

RESPONSE · 201

```
{ "data": { "id": "c1a2...", "body": "was this  
the one from the trip?" }, "error": null }
```

Rules Engine

Transparent auto-categorization rules — conditions and actions are plain JSON, visible and editable by the user. Base path `/rules`.

GET `/rules`

Lists the caller's rules, ordered by priority.

REQUEST

No params.

RESPONSE · 200

```
{ "data": [ { "id": "3fce267a-...", "name":  
  "Coffee rule", "priority": 1,  
    "conditions": { "descriptionContains":  
      "Coffee" },  
    "actions": { "setCategory": "Dining" },  
    "isActive": true } ], "error": null }
```

POST /rules

Creates a rule. conditions supports description contains/not-contains/equals/regex, amount comparisons, source, category, and date bounds — or a nested all/any/not DSL for advanced logic. actions supports setting/clearing category, notes, and hidden state.

REQUEST BODY

FIELD	TYPE	NOTES
name	string	required
priority	number?	lower runs first
conditions	object	required, see above
actions	object	required, see above
isActive	boolean?	default true

```
{ "name": "Coffee rule", "priority": 1,
  "conditions": { "descriptionContains":
    "Coffee" },
  "actions": { "setCategory": "Dining" } }
```

RESPONSE · 201

```
{ "data": { "id": "3fce267a-...", "name":
  "Coffee rule",
  "priority": 1, "isActive": true,
  "createdAt": "2026-07-29T02:04:46.466Z" },
  "error": null }
```

PUT /rules/:id

Updates a rule. Same fields as create, all optional.

REQUEST BODY

```
{ "priority": 2, "isActive": false }
```

RESPONSE · 200

```
{ "data": { "id": "3fce267a-...", "priority":
  2, "isActive": false }, "error": null }
```

POST /rules/:id/execute

Runs a rule against the caller's full transaction history (not just new transactions).

REQUEST

No body.

RESPONSE · 200

```
{ "data": { "matched": 7, "updated": 7 },
  "error": null }
```

GET /rules/suggestions

Suggests new rules from categorization history, recurring uncategorized merchants, refunds, transfers, and fees, with a confidence score and match count.

REQUEST

No params.

RESPONSE · 200

```
{ "data": [ { "reason":  
  "recurring_uncategorized_merchant",  
  "suggestedCondition": {  
    "descriptionContains": "NETFLIX" },  
    "suggestedAction": { "setCategory":  
      "Subscriptions" },  
    "confidence": 0.86, "matchCount": 6 } ],  
  "error": null }
```

MODULE 06

Exports

Server-generated, watermarked exports delivered via signed, single-use download links.
Requires an active Pro or Elite plan. Base path /exports.

Plan: Pro or Elite required for all routes except download

GET /exports/csv **PLAN: PRO**

Generates a CSV of the caller's transactions (respecting the same filters as /transactions) and returns a one-time signed download URL.

QUERY PARAMS

Same filters as GET /transactions (start_date, end_date, category, source, search, etc.)

RESPONSE · 200

```
{ "data": { "status": "signed",  
  "downloadUrl": "/api/exports/download/1DKq-  
jsNxT6iB2s0...",  
  "expiresAt": "2026-07-29T02:06:24.816Z",  
  "singleUse": true, "expiresInSeconds": 120,  
  "storageProvider": "local" }, "error": null }
```

GET /exports/json **PLAN: PRO**

Same as CSV, but as a structured JSON file.

REQUEST

Same filters as above.

RESPONSE · 200

Same signed-URL shape as /exports/csv.

GET /exports/xlsx **PLAN: PRO**

Same as CSV, but as an Excel workbook.

REQUEST

Same filters as above.

RESPONSE · 200

Same signed-URL shape as /exports/csv.

GET **/exports/pdf** **PLAN: PRO**

Same as CSV, but as a PDF ledger snapshot.

REQUEST

Same filters as above.

RESPONSE · 200

Same signed-URL shape as `/exports/csv`.

GET **/exports/download/:token** **PUBLIC**

Consumes a signed download URL and streams the file. Each token works exactly once and expires after 120 seconds — a second request returns `410 Gone`. The returned file carries an embedded watermark (user id, generation time, trace id).

PARAMS

FIELD	NOTES
<code>:token</code>	from the <code>downloadUrl</code> above

RESPONSE · 200

Binary file stream with the appropriate `Content-Type`. Second call → `410: "Download Link has already been used."`

POST **/exports/sheets** **PLAN: PRO**

Pushes the caller's ledger to their connected Google Sheets mirror (creates the sheet on first use).

REQUEST

Optional filter query params, same as CSV.

RESPONSE · 200

```
{ "data": { "spreadsheetUrl":  
  "https://docs.google.com/spreadsheets/d/1a2b3c/  
  edit",  
  "tab": "LedgerOne Sync", "rowsSynced": 148 },  
  "error": null }
```

Plaid

Bank connection via Plaid Link, including re-auth ("update mode") when a connection breaks. Base path /plaid.

POST /plaid/link-token

Creates a link token to open Plaid Link for a new bank connection. (Equivalent to /accounts/link.)

REQUEST

No body.

RESPONSE · 200

```
{ "data": { "linkToken": "link-sandbox-8f2..." }, "error": null }
```

POST /plaid/exchange

Exchanges a Plaid Link public_token for a permanent, encrypted access token and imports the connected accounts.

REQUEST BODY

```
{ "publicToken": "public-sandbox-8f2..." }
```

RESPONSE · 200

```
{ "data": { "accountsLinked": 2 }, "error": null }
```

POST /plaid/update-link-token

Creates a Plaid "update mode" link token for re-authenticating an existing, broken connection (e.g. after a password change at the bank).

REQUEST BODY

```
{ "accountId": "SnJuTFVMMXNBaUIv..." }
```

RESPONSE · 200

```
{ "data": { "linkToken": "link-sandbox-9c1..." }, "error": null }
```

POST /plaid/repair-complete

Marks an account's re-auth as complete after a successful Plaid Link "update mode" flow.

REQUEST BODY

```
{ "accountId": "SnJuTFVMMXNBaUIv..." }
```

RESPONSE · 200

```
{ "data": { "syncStatus": "idle" }, "error": null }
```

POST /plaid/webhook **PUBLIC**

Receives Plaid webhook events (new transactions, item errors, permission revocation). Called by Plaid's servers, not the frontend — signature-verified via the Plaid-Verification header.

REQUEST

Raw Plaid webhook payload; not called by the frontend.

RESPONSE · 200

```
{ "data": { "received": true }, "error": null }
```

MODULE 08

Teller

A second bank-data provider, used as a Plaid alternative/backup. Base path /teller.

GET /teller/config

Returns the Teller Connect widget configuration (application ID, environment, requested products).

REQUEST

No params.

RESPONSE · 200

```
{ "data": { "applicationId": "app_p6qft...",
"environment": "sandbox",
  "products": ["transactions", "balance"] },
"error": null }
```

POST /teller/enrollment

Completes a Teller Connect enrollment: stores the encrypted access token and imports the connected accounts.

REQUEST BODY

```
{ "accessToken": "token_abc123",
  "enrollment": { "id": "enr_xyz",
"institution": { "name": "Chase" } } }
```

RESPONSE · 200

```
{ "data": { "accountsLinked": 1 }, "error":
null }
```

POST /teller/sync

Triggers an on-demand sync of transactions and balances for Teller-connected accounts.

REQUEST

No body.

RESPONSE · 200

```
{ "data": { "synced": 1, "newTransactions": 5
}, "error": null }
```

Households (Couples)

Shared finance for couples and families — roles, invites, shared dashboards, goals, and planning tools. Base path /households.

GET /households

Lists households the caller belongs to.

REQUEST

No params.

RESPONSE · 200

```
{ "data": [ { "id": "d0cdffda-...", "name": "Doe Household", "createdByUserId": "ec92652a-..." } ], "error": null }
```

POST /households

Creates a household; the caller becomes its owner.

REQUEST BODY

FIELD	TYPE	NOTES
name	string	required, 2–120 chars
metadata	object?	optional

```
{ "name": "Doe Household" }
```

RESPONSE · 201

```
{ "data": { "id": "d0cdffda-...", "name": "Doe Household", "members": [{ "role": "owner", "status": "active", "user": { "id": "ec92652a-...", "email": "jane@example.com" } } ] }, "error": null }
```

GET /households/:id

Gets a household's detail. Restricted to active members.

REQUEST

No params.

RESPONSE · 200

Same shape as the create response.

GET /households/:id/dashboard

The shared household dashboard: combined balances by ownership (mine/theirs/joint), six-month cash flow, recent shared transactions, and a couples financial health score. Individual balances are hidden when privacy mode is on.

REQUEST

No params.

RESPONSE · 200

```
{ "data": { "members": [...], "accounts": {
  "mine": [], "theirs": [], "joint": [] },
  "cashflow": [...], "recentTransactions":
  [...],
  "healthScore": { "score": 78, "components":
  {...}, "recommendations": [...] }
}, "error": null }
```

PATCH /households/:id/privacy

Toggles financial-independence / privacy mode, which hides individual mine/theirs balances on the shared dashboard while keeping joint data visible.

REQUEST BODY

```
{ "enabled": true, "hideIndividualBalances":
true,
  "hideIndividualTransactions": false }
```

RESPONSE · 200

```
{ "data": { "enabled": true }, "error": null }
```

GET /households/:id/members

Lists members, their roles, and status.

REQUEST

No params.

RESPONSE · 200

```
{ "data": [ { "userId": "ec92652a-...", "role":
"owner", "status": "active" } ], "error": null
}
```

PATCH /households/:id/members/:memberId

Changes a member's role or status. Owner/admin only; the last active owner cannot be demoted or removed.

REQUEST BODY

FIELD	TYPE
-------	------

role	owner admin member viewer accountant advisor
-------------	--

status	active inactive removed
---------------	-------------------------

```
{ "role": "admin" }
```

RESPONSE · 200

```
{ "data": { "role": "admin", "status": "active" }, "error": null }
```

GET /households/:id/invites

Lists pending invites for the household.

REQUEST

No params.

RESPONSE · 200

```
{ "data": [ { "id": "inv_1", "email": "partner@example.com", "role": "member", "status": "pending" } ], "error": null }
```

POST /households/:id/invites

Invites someone by email. Owner/admin only. Sends an email with a single-use, expiring accept link.

REQUEST BODY

FIELD	TYPE	NOTES
-------	------	-------

email	string	required
--------------	--------	----------

role	string?	owner, admin, member, viewer, accountant, advisor — default member
-------------	---------	--

```
{ "email": "partner@example.com", "role": "member" }
```

RESPONSE · 201

```
{ "data": { "id": "inv_1", "status": "pending" }, "error": null }
```

POST /households/invites/accept

Accepts an invite using its token. Caller's email must match the invited email.

REQUEST BODY

```
{ "token": "a1b2c3d4e5f6..." }
```

RESPONSE · 200

```
{ "data": { "householdId": "d0cdfdda-...",  
  "role": "member" }, "error": null }
```

GET /households/:id/goals

Lists shared household goals.

REQUEST

No params.

RESPONSE · 200

```
{ "data": [ { "id": "g_1", "name": "House down  
payment",  
  "targetAmount": 50000, "currentAmount":  
12000, "status": "active" } ], "error": null }
```

POST /households/:id/goals

Creates a shared goal.

REQUEST BODY

FIELD	TYPE	NOTES
name	string	required, max 120
targetAmount	number	required, > 0
currentAmount	number?	default 0
targetDate	string?	ISO date
priority	string?	low, medium, high

```
{ "name": "House down payment", "targetAmount":  
50000, "priority": "high" }
```

RESPONSE · 201

```
{ "data": { "id": "g_1", "status": "active" },  
  "error": null }
```

PATCH /households/:id/goals/:goalId

Updates a goal — contribute progress, change priority/date, or mark complete/reopen via status.

REQUEST BODY

```
{ "currentAmount": 15000 }
```

RESPONSE · 200

```
{ "data": { "id": "g_1", "currentAmount": 15000 }, "error": null }
```

POST /households/:id/fair-split

Calculates a fair split of a shared expense — equal, custom percentage, or income-weighted.

REQUEST BODY

```
{ "expenseAmount": 200, "splitMode": "income_weighted", "mineMonthlyIncome": 6000, "yoursMonthlyIncome": 4000 }
```

RESPONSE · 200

```
{ "data": { "mineShare": 120, "yoursShare": 80, "minePercent": 60, "yoursPercent": 40 }, "error": null }
```

POST /households/:id/debt-payoff

Builds a debt payoff plan (avalanche, snowball, or custom order) with a monthly timeline and a suggested partner contribution split.

REQUEST BODY

```
{ "strategy": "avalanche", "monthlyExtraPayment": 300, "debts": [ { "name": "Visa", "balance": 4200, "annualPercentageRate": 21.9, "minimumPayment": 100 } ] }
```

RESPONSE · 200

```
{ "data": { "payoffOrder": ["Visa"], "monthsToPayoff": 14, "totalInterestPaid": 386.20, "timeline": [...], "recommendations": [...] }, "error": null }
```

POST /households/:id/future-scenarios

Compares up to 6 projected financial scenarios (goal, net worth, income change, expense change) side by side.

REQUEST BODY

```
{ "scenarios": [ { "name": "Buy a house",  
  "type": "goal",  
  "startingBalance": 12000,  
  "monthlyContribution": 800,  
  "targetAmount": 50000, "horizonMonths": 48 }  
] }
```

RESPONSE · 200

```
{ "data": { "scenarios": [ { "name": "Buy a  
house", "finalBalance": 50400,  
  "earliestTargetMonth": 47 } ],  
  "bestFinalBalance": "Buy a house" }, "error":  
null }
```

GET /households/:id/money-date-prompts

Generates conversation prompts from real household data — cash flow trends, goal progress, ownership balance, and sync health.

REQUEST

No params.

RESPONSE · 200

```
{ "data": [ { "topic": "goal_progress",  
  "prompt": "You're 24% toward House down  
payment — want to revisit the timeline?",  
  "priority": "medium" } ], "error": null }
```

GET /households/:id/accountant-tasks

Lists tasks assigned to an accountant/advisor collaborator (document requests, review items).

REQUEST

No params.

RESPONSE · 200

```
{ "data": [ { "id": "t_1", "title": "Send Q3  
statements", "status": "open" } ], "error":  
null }
```

POST /households/:id/accountant-tasks

Creates a task for an accountant/advisor collaborator.

REQUEST BODY

```
{ "title": "Send Q3 statements", "priority":  
  "medium" }
```

RESPONSE · 201

```
{ "data": { "id": "t_1", "status": "open" },  
  "error": null }
```

Planning

Budgets, personal goals, investment holdings, net worth snapshots, and recurring-transaction detection. Base path /planning.

GET

/planning/budgets

Lists budgets — personal or, if householdId is passed, shared.

QUERY PARAMS

FIELD	NOTES
householdId	optional

RESPONSE · 200

```
{ "data": [ { "id": "b_1", "name": "Groceries",
"category": "Food",
  "period": "monthly", "limitAmount": 600,
"spentAmount": 312.40, "status": "active" } ],
"error": null }
```

POST

/planning/budgets

Creates a budget.

REQUEST BODY

FIELD	TYPE	NOTES
householdId	string	required
name	string	required, max 120
category	string?	max 80
period	string?	weekly, monthly, quarterly, annual
limitAmount	number	required, > 0

RESPONSE · 201

```
{ "data": { "id": "b_1", "status": "active" },
"error": null }
```

```
{ "householdId": "d0cdfdda-...", "name":
"Groceries",
  "category": "Food", "period": "monthly",
"limitAmount": 600 }
```

PATCH /planning/budgets/:id

Updates a budget's limit, spend, or status.

REQUEST BODY

```
{ "limitAmount": 650 }
```

RESPONSE · 200

```
{ "data": { "id": "b_1", "limitAmount": 650 },  
  "error": null }
```

GET /planning/goals

Lists the caller's personal (non-household) goals.

REQUEST

No params.

RESPONSE · 200

```
{ "data": [ { "id": "pg_1", "name": "Emergency  
fund", "targetAmount": 10000,  
              "currentAmount": 3200, "status": "active" }  
], "error": null }
```

POST /planning/goals

Creates a personal goal, optionally with automation rules (e.g. recurring auto-contributions).

REQUEST BODY

```
{ "name": "Emergency fund", "targetAmount":  
  10000, "priority": "high" }
```

RESPONSE · 201

```
{ "data": { "id": "pg_1", "status": "active" },  
  "error": null }
```

PATCH /planning/goals/:id

Updates goal progress or status.

REQUEST BODY

```
{ "currentAmount": 3800 }
```

RESPONSE · 200

```
{ "data": { "id": "pg_1", "currentAmount": 3800  
}, "error": null }
```

GET /planning/investments

Lists tracked investment holdings.

REQUEST

No params.

RESPONSE · 200

```
{ "data": [ { "id": "inv_1", "symbol": "VTI",
"assetClass": "etf",
  "quantity": 12.5, "price": 265.40 } ],
"error": null }
```

POST /planning/investments

Adds an investment holding.

REQUEST BODY

FIELD	TYPE	NOTES
symbol	string	required, max 20
name	string	required, max 120
assetClass	string?	stock, etf, mutual_fund, bond, crypto, cash, other
quantity, price	number	required, >= 0

```
{ "symbol": "VTI", "name": "Vanguard Total
Stock Market ETF",
  "assetClass": "etf", "quantity": 12.5,
"price": 265.40 }
```

RESPONSE · 201

```
{ "data": { "id": "inv_1" }, "error": null }
```

GET /planning/net-worth

Net worth over time, from recorded snapshots.

REQUEST

No params.

RESPONSE · 200

```
{ "data": [ { "snapshotDate": "2026-07-01",
"assets": 42000,
  "liabilities": 8000, "netWorth": 34000 } ],
"error": null }
```

POST /planning/net-worth/snapshots

Records a point-in-time net worth snapshot.

REQUEST BODY

```
{ "assets": 42000, "liabilities": 8000 }
```

RESPONSE · 201

```
{ "data": { "snapshotDate": "2026-07-29",  
  "netWorth": 34000 }, "error": null }
```

GET /planning/recurring

Lists detected recurring transactions (subscriptions, repeating bills).

REQUEST

No params.

RESPONSE · 200

```
{ "data": [ { "merchant": "NETFLIX",  
  "averageAmount": 15.49,  
  "cadence": "monthly", "nextExpectedDate":  
  "2026-08-14" } ], "error": null }
```

POST /planning/recurring/detect

Re-runs recurring-transaction detection over the caller's transaction history.

REQUEST

No body.

RESPONSE · 200

```
{ "data": { "detected": 6 }, "error": null }
```

MODULE 11

Bill Pay

Payees, bill schedules, and payment recording. Base path `/bill-pay`.

GET `/bill-pay/summary`

Upcoming and overdue totals across all bills.

REQUEST

No params.

RESPONSE · 200

```
{ "data": { "upcomingCount": 3,
"upcomingTotal": 420.50,
  "overdueCount": 1, "overdueTotal": 89.00 },
"error": null }
```

GET `/bill-pay/payees`

Lists saved payees.

REQUEST

No params.

RESPONSE · 200

```
{ "data": [ { "id": "p_1", "name": "City Water
Utility", "category": "Utilities" } ], "error":
null }
```

POST `/bill-pay/payees`

Creates a payee.

REQUEST BODY

```
{ "name": "City Water Utility", "category":
"Utilities" }
```

RESPONSE · 201

```
{ "data": { "id": "p_1" }, "error": null }
```

GET /bill-pay/bills

Lists bills, optionally filtered by status.

QUERY PARAMS

FIELD	NOTES
status	optional

RESPONSE - 200

```
{ "data": [ { "id": "bl_1", "name": "Water bill", "amount": 89.00, "dueDate": "2026-08-05", "status": "pending", "autopay": false } ], "error": null }
```

POST /bill-pay/bills

Creates a bill.

REQUEST BODY

FIELD	TYPE	NOTES
name	string	required
amount	number	required, > 0
dueDate	string	required, ISO date
payeeId	string?	optional
recurrence	string?	none, weekly, monthly, quarterly, yearly
autopay	boolean?	default false
reminderDays	number?	0-30

RESPONSE - 201

```
{ "data": { "id": "bl_1", "status": "pending" }, "error": null }
```

```
{ "name": "Water bill", "amount": 89, "dueDate": "2026-08-05", "payeeId": "p_1", "recurrence": "monthly" }
```

PATCH /bill-pay/bills/:id

Updates a bill's amount, due date, autopay, or status.

REQUEST BODY

```
{ "amount": 92.50 }
```

RESPONSE - 200

```
{ "data": { "id": "bl_1", "amount": 92.50 }, "error": null }
```

POST /bill-pay/bills/:id/pay

Records a manual payment against a bill.

REQUEST BODY

```
{ "amount": 89, "method": "manual" }
```

RESPONSE · 200

```
{ "data": { "id": "bl_1", "status": "paid" },  
  "error": null }
```

POST /bill-pay/bills/:id/initiate-payment

Initiates an electronic payment for a bill (ACH, bank bill-pay, or card). Note: the external payment rail is not configured in this environment — this records the payment intent server-side.

REQUEST BODY

```
{ "amount": 89, "method": "ach" }
```

RESPONSE · 200

```
{ "data": { "id": "bl_1", "status": "scheduled"  
}, "error": null }
```

Credit Score

Credit score history and trend tracking. Base path /credit-score.

GET

/credit-score/summary

Latest score, change since last entry, and trend.

REQUEST

No params.

RESPONSE · 200

```
{ "data": { "latest": 742, "previous": 738,
"change": 4,
  "averageScore": 740, "entryCount": 6,
"trend": [...] }, "error": null }
```

GET

/credit-score/entries

Lists individual score entries.

QUERY PARAMS

FIELD	NOTES
bureau	optional filter

RESPONSE · 200

```
{ "data": [ { "id": "cs_1", "score": 742,
"bureau": "experian",
  "scoreDate": "2026-07-01" } ], "error": null
}
```

POST /credit-score/entries

Logs a score entry manually.

REQUEST BODY

FIELD	TYPE	NOTES
score	number	required, 300–850
bureau	string?	experian, equifax, transunion, unknown
scoreDate	string	required, ISO date

```
{ "score": 742, "bureau": "experian",  
  "scoreDate": "2026-07-01" }
```

RESPONSE · 201

```
{ "data": { "id": "cs_1" }, "error": null }
```

POST /credit-score/pull

Requests a fresh score pull from a provider. Note: direct bureau/provider integration is not configured in this environment; this records a consented pull request server-side.

REQUEST BODY

```
{ "bureau": "experian", "consent": {  
  "acceptedAt": "2026-07-29T02:00:00.000Z" } }
```

RESPONSE · 200

```
{ "data": { "status": "requested" }, "error":  
null }
```

Tax

Tax returns, intake workflow, document attachment, package export, and sandbox e-file.
Base path /tax.

GET

/tax/returns

Lists the caller's tax returns.

REQUEST

No params.

RESPONSE · 200

```
{ "data": [ { "id": "62d53435-...", "taxYear": 2025, "filingType": "individual", "status": "draft" } ], "error": null }
```

POST

/tax/returns

Creates a new tax return.

REQUEST BODY

FIELD	TYPE	NOTES
taxYear	number	required, integer
filingType	string	required — individual or business
jurisdictions	string[]	required, e.g. ["US"]

```
{ "taxYear": 2025, "filingType": "individual", "jurisdictions": ["US"] }
```

RESPONSE · 201

```
{ "data": { "id": "62d53435-...", "taxYear": 2025, "filingType": "individual", "jurisdictions": ["US"], "status": "draft", "summary": {}, "createdAt": "2026-07-29T02:04:03.742Z" }, "error": null }
```

PATCH /tax/returns/:id

Updates a return's status or summary.

REQUEST BODY

```
{ "status": "ready" }
```

RESPONSE · 200

```
{ "data": { "id": "62d53435-...", "status":  
  "ready" }, "error": null }
```

POST /tax/returns/:id/documents

Attaches document metadata (e.g. a W-2 or 1099) to a return.

REQUEST BODY

```
{ "docType": "W-2", "metadata": { "employer":  
  "Acme Inc." } }
```

RESPONSE · 201

```
{ "data": { "id": "doc_1", "docType": "W-2" },  
  "error": null }
```

GET /tax/returns/:id/intake

Gets the full income/deductions/credits intake worksheet for a return.

REQUEST

No params.

RESPONSE · 200

```
{ "data": { "intake": { "income": {},  
  "deductions": {}, "credits": {} },  
  "readiness": { "complete": false, "missing":  
    ["W-2 income"] } }, "error": null }
```

PUT /tax/returns/:id/intake

Saves the intake worksheet as a draft, or submits it (marks ready) with submit: true.

REQUEST BODY

```
{ "intake": { "income": { "wages": 85000 } },  
  "submit": false }
```

RESPONSE · 200

```
{ "data": { "saved": true }, "error": null }
```

POST /tax/returns/:id/export

Generates a versioned tax package (manifest, readiness checklist, attached documents, SHA-256 package hash) and logs the export.

REQUEST

No body.

RESPONSE · 200

```
{ "data": { "packageVersion": 1, "packageHash":  
  "9f2a1e...",  
  "manifest": {...}, "readiness": { "complete":  
    true } }, "error": null }
```

POST /tax/returns/:id/efile

Submits a return to the sandbox e-file provider. Requires readiness checks to pass and explicit consent.

REQUEST BODY

```
{ "consentAccepted": true }
```

RESPONSE · 200

```
{ "data": { "submissionId": "efile_sandbox_1",  
  "status": "submitted" }, "error": null }
```

GET /tax/returns/:id/efile

Polls the status of a submitted e-file.

REQUEST

No params.

RESPONSE · 200

```
{ "data": { "submissionId": "efile_sandbox_1",  
  "status": "accepted" }, "error": null }
```

MODULE 14

Billing

Stripe-backed subscription management. Base path `/billing`.

GET `/billing/subscription`

Returns the caller's current plan.

REQUEST

No params.

RESPONSE · 200

```
{ "data": { "userId": "ec92652a-...", "plan":  
  "free" }, "error": null }
```

GET `/billing/production-readiness`

Reports whether live Stripe keys/webhook secrets are configured (operational health check).

REQUEST

No params.

RESPONSE · 200

```
{ "data": { "liveKeysConfigured": false,  
  "webhookConfigured": false }, "error": null }
```

POST `/billing/checkout`

Creates a Stripe Checkout session for upgrading to Pro or Elite.

REQUEST BODY

```
{ "plan": "pro", "successUrl":  
  "https://app.aarthalabs.com/settings/subscripti  
on?success=1",  
  "cancelUrl":  
  "https://app.aarthalabs.com/settings/subscripti  
on" }
```

RESPONSE · 200

```
{ "data": { "checkoutUrl":  
  "https://checkout.stripe.com/c/pay/cs_test_..."  
}, "error": null }
```

POST /billing/portal

Creates a Stripe Billing Portal session for self-service plan/payment management.

REQUEST BODY

```
{ "returnUrl":  
  "https://app.aarthalabs.com/settings/subscripti  
on" }
```

RESPONSE · 200

```
{ "data": { "portalUrl":  
  "https://billing.stripe.com/p/session/..." },  
  "error": null }
```

POST /billing/webhook **PUBLIC**

Receives Stripe webhook events (checkout completed, subscription updated/cancelled, payment failed). Called by Stripe's servers, not the frontend — signature-verified via the Stripe-Signature header.

REQUEST

Raw Stripe event payload; not called by the frontend.

RESPONSE · 200

```
{ "data": { "received": true }, "error": null }
```

Notifications

In-app notifications, email/push delivery preferences, and web push subscriptions. Base path /notifications.

GET /notifications

Lists notifications, newest first.

QUERY PARAMS

FIELD	NOTES
unreadOnly	"true"/"false"

RESPONSE · 200

```
{ "data": [ { "id": "n_1", "title": "Bill due soon", "severity": "warning", "read": false, "createdAt": "2026-07-29T01:00:00.000Z" } ], "error": null }
```

GET /notifications/preferences

Gets delivery preferences.

REQUEST

No params.

RESPONSE · 200

```
{ "data": { "emailEnabled": true, "pushEnabled": false, "minSeverity": "info" }, "error": null }
```

PATCH /notifications/preferences

Updates delivery preferences.

REQUEST BODY

```
{ "pushEnabled": true, "minSeverity": "warning" }
```

RESPONSE · 200

```
{ "data": { "emailEnabled": true, "pushEnabled": true, "minSeverity": "warning" }, "error": null }
```

GET /notifications/vapid-public-key

Returns the VAPID public key needed to register a browser push subscription.

REQUEST

No params.

RESPONSE · 200

```
{ "data": { "publicKey":  
  "BE162iUYgUivxIkv69yViEuiBIa..." }, "error":  
  null }
```

POST /notifications/push-subscriptions

Saves a browser push subscription (from the Push API).

REQUEST BODY

```
{ "endpoint":  
  "https://fcm.googleapis.com/fcm/send/...",  
  "keys": { "p256dh": "BN4G...", "auth":  
    "k9J2..." } }
```

RESPONSE · 201

```
{ "data": { "saved": true }, "error": null }
```

PATCH /notifications/:id/read

Marks a single notification as read.

REQUEST

No body.

RESPONSE · 200

```
{ "data": { "id": "n_1", "read": true },  
  "error": null }
```

POST /notifications/read-all

Marks all notifications as read.

REQUEST

No body.

RESPONSE · 200

```
{ "data": { "updated": 4 }, "error": null }
```

POST **/notifications/test**

Sends a test notification through the caller's configured channels (email/push), useful for verifying delivery setup.

REQUEST

No body.

RESPONSE · 200

```
{ "data": { "sent": true }, "error": null }
```

MODULE 16

Google Sheets

Connects a Google account for a persistent, user-owned Google Sheets ledger mirror.

Base path /google.

GET /google/connect

Returns the Google OAuth authorization URL to start connecting.

REQUEST

No params.

RESPONSE · 200

```
{ "data": { "url":  
  "https://accounts.google.com/o/oauth2/v2/auth?..  
  .." }, "error": null }
```

POST /google/exchange

Exchanges the OAuth authorization code for a connected Google account.

REQUEST BODY

```
{ "code": "4/0AY0e-g7..." }
```

RESPONSE · 200

```
{ "data": { "connected": true }, "error": null  
}
```

DELETE /google/disconnect

Disconnects the linked Google account.

REQUEST

No body.

RESPONSE · 200

```
{ "data": { "disconnected": true }, "error":  
  null }
```

GET /google/status

Connection status and Drive mirror metadata (spreadsheet URL, last sync).

REQUEST

No params.

RESPONSE · 200

```
{ "data": { "connected": true,
  "driveMirror": { "status": "active",
    "spreadsheetUrl":
      "https://docs.google.com/...",
    "lastSyncAt": "2026-07-29T01:50:00.000Z" } },
  "error": null }
```

POST /google/data-system-mode

Switches whether Google Sheets acts as a best-effort mirror or the Sheets-first primary record.

REQUEST BODY

```
{ "mode": "google_sheets_mirror" }
```

RESPONSE · 200

```
{ "data": { "mode": "google_sheets_mirror" },
  "error": null }
```

Security, Compliance & Admin

Abuse/risk signals, SOC 2 evidence reporting, and the admin user directory.

GET /security/risk

The caller's current abuse/risk score, built from recent failed logins, invalid tokens, and unusual export activity.

REQUEST

No params.

RESPONSE · 200

```
{ "data": { "score": 0, "level": "low",
"windowHours": 24,
  "recentEvents": [] }, "error": null }
```

GET /compliance/soc2/readiness

A live, evidence-backed SOC 2 technical control report (access binding, encryption, monitoring, audit logging, privacy controls). Does not represent formal certification.

REQUEST

No params.

RESPONSE · 200

```
{ "data": { "framework": "SOC 2",
"readinessStatus": "technical_baseline_ready",
  "certificationStatus": "not_certified",
  "controls": [ { "id": "CC6.1",
"trustServiceCriterion": "Logical access",
  "status": "implemented" } ],
  "pendingExternalControls": ["Independent SOC
2 auditor engagement"] } }, "error": null }
```

GET **/compliance/soc2/operations**

Operational SOC 2 checklist — the non-technical, process-level items (auditor engagement, policies, training) not covered by the technical readiness report.

REQUEST

No params.

RESPONSE · 200

```
{ "data": { "checklist": [ { "item": "Auditor engaged", "done": false } ] }, "error": null }
```

GET **/admin/users** **ROLE: ADMIN**

Lists all users. Restricted to accounts with the admin role.

REQUEST

No params.

RESPONSE · 200

```
{ "data": [ { "id": "ec92652a-...", "email": "jane@example.com", "role": "user", "createdAt": "2026-07-29T02:04:26.973Z" } ], "error": null }
```

View (Presentation API)

Sanitized, read-only presentation endpoints for browser display. Never return raw database or provider IDs — used for dashboard reads. Base path /view.

GET

/view/accounts

Sanitized account list for display. Page size is capped at 25 regardless of the requested limit.

QUERY PARAMS

FIELD	NOTES
page, limit	default 1 / 25, clamped to 25

RESPONSE · 200

```
{ "data": { "accounts": [ { "viewRef": "vr_9f2a...", "institutionName": "Chase Checking", "currentBalance": 1000.00 } ], "limit": 25 }, "error": null }
```

GET

/view/transactions

Sanitized transaction list for display, using the same filters as /transactions.

QUERY PARAMS

start_date, end_date, search, category, source, min_amount, max_amount, include_hidden, page, limit (clamped to 25).

RESPONSE · 200

```
{ "data": { "transactions": [ { "viewRef": "vr_1a2b...", "description": "Coffee Shop", "amount": -4.50, "date": "2026-07-15" } ], "limit": 25 }, "error": null }
```

MODULE 19

API Keys

Manage keys for the external Public API. Requires an active Pro or Elite plan. Base path `/api-keys`.

Plan: Pro or Elite required

GET `/api-keys` **PLAN: PRO**

Lists the caller's API keys (hashed — the raw key is shown only once, at creation).

REQUEST

No params.

RESPONSE · 200

```
{ "data": [ { "id": "key_1", "name": "Zapier integration",
  "scopes": ["transactions:read"], "createdAt": "2026-07-20T00:00:00.000Z",
  "expiresAt": null } ], "error": null }
```

POST `/api-keys` **PLAN: PRO**

Creates a new API key. The raw key value is returned only in this response — store it immediately.

REQUEST BODY

```
{ "name": "Zapier integration", "scopes": ["transactions:read"] }
```

RESPONSE · 201

```
{ "data": { "id": "key_1", "apiKey": "lo_live_9f2a1e6b...", "error": null }
```

DELETE `/api-keys/:id` **PLAN: PRO**

Revokes an API key immediately.

REQUEST

No body.

RESPONSE · 200

```
{ "data": { "revoked": true }, "error": null }
```

Public API (External Integrations)

Read-only endpoints for third-party integrations, authenticated by API key rather than a user session. Base path /public/v1.

Auth: X-LedgerOne-API-Key header — no bearer token or session nonce needed

GET

/public/v1/transactions

Lists transactions for the key's owner.

HEADERS & QUERY

FIELD	NOTES
X-LedgerOne-API-Key	required header
start_date, end_date, min_amount, max_amount, category, source, search, include_hidden	optional filters
page, limit	default 1 / 25

RESPONSE · 200

```
{ "data":
{
  "transactions": [ {
    "date":
    "2026-07-15",
    "description":
    "Coffee Shop",

    "amount":
    "-4.50",
    "category":
    "Dining"
  } ],
  "total": 1
},
"error":
null }
```

GET /public/v1/transactions/summary

Income/expense/net totals for the key's owner.

HEADERS & QUERY

FIELD	NOTES
X-LedgerOne-API-Key	required header
start_date, end_date	optional

RESPONSE · 200

```
{ "data": { "income": "2000.00", "expense": "4.50", "net": "1995.50" }, "error": null }
```

GET /public/v1/transactions/cashflow

Monthly cash flow for the key's owner.

HEADERS & QUERY

FIELD	NOTES
X-LedgerOne-API-Key	required header
months	default 6

RESPONSE · 200

```
{ "data": [ { "month": "2026-07", "net": "1995.50" } ], "error": null }
```

GET /public/v1/transactions/merchants

Top merchants by spend for the key's owner.

HEADERS & QUERY

FIELD	NOTES
X-LedgerOne-API-Key	required header
limit	default 6

RESPONSE · 200

```
{ "data": [ { "merchant": "Starbucks", "total": "5.75" } ], "error": null }
```