

PrivCo API — Developer Guide

Everything you need to call the PrivCo REST API: authentication, the identify → search → enrich workflow, every customer endpoint, search filters, usage allowances, and errors.

BASE URL — EVERY PATH BELOW IS RELATIVE TO IT

<https://api.privco.com/v3>

JSON over HTTPS · API-key auth

1 · QUICKSTART

Three calls from a name to financials

Send your key in the `x-api-key` header on every request. There is no SDK to install — any HTTP client works.

```
# 1. Resolve a company to its PrivCo id
curl -H "x-api-key: $PRIVCO_API_KEY" \
  "https://api.privco.com/v3/match?name=Stripe"
#   → data[0].profile_id = 44612, profile_type = "Company"

# 2. Pull the full profile
curl -H "x-api-key: $PRIVCO_API_KEY" \
  "https://api.privco.com/v3/profile/company/44612"

# 3. Multi-year revenue history
curl -H "x-api-key: $PRIVCO_API_KEY" \
  "https://api.privco.com/v3/revenue_financials/company/44612"
```

Keep the key server-side. It identifies your account and draws on your allowances. Store it in a secret manager or environment variable — never in browser code, a public repo, or a spreadsheet you share. If it leaks, ask us to replace it.

Brand-new key? A freshly issued key can return **403 Forbidden** for the first one to two minutes while it propagates. Wait and retry before troubleshooting.

2 · HOW THE DATA IS KEYED

Profile type + profile id

Every company and investor has a numeric **profile id**. Detail endpoints take it in the path together with the **profile type**:

PATH SEGMENT	VALUES
<code>{profileType}</code>	company or investor (lower case). Anything else returns 400 naming the valid types.
<code>{profileId}</code>	The <code>profile_id</code> from <code>/match</code> , <code>/identification/company</code> , or a search result. A non-numeric id returns 400 .

Unknown ids return 200, not 404. A profile id that does not exist comes back as **200** with `name: null` and empty fields. Check `data.name` before treating a response as a real company.

3 · ENDPOINTS

The 17 customer endpoints

Grouped by the job they do. Full request and response schemas are in the attached OpenAPI file (`privco_api_v3_openapi.json`) — import it into Postman, Insomnia or any OpenAPI tool.

ENDPOINT	WHAT IT RETURNS
Identify — turn what you know into a PrivCo id	
<code>GET /match?name=...</code> <code>GET /match?website=...</code>	Companies and investors by name, or by website domain (an alternative to name, not an extra filter). 20 per page; <code>?page=</code> to page.
<code>POST /identification/company</code>	Best-match scoring from any mix of name, <code>url</code> , address, phone, country/state, LinkedIn and more. Built for record matching.
Discover — screen the database	
<code>POST /company_search</code>	Companies by industry, keyword, location, revenue, employees, growth, funding, valuation, ownership.
<code>POST /funding_search</code>	Funding rounds by type, size, date, industry, location, investor type.

POST /deal_search

M&A deals by date, size, buyer/seller/target industry, PE/VC backing, target location.

POST /people_search

People by name, title, seniority, department, plus the company filters above. Returns a `hash` per person.

GET /industry_pics?query=

GET /industry_keyword?query=

Look up the numeric industry and keyword ids that the search filters take.

Enrich — details for one profile

GET /profile/{type}/{id}

GET /general_profile/{type}/{id}

The full profile: description, address, website, classification, industry, latest revenue, valuation, funding and more. The two paths return the same data.

`latest_revenues_is_estimated` says whether the latest revenue was reported (`false`) or estimated by PrivCo's model (`true`).

GET /revenue_financials/{type}/{id}

Revenue by year. Each figure has `is_estimated` : `false` = reported (company, filing, government or press source), `true` = PrivCo estimate, `null` = source not recorded.

GET /financials/{type}/{id}

Every financial line by year — revenue, EBITDA, valuation multiples, employees, each with `is_estimated` . Filter with `?title=` and `?year=` .

GET /vc_deals/{type}/{id}

Funding rounds with amounts, valuations and investors, newest first. Each round has a `funding_type` (e.g. `PPP` , `equity funding`). Add `?ppp=exclude` to drop PPP loans or `?ppp=only` to list them alone (default `include`).

GET /ma_deals/{type}/{id}

M&A history, with the profile's role (buyer, seller, target).

GET /people/{type}/{id}

Founders, executives, board members and investors associated with the profile.

Contacts — reveal one person

GET /company_contact/{hash}

GET /investor_contact/{hash}

Business email, direct and mobile numbers, LinkedIn, titles, for a `hash` returned by `/people_search` .

4 · IDENTIFICATION

Matching your records

Send what you have; more fields give a more confident match.

```
curl -X POST -H "x-api-key: $PRIVCO_API_KEY" \
-H "Content-Type: application/json" \
-d '{"name":"Stripe","url":"stripe.com","country":"US","state":"CA"}' \
"https://api.privco.com/v3/identification/company"
```

READ	MEANING
<code>mapped_profile_id</code> / <code>mapped_profile_type</code>	The PrivCo id and type to use with the Enrich endpoints.
<code>mapping_strength</code>	5 or more = strong match · 3–4 = weak, review it · 2 or less = no match.

Field names matter. The website goes in `url` (not `website`), and `country` / `state` take 2-letter codes (`US` , `CA`). Add `?company_only=true` to exclude investor profiles.

5 · SEARCH

Request body, filters and paging

All four searches take the same shape. Every part is optional; an empty body returns everything, newest or largest first.

```
curl -X POST -H "x-api-key: $PRIVCO_API_KEY" \
-H "Content-Type: application/json" \
"https://api.privco.com/v3/company_search" -d '{
  "filters": {
    "location": {"selection": [{"type": "state", "value": "New York"}]},
    "revenue": {"min": 10000000, "max": 100000000},
    "classification": {"privateOnly": true}
  },
  "sorting": {"field": "revenue", "order": "desc"},
  "page": 1
}'
```

COMPANY_SEARCH FILTER	SHAPE
<code>query</code>	Free text on the company name.

location	<code>{"selection":[{"type":"state","value":"New York"}]}</code> — types state (full name, not "NY"), city (optional <code>"radius":"25mi"</code>), city-name , region .
industry	<code>{"selection":[73]}</code> — ids from <code>/industry_pics</code> .
keyword	<code>{"selection":[281],"condition":"must"}</code> — ids from <code>/industry_keyword</code> ; must = all, should = any.
revenue, employee, funding, latestEbitda	<code>{"min":...,"max":...}</code> in whole USD (or headcount). Add <code>"includeMissing":false</code> to drop companies with no value.
latestValuation	<code>{"min":...,"max":...}</code> in USD.
yearFounded, latestFundingYear	<code>{"min":2015,"max":2024}</code>
growth	<code>{"revenue":{"min":20,"duration":1}}</code> — percent, over 1 or 3 years; <code>employee</code> works the same way.
inclusionExclusion	<code>{"vc":"include","pe":"exclude","subsidiary":"exclude"}</code>
classification	<code>{"privateOnly":true}</code>

SEARCH	ITS OWN FILTERS	SORT FIELDS
company	as above	name, state, employee, industry, yearFounded, totalFunding, revenueGrowthRate1, revenueGrowthRate3, hasContact, revenue , valuation , latestFundingYear
funding	<code>fundingTypes</code> <code>{"selection":["equity_funding_B"]}</code> , <code>totalInUsd</code> , <code>timestamp</code> , <code>industry</code> , <code>keyword</code> , <code>location</code> , <code>investorTypes</code>	companyName, timestamp, totalInUsd, fundingType

deal	<code>query</code> , <code>timestamp</code> , <code>totalInUsd</code> , <code>targetsIndustries</code> / <code>buyersIndustries</code> / <code>sellersIndustries</code> (<code>{"selection":</code> <code>[ids]}</code>), <code>isPeDeal</code> , <code>location</code>	<code>timestamp</code>
people	<code>queryFirstName</code> , <code>queryLastName</code> , <code>queryJobTitles</code> , <code>queryAssociated</code> (company name), <code>seniorityLevels</code> <code>["Cxo"]</code> , <code>departments</code> , plus the company filters	<code>yearLastSeen</code> , <code>firstName</code> , <code>lastName</code> , <code>seniorityLevel</code> , <code>department</code> , <code>associatedName</code> , <code>state</code> , <code>industry</code>

Funding types: `equity_funding_any`, `equity_funding_seed_angel`, `equity_funding_A` ...
`equity_funding_H`, `equity_funding_incubator_university`, `debt_funding_financing`, `grant or non-equityfunding`.

PPP loans. Paycheck Protection Program loans (2020–2021 government-backed bank loans) have their own funding type, `PPP`, which is not one of the selectable tokens above, so any `fundingTypes` selection leaves them out of `/funding_search`. For one company's rounds, use `GET /vc_deals/company/{id}?ppp=exclude` (or `only`); any other value returns **400**, and `meta.query` echoes what was applied.

Sorting company search. Companies without a value sort last in either order. `valuation` ranks only valuations dated in the last three years, like the `latestValuation` filter; add `"ignoreLimit":true` inside `latestValuation` to rank older valuations too. Search rows do not include the dollar figures — fetch them with the Enrich endpoints for the companies you keep.

Dates are Unix seconds. The funding and deal `timestamp` filter takes seconds — `{"min":1767225600}` is 1 Jan 2026. A millisecond value returns **400**.

Paging stops at 10,000 results. Searches return **100 results per page** and only **pages 1–100** can be retrieved, even when `meta.total_pages` is larger; page 101 returns **400** naming the last page. To collect a larger set, split it with narrower filters (by state, industry, revenue band or year) until each query's `meta.total_results` is 10,000 or fewer.

Misspelled fields are ignored, not rejected. Unknown keys in a request body are dropped silently, so a typo returns unfiltered results. Compare `meta.total_results` with and without a new filter when you first use it. An unknown sort field is ignored too (results come back in the default order). Other invalid *values* return **400** listing the allowed ones.

How usage is counted

Your account has four allowances, usually monthly; they reset at the start of each period. Your welcome email lists your numbers, and `GET /usage` shows them live.

ALLOWANCE	WHAT DRAWS ON IT
Unique profiles	Each distinct company or investor you open with an Enrich endpoint — counted once per period, however many times you call it.
Finance	Financial-history calls (<code>/financials</code> , <code>/revenue_financials</code>).
Points	Each search page that returns results (company, funding, deal and people search) costs 1 point ; a failed or empty search is free. Other Enrich calls are metered per call.
Deals	Deal-history calls, metered per call.

`/match` , the four searches and the industry look-ups do not draw on your unique-profile allowance — identify and screen, then enrich the profiles you need. When an allowance runs out, calls that need it return **403** with `code` `quota_exceeded` until the next period begins. The body names the allowance, how much is used, its limit and the date it resets:

```
{
  "statusCode": 403,
  "code": "quota_exceeded",
  "message": "Unique Profile Quota exceeded",
  "allowances": [
    {
      "name": "unique_profiles",
      "used": 500,
      "limit": 500,
      "period": "monthly",
      "resets_on": "2026-11-01"
    }
  ],
  "subscription_ends": "2027-01-09",
  "next_step": "This account's API allowance for the period is used up. It resets on 2026-11-01.",
  "requestId": "..."
}
```

`name` is one of `unique_profiles` , `points` , `finance` or `deals` ; `resets_on` is `null` when the allowance does not reset before your subscription ends.

Check your usage

```
curl -H "x-api-key: $PRIVCO_API_KEY" "https://api.privco.com/v3/usage"
```

Free to call. Returns, for each allowance, its `refresh_period` , `total_quota` , `used_this_period` and `remaining_this_period` (`data.unique_profile` ; `data.calls.general_profile` is the points allowance, plus `financials` and `deals`), with `subscription_start` and `subscription_end` .

Status codes

STATUS	MEANING & WHAT TO DO
200 / 201	Success. POST searches answer 201. Results are in <code>data</code> , paging in <code>meta</code> .
400	Invalid request; <code>error.message</code> says which field and lists the allowed values. Also returned for a search page beyond the 10,000-result window or a millisecond timestamp (see §5).
401	The key reached us but has no API access: no API subscription (<code>Subscription not found</code>), or the endpoint is not in your plan (<code>Un-Authorized Access, Permission Denied</code>).
403	Three cases, told apart by the body. <code>code</code> <code>quota_exceeded</code> : an allowance is used up (see §6; retrying will not help until <code>resets_on</code>). <code>code</code> <code>subscription_expired</code> : your API subscription is outside its term (<code>subscription.start_date</code> / <code>end_date</code>). A bare <code>{"message": "Forbidden"}</code> : the <code>x-api-key</code> is missing or unrecognised. New keys: wait two minutes.
429	Too many requests for your plan's rate limit. Back off and retry.
500	Server error; retry with backoff.

Reporting a problem? Include the `requestId` from the error body — it lets us find your request in our logs.

8 · ALSO AVAILABLE

PrivCo in AI assistants

The same data is available as 17 tools for Claude, ChatGPT, Microsoft Copilot and any MCP client at `https://mcp.privco.com/mcp` — sign in with your PrivCo account, no key needed (included with List Builder and Power plans; it draws on the same allowances as your API key). For a local MCP server that uses this API key: `npm install -g privco-data-mcp` . Setup guides for each assistant are available on request.

Questions, a higher allowance, or a replacement key? Contact your PrivCo account manager or email sales@privco.com.