Generating an API Key and Making Your First Request

Last updated: May 28, 2026 · 7 min read · Article ID DOC-401

This guide walks you through the steps to authenticate to the Trailmark API: generating an API key, sending an authenticated request, and verifying it works. After this guide you'll be able to call any Trailmark API endpoint from your own code or from a tool like `curl`, Postman, or Insomnia.

Before you start

- You have a Trailmark workspace on a plan that includes API access (Team or higher).
- You are signed in as a member with the *Developer* or *Admin* role. Other roles cannot create API keys.
- You have a terminal with `curl`, or any language environment that can make HTTPS requests.

Step 1 — Generate an API key

1. Sign in to `app.trailmark.com` and go to **Settings** → **Developers** → **API Keys**.
2. Click **New API Key**.
3. Give the key a descriptive name (e.g. `internal-reporting-pipeline`). Names help you audit usage and revoke unused keys later.
4. Choose a scope. We recommend the smallest scope that fits your use case:
   - `read:workspace` — read-only access to workspace data.
   - `read:write:workspace` — read and write for non-billing resources.
   - `admin:full` — everything, including billing. Use only for admin tooling.
5. Click **Create key**. The key value (a string beginning with `tm_live_`) is shown once. Copy it immediately and store it in a secrets manager.

> **Important:** Trailmark never displays a generated key a second time. If you lose it, you must rotate it and update every system that used it. Store new keys in your secrets manager before closing the dialog.

Step 2 — Authenticate a request

Every Trailmark API request is authenticated with a `Bearer` token in the `Authorization` header. The base URL for the API is `https://api.trailmark.com/v1`.

cURL

```
curl https://api.trailmark.com/v1/workspaces/me \
  -H "Authorization: Bearer tm_live_..." \
  -H "Accept: application/json"
```

Python (with the requests library)

```
import os, requests

resp = requests.get(
    "https://api.trailmark.com/v1/workspaces/me",
    headers={"Authorization": f"Bearer {os.environ['TRAILMARK_API_KEY']}"},
    timeout=10,
)
resp.raise_for_status()
print(resp.json())
```

Node.js (with fetch)

```
const res = await fetch("https://api.trailmark.com/v1/workspaces/me", {
  headers: { Authorization: `Bearer ${process.env.TRAILMARK_API_KEY}` },
});
if (!res.ok) throw new Error(`HTTP ${res.status}`);
console.log(await res.json());
```

A successful call returns a JSON object describing your workspace:

```
{
  "id": "ws_3kf9...",
  "name": "Acme Sales",
  "plan": "team",
  "members_count": 14,
  "created_at": "2025-11-02T18:01:12Z"
}
```

Step 3 — Handling common errors

`401 Unauthorized`
: The API key is missing, malformed, or invalid. Re-check the `Authorization` header — it should be exactly `Bearer tm_live_...` with one space.

`403 Forbidden`
: The key is valid but the scope is insufficient. Generate a new key with the required scope or use an existing key with broader scope.

`404 Not Found`
: The resource does not exist in your workspace, or the workspace identifier in the URL is wrong.

`429 Too Many Requests`
: You've exceeded the rate limit. The response includes `Retry-After` in seconds. Back off and retry.

`5xx`
: Transient server-side error. Retry with exponential backoff. If it persists, check `status.trailmark.com`.

Step 4 — Rate limits

The default rate limit is 600 requests per minute per workspace. Every response includes the following headers so your client can pace itself:

- `X-RateLimit-Limit` — your current limit (600).
- `X-RateLimit-Remaining` — requests remaining in the current window.
- `X-RateLimit-Reset` — Unix timestamp when the window resets.

If you have a legitimate use case that needs more, contact `developers@trailmark.com`.

Step 5 — Rotating and revoking keys

1. From **Settings** → **Developers** → **API Keys**, find the key by name.
2. Click **Rotate** to issue a new value (the old value remains active for 24 hours so you can swap in production), or **Revoke** to invalidate the key immediately.
3. Update every system that uses the key before the 24-hour grace window expires.

> **Tip:** Set a calendar reminder to rotate production keys at least every 90 days. Workspaces on the Enterprise plan can enforce this automatically.

Next steps

- Read the full **API Reference** at `developers.trailmark.com/api`.
- Try the **Webhooks** guide to receive real-time events.

- Use the official SDKs for Python, Node.js, and Go — they handle retries, pagination, and rate limits for you.

Related articles

- OAuth 2.0 — building a Trailmark app for other workspaces
- Webhooks reference
- Idempotency keys
- SDK quickstarts