

PARTNER GUIDE

The Num eSIM API

How to start, how it works and how it is billed, followed by the complete API reference and terms. Everything a travel company or reseller needs to issue eSIMs from its own product.

CONTENTS

What is in this guide

PART 1 · THE PROCESS

At a glance

From sign-up to live

Sandbox and live

What your customer receives

Billing, credit and cancellation

Being ready for production

PART 2 · API REFERENCE

Authentication, plans, orders, webhooks, errors

PART 3 · TERMS

Num API Terms

PART 1 · THE PROCESS

At a glance

The Num API lets your product issue prepaid travel eSIMs. Your system asks for a plan; the response is the eSIM: an activation code, a ready-made QR image and links that open the phone's own installer.

Num can also email it to your customer under your brand name.

190+

destinations, plus
regional and global
plans

1 call

to issue an eSIM with
its QR image

\$0

setup fee, monthly fee
or minimum

USD

every price, charge and
balance

Why partners choose it: there is no contract to negotiate before you can start, no minimum volume, and nothing to build around the API. The sandbox is yours the moment you sign up.

From sign-up to live

1 Sign up

You · 1 minute

Create an account at numesim.uk/business with your work email and agree to the API terms. Your sandbox exists at once, with \$100 of test credit. There is no approval for sandbox.

2 Build in sandbox

You

Create a test key (numk_test_...) and integrate. Sandbox orders return demo eSIMs and cost nothing. Test credit refills itself.

3 Prove it is production-ready

You

Simulate a short balance, a failed order and a delayed eSIM. Connect a webhook. The go-live checklist in your console ticks itself as your integration exercises each case.

4 Request live access	You, then Num
Press one button in the console. We agree your pricing and switch live on for your account.	
5 First credit by invoice	Num sales, then you
We send an invoice in US dollars. When it is paid, our team adds the credit to your balance and it appears on your statement with the invoice number.	
6 Go live	You
Create a live key (numk_live_...) and change it in your configuration. Nothing else in your integration changes.	
7 Top up when you need to	You, then Num
Ask for another invoice when your balance runs low. A low-balance webhook warns you before orders are refused.	

Sandbox and live

	SANDBOX	LIVE
Available	Immediately on sign-up	When Num switches it on for your account
Credit	\$100 of test credit, refilled automatically	Real prepaid credit, added against a paid invoice
eSIMs	Demo eSIMs. Not real.	Real eSIMs on real networks
Cost to you	Nothing	The wholesale price of each order
Behaviour	Identical: the same endpoints, responses, prices, errors and webhooks	

Sandbox eSIMs are not real. A sandbox activation code looks right and scans as a QR code, but it will not install on a phone and never touches a mobile network. Every sandbox order is marked `"demo": true`. Never give a sandbox code to a traveller.

What your customer receives

You choose how the eSIM reaches your customer. Both options install the same eSIM.

OPTION	HOW IT WORKS
Num emails it for you	Pass the customer's email address on the order. Num sends the eSIM with your brand name as the sender name and heading, the QR code, the activation code, install buttons and install steps. Replies go to your support address.

OPTION	HOW IT WORKS
--------	--------------

You deliver it yourself	Every order returns a hosted QR image link you can place in your own email or confirmation page, the activation code, and install links for iPhone and Android.
--------------------------------	---

Num also emails **you** whenever an eSIM is issued on your account, with the plan, the charge and your balance. Brand name, reply-to address and notification address are set in the console under Branding & email. In sandbox, the customer email is never sent to the traveller: a preview goes to your own verified login address instead.

Billing, credit and cancellation

Currency	US dollars, for every price, charge and balance, whatever the destination.
Model	Prepaid credit. Num invoices you; credit is added to your balance when the invoice is paid.
What an eSIM costs	The wholesale price shown for that plan in the catalogue when you order it, less any discount agreed for your account.
Fees	None. No setup fee, no monthly fee, no minimum spend.
If your balance is short	The order is refused and nothing is issued. Nothing is provided on credit you have not paid for.
Refunds	Credit is prepaid and non-refundable , and an issued eSIM is not refunded.
Cancelling an eSIM	You can cancel an eSIM at any time. It stops working. Cancelling does not return credit.
If an eSIM cannot be issued	That charge is reversed to your balance automatically.
Num's rights	Num may cancel or suspend an eSIM or an account at any time, as set out in the API Terms (Part 3).
Your records	A full statement in the console and by API: every credit, order and reversal, with a running balance.

Being ready for production

An integration that passes in sandbox is ready for live, because sandbox can produce every outcome live can. Your console keeps this checklist and ticks each item the first time your integration does it.

CHECK	HOW IT IS COMPLETED
Create a test API key	Console → API keys
Fetch the catalogue	GET /v1/catalog
Place a sandbox order	POST /v1/orders returns active
Read an order back	GET /v1/orders/{id}
Retry an order safely	The same Idempotency-Key sent twice
Handle insufficient credit	Header Num-Simulate: insufficient_credit
Handle a failed order	Header Num-Simulate: provisioning_failed
Handle a delayed eSIM	Header Num-Simulate: pending
Set a webhook endpoint	Console → Webhooks
Receive a signed webhook	Your endpoint answers 2xx
Check your balance or statement	GET /v1/balance or GET /v1/ledger

Num Corporate API — v1

Issue travel eSIMs from your own product. One call returns an activation code, a hosted QR image and one-tap install links for your customer.

This page is the complete reference, and it is written so that an integration which passes in sandbox is ready for production. Section 10 lists the cases your code must handle, how to trigger each one in sandbox, and a checklist we track for you automatically.

Base URL:

```
https://numesim.uk/v1
```

JSON in, JSON out, UTF-8. All money is **USD**, always, whatever country the plan is for.

1. Quickstart

1. **Sign up** at numesim.uk/business. Your sandbox account exists the moment you do: no approval, \$100 of test credit already on it.
2. **Create a test key** in the console under API keys. It starts `numk_test_`.
3. **Price a destination**, then **place an order**:

```
curl "https://numesim.uk/v1/catalog?country=JP" \
  -H "Authorization: Bearer numk_test_..."

curl -X POST https://numesim.uk/v1/orders \
  -H "Authorization: Bearer numk_test_..." \
  -H "Idempotency-Key: booking-48213" \
  -H "Content-Type: application/json" \
  -d '{"planId":"np_3f9a61c0d2e47b18aa","country":"JP","funding":"credit"}'
```

The response is the eSIM: an `activationCode`, a `qrUrl` you can put straight into an email, and its `iccid`.

2. Sandbox and live

There are two environments. The key prefix chooses between them, and they never mix.

	SANDBOX	LIVE
Key prefix	<code>numk_test_</code>	<code>numk_live_</code>
Available	Immediately, on sign-up	When Num switches it on for your account

	SANDBOX	LIVE
Credit	\$100 of test credit, refilled automatically	Real prepaid credit, added by Num against a paid invoice
eSIMs	Demo eSIMs. Not real.	Real eSIMs on real networks
Cost to you	Nothing, ever	The <code>wholesale</code> price of each order
Behaviour	Identical to live	—

Sandbox eSIMs are not real. A sandbox order never contacts a mobile network. Its activation code looks like `LPA:1$demo.esim.test$...`, its ICCID starts `89000000000`, and every sandbox order carries `"demo": true`. A demo code will scan as a QR but will not install on a phone. Never show a sandbox code to a traveller; branch on `demo` if your test and production systems share code.

Sandbox is always usable. Test credit refills itself when an order would run it dry, so an integration is never stopped by a balance that was never real. To rehearse an empty balance, use the simulation header in section 10.

Everything else is the same in both: the endpoints, the response shapes, the prices, the errors and the webhooks. Going live is a change of key, not a change of code.

3. Authentication

A bearer key on every request:

```
Authorization: Bearer numk_test_9f2c...
```

Keys are created in the business console and shown **once**. We store only a SHA-256 hash, so a lost key cannot be recovered, only replaced. Create one key per system so you can revoke one without stopping the others. Revocation takes effect within a minute.

There is no CORS. These endpoints are server-to-server. A key that reaches browser or mobile-app code is a public key, and anyone reading the page can spend your credit. Call the API from your backend.

4. Plans and prices

```
GET /v1/catalog?country=JP
```

`country` is an ISO 3166-1 alpha-2 code.

```
{
  "country": "JP",
  "currency": "USD",
  "plans": [
    {
      "planId": "np_3f9a61c0d2e47b18aa",
      "name": "Japan 1GB 7Days",
      "country": "JP",
      "dataMb": 1024,
      "days": 7,
      "topUp": true,
      "wholesale": { "amountMinor": 155, "amount": "1.55", "currency": "USD" },
      "retail": { "amountMinor": 186, "amount": "1.86", "currency": "USD" }
    }
  ]
}
```

FIELD	MEANING
<code>planId</code>	Num's identifier for the plan. Stable: the same plan always has the same id. Send it back when ordering
<code>wholesale</code>	What an order of this plan costs you
<code>retail</code>	A suggested price for your customer, from the markup you set in the console. Yours to use or ignore
<code>dataMb</code> , <code>days</code>	Data allowance and validity

Rules that keep an integration correct in production:

- **Money is integers in minor units.** `155` is \$1.55. Never parse `amount` as a float for arithmetic; it is there for display.
- **Prices move.** Read the catalogue rather than hard-coding prices, and cache it for no longer than an hour. An order is charged the wholesale price in force when it is placed.
- **Plans come and go.** A `planId` you stored last month may no longer be sold; ordering it returns `404`. Refresh from the catalogue.
- Countries and packages you have switched off in the console do not appear.

5. Orders

`POST /v1/orders`

Requires an `Idempotency-Key` header (section 7).

```
{ "planId": "np_3f9a61c0d2e47b18aa", "country": "JP", "funding": "credit" }
```

`201 Created`:

```
{
  "id": "NAH9XIRi4HjCaQwsrfc5",
  "status": "active",
  "funding": "credit",
  "mode": "sandbox",
  "demo": true,
  "country": "JP",
  "planId": "np_3f9a61c0d2e47b18aa",
  "planName": "Japan 1GB 7Days",
  "charged": { "amountMinor": 155, "amount": "1.55", "currency": "USD" },
  "activationCode": "LPA:1$demo.esim.test$NAH9XIRI4HJCAQWSRFC5",
  "iccid": "89000000000NAH9XIRi4",
  "qrUrl": "https://numesim.uk/v1/qr/...png",
  "install": null,
  "customerEmail": null,
  "email": { "customer": null, "partner": "sent", "note": null },
  "createdAt": "2026-10-04T09:12:44.000Z"
}
```

Optional request fields: `customerEmail`, `customerName` and `sendEmail`, to have Num email the eSIM to your traveller. See section 6.

Order status

STATUS	MEANING	WHAT YOU DO
<code>active</code>	The eSIM is issued. <code>activationCode</code> and <code>qrUrl</code> are present	Give it to your customer
<code>pending</code>	Paid for, and the network has not released the profile yet. Usually seconds, occasionally minutes	Wait for the <code>order.active</code> webhook, or poll <code>GET /v1/orders/{id}</code>
<code>failed</code>	It could not be issued. The charge has been reversed to your balance	Tell your customer; order again if you wish
<code>cancelling</code>	A cancellation is being carried out	Nothing; it becomes <code>cancelled</code>
<code>cancelled</code>	The eSIM has been stopped. Credit is not returned	—

Always handle `pending`. Most orders are `active` in the response. Some are not, and code that assumes an activation code is always present will break on the day it matters. Section 10 shows how to trigger a pending order in sandbox.

`GET /v1/orders` **and** `GET /v1/orders/{id}`

`GET /v1/orders?limit=50` returns your newest orders first (maximum 100). `GET /v1/orders/{id}` returns one, and is how you poll a pending order. Poll no more often than every five seconds; prefer the webhook.

`POST /v1/orders/{id}/cancel`

Stops an eSIM. Returns the order, with status `cancelled` (sandbox) or `cancelling` then `cancelled` (live).

Cancelling does not return credit. Credit is prepaid and non-refundable; a cancellation ends the eSIM, it does not undo the purchase. Cancelling an order that is already cancelled is not an error.

6. Giving the eSIM to your customer

An active order gives you three ways to deliver it. Use whichever fits your product; they all install the same eSIM.

FIELD	WHAT IT IS	USE IT FOR
<code>qrUrl</code>	A hosted PNG of the activation QR code. Needs no API key; the link itself is unguessable	Emails, confirmation pages, PDFs: put it in an <code></code> tag
<code>activationCode</code>	The <code>LPA:1\$...</code> string the QR encodes	Rendering your own QR, or manual entry on the phone
<code>install.ios</code> , <code>install.android</code>	A link that opens the phone's own eSIM installer	A button shown on the phone that will use the eSIM

`GET /v1/orders/{id}/qr` returns the same PNG with your bearer key, if you would rather fetch the image yourself and serve it from your own domain.

Notes:

- The QR is scanned by **another** screen. A traveller reading your email on the phone they want to install on cannot scan it; give them the `install` link or the activation code as well.
- `install` is `null` in sandbox: a demo code would open the installer and then fail in front of the user.
- An activation code can normally be installed once. Treat `qrUrl` like the eSIM itself and send it only to the customer it belongs to.

Let Num email it for you

Pass the traveller's address on the order and Num sends the eSIM to them, under **your** brand name:

```
{ "planId": "np_3f9a61c0d2e47b18aa", "country": "JP", "funding": "credit",  
  "customerEmail": "traveller@example.com", "customerName": "Alex" }
```

The email carries your brand name as its heading and sender name, the QR code, the activation code, install buttons and install steps. Replies go to the support address you set. Set your brand name and addresses in the console under Branding & email, and send yourself a preview from there.

FIELD	MEANING
<code>customerEmail</code>	Optional. Where to send the eSIM

FIELD	MEANING
<code>customerName</code>	Optional. Used in the greeting
<code>sendEmail</code>	Optional, default <code>true</code> . Send <code>false</code> to store the address without emailing

The order reports what happened:

```
"customerEmail": "traveller@example.com",
"email": { "customer": "sent", "partner": "sent", "note": null }
```

EMAIL.CUSTOMER	MEANING
<code>sent</code>	Emailed to the traveller (live)
<code>preview_sent</code>	Sandbox: a preview went to your login email, not to the traveller
<code>skipped</code>	Not sent; <code>note</code> says why
<code>null</code>	No <code>customerEmail</code> was given

- **Sandbox never emails a traveller.** The address you pass is recorded, and a preview of the email is sent to your own login address instead, once you have verified it. This is deliberate: a sandbox eSIM is not real, and a test must not be able to reach a real person.
- The email is sent when the order becomes `active`. For a `pending` order that is when the eSIM is released, not when you placed the order.
- `POST /v1/orders/{id}/email` sends it again. Put a `customerEmail` in the body to correct a mistyped address first.
- The email is sent by Num. Your brand name is shown as plain text; logos and custom HTML are not supported.

You are told about every purchase too. Each time an eSIM is issued on your account, Num emails your notification address with the plan, the charge and your balance. Turn this off in the console if you would rather rely on the `order.active` webhook.

Installation, for your help pages: Settings → Mobile Service → Add eSIM → Use QR Code, then turn on Data Roaming for the new line.

7. Idempotency

`POST /v1/orders` requires an `Idempotency-Key` header: any string that is unique to the purchase you are making, such as your own booking reference.

Send the same key again and you get the **original order back** with `200`, not a second eSIM and not a second charge. This is what makes a retry safe: if a request times out and you do not know whether it worked, send it again with the same key.

- Use one key per purchase, never one per attempt.

- Do not reuse a key for a different plan; the original order is what returns.

8. Webhooks

Set one https endpoint per environment in the console under Webhooks. We send a `POST` with a JSON body whenever something happens that you would otherwise have to poll for.

Events

EVENT	WHEN	DATA
<code>order.active</code>	An eSIM is issued, straight away or after being pending	<code>{ order }</code>
<code>order.failed</code>	An order could not be issued; the charge was reversed	<code>{ order }</code>
<code>order.cancelled</code>	An eSIM was cancelled, by you or by Num	<code>{ order, cancelledBy }</code>
<code>credit.added</code>	Num added credit to your balance	<code>{ amount, balance, reference }</code>
<code>balance.low</code>	An order took your balance below your threshold (default \$20)	<code>{ balance, threshold }</code>
<code>ping</code>	You pressed "Send test event"	<code>{ message }</code>

Payload

```
{
  "id": "evt_8tQ2mWc1YbN4",
  "type": "order.active",
  "mode": "live",
  "createdAt": "2026-10-04T09:12:46.000Z",
  "data": { "order": { "id": "NAH9XIRi4HjCaQwsrfc5", "status": "active", "...": "..." } }
}
```

`order` is exactly the object `GET /v1/orders/{id}` returns.

Verifying the signature

Every delivery carries:

```
Num-Signature: t=1791112366,v1=5f2b...e9
Num-Event: order.active
Num-Delivery: 8tQ2mWc1YbN4
```

`v1` is the HMAC-SHA256, in hex, of the string `{t}.{raw request body}` using your endpoint's signing secret (shown once when you save the endpoint; saving again rotates it). Verify it before trusting

anything in the body, compare in constant time, and reject a timestamp more than five minutes old.

Node:

```
const crypto = require("crypto");

function verify(rawBody, header, secret) {
  const parts = Object.fromEntries(header.split(",").map((p) => p.split("=")));
  if (Math.abs(Date.now() / 1000 - Number(parts.t)) > 300) return false;
  const expected = crypto.createHmac("sha256", secret)
    .update(parts.t + "." + rawBody).digest("hex");
  const a = Buffer.from(expected), b = Buffer.from(parts.v1 || "");
  return a.length === b.length && crypto.timingSafeEqual(a, b);
}
```

Python:

```
import hashlib, hmac, time

def verify(raw_body: bytes, header: str, secret: str) -> bool:
    parts = dict(p.split("=", 1) for p in header.split(","))
    if abs(time.time() - int(parts["t"])) > 300:
        return False
    signed = parts["t"].encode() + b"." + raw_body
    expected = hmac.new(secret.encode(), signed, hashlib.sha256).hexdigest()
    return hmac.compare_digest(expected, parts.get("v1", ""))
```

PHP:

```
function verify(string $rawBody, string $header, string $secret): bool {
    parse_str(str_replace(',', '&', $header), $parts);
    if (abs(time() - (int) $parts['t']) > 300) return false;
    $expected = hash_hmac('sha256', $parts['t'] . '.' . $rawBody, $secret);
    return hash_equals($expected, $parts['v1'] ?? '');
}
```

Use the **raw** request body, byte for byte. Parsing the JSON and re-serialising it changes the bytes and the signature will not match.

Delivery rules

- Answer with any `2xx` within **5 seconds**. Do the real work after you have answered; a slow handler is treated as a failure.
- A failed delivery is retried after about 1 minute, 5 minutes, 30 minutes, 2 hours, 6 hours and 12 hours, then abandoned. Seven attempts in all.
- **The same event can arrive more than once.** Deduplicate on the event `id` and make your handler safe to run twice.
- Events can arrive out of order. Trust the `status` in the payload, or fetch the order, rather than the order in which events reached you.
- Redirects are not followed. Give us the final URL.

- The console shows every delivery, its attempts and the last error.

Webhooks are a convenience, not the record. `GET /v1/orders/{id}` is always authoritative, and an integration should still work, more slowly, with the endpoint switched off.

9. Errors

Standard HTTP status with a JSON body:

```
{ "error": "Insufficient credit.", "detail": { "balanceMinor": 40, "requiredMinor": 155, "shortfallMinor": 115 } }
```

STATUS	MEANING	RETRY?
400	The request is malformed: a missing header, a bad country code	No. Fix the request
401	Missing, unknown or revoked key	No
402	Not enough credit for this order. Nothing was issued	After credit is added
403	The key may not do this: live is not enabled, the account is suspended, the country is switched off	No
404	No such order, or the plan is not sold	No. Refresh the catalogue
409	The order is in a state that does not allow this	No
429	Rate limit reached	Yes, after a short wait
500	Something went wrong on our side	Yes, with the same Idempotency-Key
424	The eSIM could not be issued. The charge has been reversed	Yes, with a new Idempotency-Key

Retry with exponential backoff, starting at one second. Any other `5xx`, including an HTML error page from the network in front of the API, means the same as `500`. On a timeout or a `5xx` you do not know whether the order exists, which is exactly what the idempotency key is for.

10. Building for production

Sandbox lets you rehearse every outcome production can produce. Send the `Num-Simulate` header on `POST /v1/orders` with a test key:

NUM-SIMULATE	WHAT HAPPENS	WHAT YOUR CODE MUST DO
<code>insufficient_credit</code>	<code>402</code> , nothing issued, nothing charged	Stop selling, alert whoever tops up, tell the customer plainly
<code>provisioning_failed</code>	The order is charged, fails, and is reversed. <code>424</code> , then <code>order.failed</code>	Not deliver anything; retry with a new key or offer another plan
<code>pending</code>	The order returns <code>pending</code> with no activation code, and becomes <code>active</code> about 15 seconds later with <code>order.active</code>	Wait for the webhook or poll, then deliver

The header is rejected with `400` on a live key. Nothing a client sends can make a real order fail.

Go-live checklist

We tick each of these the first time your account does it in sandbox. See it in the console under Go live, or fetch it:

```
curl https://numsim.uk/v1/readiness -H "Authorization: Bearer numk_test_..."
```

```
{ "liveEnabled": false, "done": 7, "total": 11,
  "items": [ { "key": "catalog", "label": "Fetch the catalogue", "done": true, "at": "2026-10-04T09:10:02.000Z" } ] }
```

CHECK	HOW IT IS COMPLETED
Create a test API key	Console → API keys
Fetch the catalogue	<code>GET /v1/catalog</code>
Place a sandbox order	<code>POST /v1/orders</code> returns <code>active</code>
Read an order back	<code>GET /v1/orders/{id}</code>
Retry an order safely	The same <code>Idempotency-Key</code> sent twice
Handle insufficient credit	<code>Num-Simulate: insufficient_credit</code>
Handle a failed order	<code>Num-Simulate: provisioning_failed</code>
Handle a delayed eSIM	<code>Num-Simulate: pending</code> , then the order reaches <code>active</code>
Set a webhook endpoint	Console → Webhooks
Receive a signed webhook	Your endpoint answers <code>2xx</code>
Check your balance or statement	<code>GET /v1/balance</code> or <code>GET /v1/ledger</code>

The checklist records that each path was exercised. It cannot see whether your handler did the right thing, so also confirm for yourself:

- Keys live in a secret store on your server, never in a browser, an app or a repository.
- You store our order `id` against your own booking.
- A sandbox order (`"demo": true`) can never reach a real customer.
- If Num emails your travellers, you have sent yourself a preview and your brand name and reply-to address are right.
- Webhook signatures are verified and events are deduplicated by `id`.
- Something alerts a person when the balance is low, before orders start returning `402`.
- Your customer is told what to do if the eSIM does not arrive within a minute.

Going live

1. Finish the checklist and press **Request live access** in the console.
2. We agree pricing with you, switch live on for your account, and send an invoice for your first credit.
3. When the invoice is paid we add the credit. It appears on your statement and in `GET /v1/balance` at once, and `credit.added` fires.
4. Create a `numk_live_` key, set your live webhook endpoint, and change the key in your configuration. Nothing else changes.

11. Account, balance and statement

`GET /v1/me`

```
{ "companyName": "Northwind Travel", "mode": "sandbox", "liveEnabled": false,
  "discountPercent": 10, "currency": "USD",
  "balance": { "amountMinor": 9845, "amount": "98.45", "currency": "USD" } }
```

`GET /v1/balance`

```
{ "mode": "live", "balance": { "amountMinor": 48210, "amount": "482.10", "currency": "USD" } }
```

`GET /v1/ledger?limit=50`

Your statement, newest first: every credit, order and reversal with the balance after it.

```
{ "mode": "live", "entries": [
  { "id": "9dK...", "type": "order", "amountMinor": -155, "amount": "-1.55", "currency": "USD",
    "balanceAfter": "482.10", "ref": "NAH9XIRi4HjCaQwsr5", "note": "Japan 1GB 7Days · JP",
    "createdAt": "2026-10-04T09:12:44.000Z" } ] }
```

TYPE	MEANING
topup	Credit added by Num against an invoice. <code>ref</code> is the invoice number
order	An eSIM you ordered. <code>ref</code> is the order id
refund	A charge reversed because an eSIM could not be issued
adjustment	A correction by Num, or sandbox test credit

12. Billing and cancellation

Currency	USD. Every price, charge and balance
Model	Prepaid credit. Num invoices you; credit is added when the invoice is paid
Topping up	Ask your account manager for an invoice. There is no card payment in the console
What an order costs	The <code>wholesale</code> price in the catalogue when the order is placed
Fees	None. No setup fee, no monthly fee, no minimum
If the balance is short	The order is refused with <code>402</code> and nothing is issued
If an eSIM cannot be issued	The charge is reversed to your balance automatically
Refunds	Credit is non-refundable , and an issued eSIM is not refunded
Cancelling	You can cancel an eSIM at any time. Cancelling does not return credit
Num's rights	Num may cancel or suspend an eSIM or an account at any time, as set out in the API Terms

By creating an account you agree to the [Num API Terms](#).

13. Rate limits

120 requests per minute, per key. Over that returns `429`; wait and retry. The catalogue changes slowly, so cache it rather than fetching it per page view.

14. Known limits, stated plainly

- **Hosted checkout is sandbox only.** `funding: "checkout"`, a payment page for your traveller, can be rehearsed in sandbox and is not available in live. Live orders use `funding: "credit"`.

- **Live cancellation is not instant.** A live order moves to `cancelling` while the eSIM is stopped, then to `cancelled`. Sandbox cancels at once.
- **One endpoint per environment.** A single webhook URL for sandbox and a single one for live.
- **Top-ups of an existing eSIM** are not exposed in this version. `topUp` tells you whether a plan supports them.
- **One currency.** Everything is USD. You may charge your customer in any currency you like; what you pay us is USD.

Num API Terms

Version 2026-10-04.

These terms apply to every business that creates an account to use the Num API or the Num business console. By creating an account you agree to them on behalf of your company. They are written to be read; if anything is unclear, ask us before you rely on it.

"Num", "we" and "us" mean WorkersLab LLC, 30 N Gould St Ste R, Sheridan, WY 82801-6317, United States. "You" means the company whose account it is.

1. What you get

An account with a sandbox environment for building and testing, and, once we switch it on for you, a live environment in which orders issue real travel eSIMs.

Sandbox is not real. Sandbox eSIMs are demonstration data. They do not connect to any network and must not be given, sold or shown to a customer as a working eSIM. Sandbox credit has no monetary value.

2. Your account

- You are responsible for everything done with your account and your API keys. Keep keys secret and on your own servers. Tell us at once if a key may have leaked; we will revoke it.
- The information you give us about your company must be true, and you must keep your contact address current.
- Live access is granted at our discretion. We may ask what you intend to sell and to whom before granting it, and we may decline.

3. Prices and payment

- All prices, charges and balances are in **United States dollars**.
- The price of an order is the wholesale price shown for that plan when the order is placed. Prices can change at any time; the catalogue is the current price.
- Live credit is **prepaid**. We issue an invoice; when it is paid we add that amount to your balance. Orders draw on the balance. We do not issue eSIMs on credit you have not paid for.
- You set the price you charge your own customers, and you are responsible for any taxes on your sales.

4. No refunds; cancellation

- **Credit is non-refundable.** Prepaid credit that you have not spent is not returned, including when an account is closed.
- **An issued eSIM is not refunded**, whether or not it is installed or used.
- You may **cancel** an eSIM at any time through the API or the console. A cancelled eSIM stops working and cannot be restored. Cancelling does not return credit.
- If we are unable to issue an eSIM you ordered, the charge for that order is reversed to your balance. That is the only circumstance in which a charge is reversed.

5. Our right to cancel or suspend

We may cancel or suspend any eSIM, any order, any API key or your account, at any time and without notice, including where we believe there is fraud, misuse, a breach of these terms, a risk to a network or to other customers, an unpaid invoice, or a legal or regulatory reason to do so. Where we reasonably can, we will tell you what we have done and why. A cancellation under this section does not entitle you to a refund or a credit.

6. What you may not do

- Resell access to the API itself, or let another company use your keys.
- Use eSIMs for unlawful purposes, for automated or machine-to-machine traffic not agreed with us, or in a way that breaches a network's fair-use rules.
- Attempt to discover, contact or deal directly with our suppliers, or to work out our costs. The identity of our suppliers and the prices we pay are confidential.
- Probe, overload or attempt to circumvent the API's limits or security.
- Describe yourself as Num, or imply that we endorse your product, without our written agreement.

7. Your customers

You are the seller to your customers. You are responsible for telling them what they are buying, for their support in the first instance, and for complying with the consumer, telecommunications and data-protection law that applies to you. Give us only the personal data we need to provide the service; an order requires none.

8. Service

We aim for the API to be available at all times and for eSIMs to be issued within seconds, but we do not guarantee either. Mobile networks are operated by third parties: coverage, speed and availability vary by

place and time and are outside our control. Plans and destinations may be withdrawn.

9. Liability

To the extent the law allows, the service is provided as is. We are not liable for indirect or consequential loss, lost profit, or loss arising from a network being unavailable. Our total liability to you in any twelve-month period is limited to the amount you paid us for orders in that period.

10. Changes and ending

We may change these terms. When a change is material we will tell the contact on your account, and continuing to use the API after that is acceptance of the new terms. You may stop using the API at any time. Sections 3, 4, 6, 7 and 9 continue to apply afterwards.

11. Law

These terms are governed by the laws of the State of Wyoming, United States, and its courts have jurisdiction, unless a law that applies to you cannot be excluded by agreement.

Questions: emma@workerslab.com.