



Travel eSIM API

Customer Integration Guide

A complete server-to-server reference for catalog discovery, wallet checks, eSIM ordering, installation delivery, transaction lookup, testing, and production launch.

VERSION	RELEASE	BASE URL
1.0	16 September 2026	https://www.simguyz.com/api/v1

Prepared for approved SimGuyz API customers
Keep API tokens and installation data confidential.

Contents

01

Quick Start
Base URL, environments, credentials, and first call

02

System Architecture
Secure integration boundary and transaction ownership

03

Authentication & Security
Bearer tokens, scopes, and safe storage

04

Endpoint Reference
Wallet, catalog, ordering, and lookup

05

Ordering Workflow
Validation, idempotency, wallet, and fulfillment

06

Installation Delivery
iOS, QR, LPA, and manual activation

07

Errors & Recovery
Status codes, retries, and failure behavior

08

Integration Examples
cURL, Python, and JavaScript

09

Testing & Go-Live
Acceptance tests and production checklist

10

Support Reference
Required support identifiers and glossary

Developer resources

PDF: www.simguyz.com/travel-esim-api | **OpenAPI:** www.simguyz.com/developer/travel-esim-guide/openapi
Postman: www.simguyz.com/developer/travel-esim-guide/postman | **Support:** support@simguyz.com

Key terms: eSIM = digital SIM profile; ICCID = SIM-profile identifier; IMSI = subscriber identity; LPA = installation string; SKU = catalog identifier; idempotency = duplicate-order protection; external ID = your order reference.

01 Quick Start

The SimGuyz Travel eSIM API is a JSON REST API designed for server-to-server use. Your backend reads the live catalog, submits an order, stores the returned transaction identifiers, and delivers the installation method to your customer.

Base URL <https://www.simguyz.com/api/v1>

Environments

Token prefix	Environment	Fulfillment	Wallet
sg_test_...	Test	Simulated result	Test balance
sg_live_...	Live	Real eSIM provisioning	Live balance

The token prefix selects the environment automatically. Never send a test token to a live workflow or a live token to a test workflow.

Fastest integration path

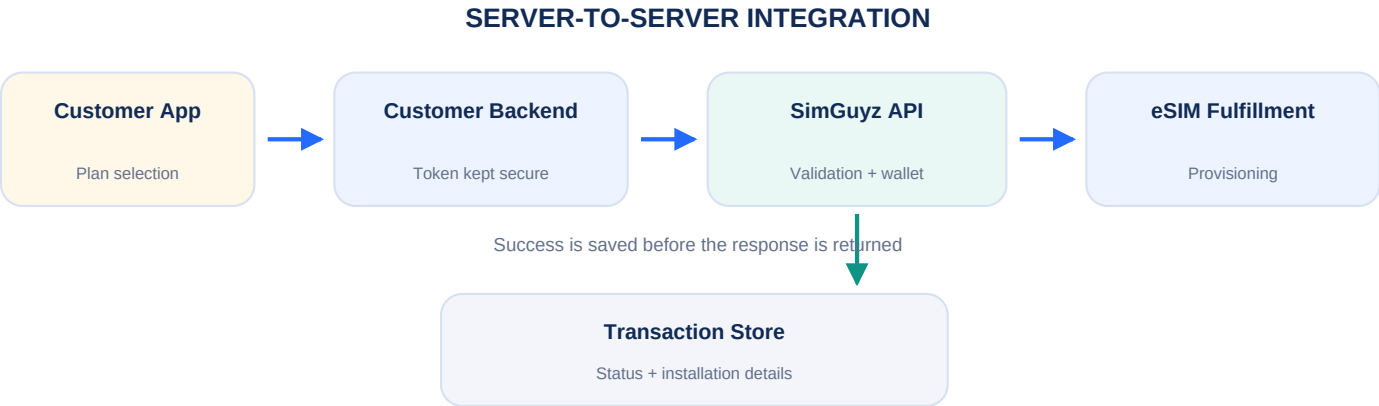
Step	Action	Endpoint
1	Confirm available balance	GET /wallet
2	Load current plans and final USD prices	GET /catalog/esim
3	Create one order with a unique idempotency key	POST /esim/orders
4	Store the result and display an installation method	Order response
5	Retrieve the saved result when required	GET /transactions/{id}

Required scopes

Operation	Accepted scopes
Read catalog	catalog:read, esim:read, esim:write, or admin:*
Create order	esim:write, activation:write, or admin:*
Read wallet	wallet:read, wallet:*, or admin:*
Read transaction	tx:read, tx:*, or admin:*

02 System Architecture

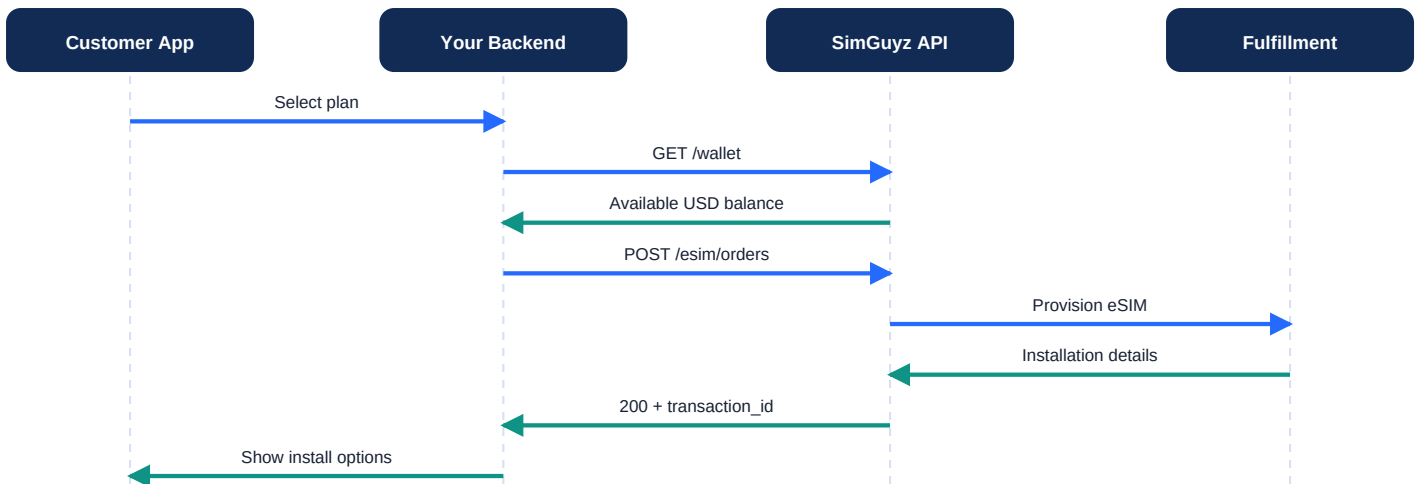
API tokens must remain in your trusted backend. Browser and mobile clients communicate with your backend; only your backend communicates with SimGuyz.



Integration responsibilities

Your system	SimGuyz
Secure token storage and customer authentication	Token authentication, scope, environment, and access checks
Catalog presentation and customer checkout	Current catalog and final USD order price
Unique external references and idempotency keys	Duplicate request protection and saved transaction result
Installation UX and customer support handoff	eSIM provisioning details and transaction lookup

Request sequence



03 Authentication & Security

Send the API token as a Bearer credential on every request. Write operations also require JSON content type and an idempotency key.

```

Authorization: Bearer sg_test_XXXXX
Accept: application/json
Content-Type: application/json
Idempotency-Key: esim-order-1001
  
```

Security requirements

Control	Requirement
Token storage	Store in a server-side secret manager or encrypted environment variable.
Client applications	Never embed tokens in browser JavaScript, mobile apps, URLs, or QR codes.
Transport	Use HTTPS only. Reject downgraded or certificate-invalid connections.
Logging	Redact tokens, activation codes, QR payloads, ICCIDs, and customer identifiers.
Access scope	Issue the narrowest scopes required by the integration.
Rotation	Rotate exposed or retired keys and update the backend secret atomically.

A token is treated like a payment credential because successful live orders debit the associated API wallet.

Request identifiers

Identifier	Created by	Use
request_id	SimGuyz	Trace one HTTP request in logs and support.
transaction_id	SimGuyz	Retrieve the saved order and fulfillment result.
external_id	Your system	Link SimGuyz activity to your own order.
Idempotency-Key	Your system	Prevent duplicate fulfillment during retries.

04 Endpoint Reference

Method	Endpoint	Purpose	Success
GET	/wallet	Current API wallet balance	200
GET	/catalog/esim	Available plans and final USD prices	200
POST	/esim/orders	Purchase and provision one eSIM	200
GET	/transactions/{transaction_id}	Retrieve a saved order result	200

GET /wallet

```
curl "https://www.simguyz.com/api/v1/wallet" \
-H "Authorization: Bearer sg_test_xxxxx" \
-H "Accept: application/json"
```

```
{
  "request_id": "req_4f8fb5f9",
  "status": "success",
  "environment": "test",
  "client_id": 42,
  "client_name": "Example Wireless",
  "billing_mode": "prepaid",
  "balance": "500.00",
  "currency": "USD"
}
```

Use this balance as a pre-check only. The order endpoint performs the authoritative balance check at submission time.

GET /catalog/esim

```
curl "https://www.simguyz.com/api/v1/catalog/esim" \
-H "Authorization: Bearer sg_test_xxxxx" \
-H "Accept: application/json"
```

```
{
  "request_id": "req_12bc49d1",
  "status": "success",
  "environment": "test",
  "currency": "USD",
  "count": 1,
  "plans": [
    {
      "sku": "esim:867924fdaf913e7fe11434972587449d",
      "plan_id": "867924fdaf913e7fe11434972587449d",
      "name": "Albania 0.49GB",
      "coverage": "Albania",
      "country_code": "AL",
      "data_gb": "0.49",
      "validity_days": "1",
      "price": "0.55",
      "sell_price": "0.55",
      "currency": "USD"
    }
  ]
}
```

Catalog fields

Field	Type	Description
sku	string	Customer-facing product identifier; recommended order input.
plan_id	string	Stable Travel eSIM plan identifier.
name	string	Display name for checkout.
coverage	string	Included country or region text.
country_code	string	ISO 3166-1 alpha-2 code when applicable.
data_gb	string	Included high-speed data in gigabytes.
validity_days	string	Plan validity in days.
price	string	Final order price in the stated currency.
sell_price	string	Compatibility alias of price.
currency	string	USD.

Refresh the catalog regularly and use the returned price at checkout. Do not maintain a separate local pricing formula.

POST /esim/orders

Field	Required	Type	Description
sku	Conditional	string	Catalog SKU; required when plan_id is omitted.
plan_id	Conditional	string	Catalog plan ID; required when sku is omitted.
external_id	Recommended	string	Your unique order or invoice reference.
user_id	Optional	string	Your customer/subscriber reference.

```
curl -X POST "https://www.simguyz.com/api/v1/esim/orders" \
-H "Authorization: Bearer sg_test_xxxxx" \
-H "Accept: application/json" \
-H "Content-Type: application/json" \
-H "Idempotency-Key: esim-order-1001" \
-d '{
  "sku": "esim:867924fdaf913e7fe11434972587449d",
  "external_id": "ORDER-1001",
  "user_id": "CUSTOMER-7788"
}'
```

Test success response

```
{
  "request_id": "req_810ce655",
  "status": "success",
  "transaction_id": "302",
  "product_type": "esim",
  "product_key": "esim:867924fdaf913e7fe11434972587449d",
  "provider": "simguyz",
  "amount_charged": "0.55",
  "currency": "USD",
  "environment": "test",
  "data": {
    "provider_status": "approved",
    "sku": "esim:867924fdaf913e7fe11434972587449d",
    "plan_id": "867924fdaf913e7fe11434972587449d",
    "iccid": "test_iccid_302",
    "activation_code": "ACT-302",
    "qr_code_url": "https://sandbox.example.com/qr/302",
    "balance_after": "499.45",
    "sandbox": true
  }
}
```

Live success response

```
{
  "request_id": "req_37dc9cf2",
  "status": "success",
  "transaction_id": "4102",
  "product_type": "esim",
  "product_key": "esim:867924fdaf913e7fe11434972587449d",
  "provider": "simguyz",
  "amount_charged": "0.55",
  "currency": "USD",
  "environment": "live",
  "data": {
    "provider_status": "approved",
    "sku": "esim:867924fdaf913e7fe11434972587449d",
    "plan_id": "867924fdaf913e7fe11434972587449d",
    "order_id": "ES-839201",
    "active_plan_id": "AP-50211",
    "iccid": "89010000000000000000",
    "imsi": "3100000000000000",
    "activation_code": "LPA:1$example.smdp.io$ABC123",
    "qr_code": "LPA:1$example.smdp.io$ABC123",
    "qr_code_url": "https://example.invalid/qr/ES-839201",
    "ios_installation_url": "https://example.invalid/install/ES-839201",
    "data_package_mb": 500,
    "data_used_mb": 0,
    "data_remaining_mb": 500,
    "created_at": "2026-09-16T10:20:30Z",
    "balance_after": "499.45"
  }
}
```

Fulfillment fields

Field	Description
transaction_id	SimGuyz identifier used for lookup and support.
amount_charged	Final USD amount debited after successful fulfillment.
data.iccid	Provisioned eSIM ICCID.
data.activation_code	Manual LPA/activation code when available.
data.qr_code	QR payload; may contain an LPA string.
data.qr_code_url	Hosted QR image URL when available.
data.ios_installation_url	Direct installation link for compatible iOS devices.
data.data_package_mb	Total data allowance in MB when available.
data.data_used_mb	Consumed data in MB when available.
data.data_remaining_mb	Remaining data in MB when available.
data.balance_after	API wallet balance after successful debit.

Optional fields may be omitted when they are not supplied for a specific product. Build the installation UI using field availability, not fixed assumptions.

GET /transactions/{transaction_id}

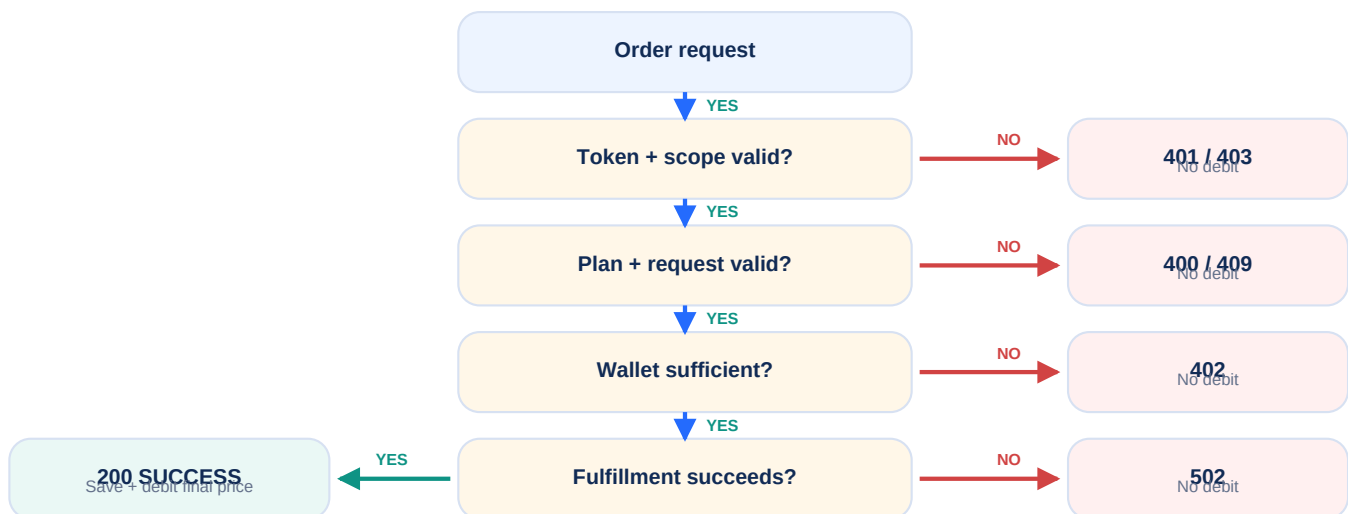
```
curl "https://www.simguyz.com/api/v1/transactions/4102" \
-H "Authorization: Bearer sg_live_xxxxx" \
-H "Accept: application/json"
```

```
{
  "request_id": "req_744a8b17",
  "status": "success",
  "transaction": {
    "id": 4102,
    "request_id": "req_37dc9cf2",
    "environment": "live",
    "product_type": "esim",
    "product_key": "esim:867924fdaf913e7fe11434972587449d",
    "provider": "simguyz",
    "provider_transaction_id": "ES-839201",
    "external_id": "ORDER-1001",
    "status": "success",
    "sell_price": "0.55",
    "currency": "USD",
    "error_code": null,
    "error_message": null,
    "response": {
      "sku": "esim:867924fdaf913e7fe11434972587449d",
      "plan_id": "867924fdaf913e7fe11434972587449d",
      "iccid": "89010000000000000000",
      "activation_code": "LPA:1$example.smdp.io$ABC123"
    }
  },
  "created_at": "2026-09-16T10:20:30Z",
  "updated_at": "2026-09-16T10:20:31Z"
}
```

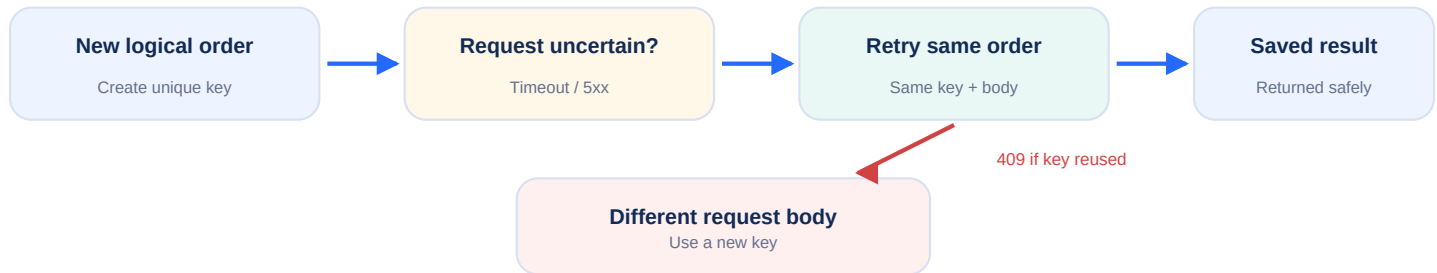
A transaction can only be retrieved by the API client that created it. Store transaction_id immediately after every successful order.

05 Ordering Workflow

The order endpoint validates access, request fields, idempotency, plan availability, and wallet balance before fulfillment. The wallet is debited only after successful fulfillment.



Idempotency and safe retry



Situation	Action
Same logical order, same body	Retry with the same Idempotency-Key.
Customer starts a different order	Generate a new Idempotency-Key.
Timeout or connection reset	Retry the identical request with the same key.
409 IDEMPOTENCY_CONFLICT	Do not retry with changed content; use the original body or a new key.

06 Installation Delivery

A successful response can contain one or more installation methods. Use the first method available in this priority order.

USE THE FIRST AVAILABLE INSTALLATION METHOD



Priority	Response field	Customer experience
1	ios_installation_url	Open on a compatible iPhone for direct installation.
2	qr_code_url	Display the hosted QR image on another screen.
3	qr_code	Generate a QR image locally when the value is an LPA payload.
4	activation_code	Provide manual installation details.

Customer instructions

Stage	Recommended message
Before installation	Confirm the device supports eSIM and is carrier-unlocked.
Install	Connect to Wi-Fi, then scan the QR or open the iOS installation link.
Travel	Enable the eSIM line and data roaming when ready to use the plan.
Privacy	Do not forward or publicly share the QR code or activation code.

Treat installation details as credentials. Do not place them in analytics events, public logs, support screenshots, or URLs controlled by third parties.

07 Errors & Recovery

```

{
  "request_id": "req_f8428040",
  "status": "failed",
  "transaction_id": "4103",
  "environment": "live",
  "error": {
    "code": "INSUFFICIENT_BALANCE",
    "message": "Wallet balance is too low."
  }
}
  
```

HTTP	Code	Meaning	Recommended action
400	VALIDATION_ERROR	Missing or invalid field	Correct the request body.

HTTP	Code	Meaning	Recommended action
400	IDEMPOTENCY_REQUIRE D	Header is missing	Send one unique key per order.
401	AUTH_MISSING	Bearer token missing	Add Authorization header.
401	AUTH_INVALID	Token invalid or wrong environment	Verify the configured token.
402	INSUFFICIENT_BALANCE	Wallet cannot cover final price	Fund wallet, then create a new order attempt.
403	SCOPE_DENIED	Required scope missing	Request the required API scope.
403	API_ACCESS_DISABLED	API access disabled	Contact SimGuyz support.
403	IP_NOT_ALLOWED	Source IP not permitted	Use an approved static egress IP.
404	NOT_FOUND	Transaction not owned by client	Verify transaction ID and token.
409	IDEMPOTENCY_CONFLIC T	Key reused with different body	Use original body or a new key.
502	UPSTREAM_ERROR	Fulfillment failed or timed out	Retry identical request with same key.

Retry policy

Response	Retry?	Rule
400 / 401 / 403 / 404	No	Correct authentication, permissions, identifiers, or validation first.
402	After funding	Fund the wallet, then create a new logical attempt with a new key.
409	Conditional	Use the original request body with the same key, otherwise use a new key.
429	Yes	Respect Retry-After when present and apply exponential backoff.
5xx / network failure	Yes	Retry identical JSON with the same key using exponential backoff.

Suggested backoff: 2 seconds, 5 seconds, 10 seconds, then 30 seconds. Cap attempts and alert your operations team when failures persist.

08 Integration Examples

Python

```
import requests

BASE_URL = "https://www.simguyz.com/api/v1"

def create_travel_esim(api_key, sku, external_id, customer_ref):
    payload = {"sku": sku, "external_id": external_id, "user_id": customer_ref}
    response = requests.post(
        f"{BASE_URL}/esim/orders",
        headers={
            "Authorization": f"Bearer {api_key}",
            "Accept": "application/json",
            "Content-Type": "application/json",
            "Idempotency-Key": external_id,
        },
        json=payload,
        timeout=45,
    )
    body = response.json()
    if not response.ok:
        error = body.get("error", {})
        raise RuntimeError(f"{error.get('code')}: {error.get('message')}")
    return body
```

JavaScript (server-side)

```
const BASE_URL = "https://www.simguyz.com/api/v1";

export async function createTravelEsim({ apiKey, sku, externalId, customerRef }) {
    const response = await fetch(`${BASE_URL}/esim/orders`, {
        method: "POST",
        headers: {
            Authorization: `Bearer ${apiKey}`,
            Accept: "application/json",
            "Content-Type": "application/json",
            "Idempotency-Key": externalId,
        },
        body: JSON.stringify({ sku, external_id: externalId, user_id: customerRef }),
    });

    const body = await response.json();
    if (!response.ok) {
        throw new Error(`${body.error?.code} || "API_ERROR": ${body.error?.message}`);
    }
    return body;
}
```

Reference implementation pattern

Layer	Implementation rule
Catalog cache	Cache briefly for performance, but refresh before checkout when practical.
Order record	Create a local pending order before calling SimGuыз.
Idempotency	Use the local order ID or a stable derived value.
Response storage	Store request_id, transaction_id, external_id, price, status, and installation fields.
Customer delivery	Expose installation data only to the authenticated purchaser.
Reconciliation	Use transaction lookup and your stored identifiers for support and audits.

09 Testing & Go-Live

Test acceptance matrix

Test	Expected result
Valid test token + GET /wallet	200 and test wallet balance.
Valid test token + GET /catalog/esim	200 with plans and USD prices.
Valid order	200, transaction ID, simulated ICCID, and installation data.
Repeat identical order with same key	Same saved response; no duplicate charge.
Reuse key with changed JSON	409 IDEMPOTENCY_CONFLICT.
Missing Bearer token	401 with structured error body.
Missing required scope	403 SCOPE_DENIED.
Insufficient test balance	402; wallet is not debited.
Transaction lookup	200 for own transaction; 404 for inaccessible transaction.
Installation UI	Uses the first available installation method.

Production launch checklist

Check	Complete when
Secrets	Live token is stored server-side and excluded from source control.
Environment separation	Test and live records, tokens, and wallets cannot be mixed.
Scopes	Live key has only the required catalog, wallet, order, and transaction scopes.
Wallet	Live wallet has adequate operating balance and low-balance monitoring.
Retries	Same logical order always reuses the same key and identical body.
Customer UX	QR, iOS link, LPA payload, and manual-code paths are tested.
Monitoring	HTTP status, request_id, transaction_id, latency, and error code are logged safely.
Support	Operations team can search by external_id and transaction_id.

10 Support Reference

When opening an order-support case, include the fields below. Never include the full API token, QR payload, or activation code.

Include	Example
Environment	test or live
request_id	req_37dc9cf2
transaction_id	4102
external_id	ORDER-1001
Timestamp	2026-09-16T10:20:30Z
HTTP status and error code	502 / UPSTREAM_ERROR