Collection API

Author: Josh Lead Backend Team
Last Updated: 2026-03-06
Version: v1.0

Base URL: `https://api.nextgate.co.tz/api/v1`

Short Description: The Collection API handles wallet top-up flows for NextGate users. It allows users to deposit money into their NextGate wallet via mobile money (USSD push) or card payment through Selcom. All subsequent platform payments (events, products) are made from the wallet balance.

Hints:

- Always provide a unique `idempotencyKey` per request to prevent duplicate charges
- For USSD channels the user receives a PIN prompt on their phone — poll `/status/{id}` to track completion
- For CARD channel the response contains a `paymentUrl` — redirect the user to complete payment
- Selcom webhooks automatically credit the wallet on confirmation — no manual step needed

---

Security & Client Guidelines

Authentication

All endpoints in this API are protected. The client must include a valid JWT Bearer token in every request header:

```
Authorization: Bearer <your_token>
```

The token is obtained from the NextGate authentication API after login. Tokens have an expiry time — when a request returns `401 UNAUTHORIZED`, the client must silently refresh the token using the refresh token endpoint and then retry the original request once. If the refresh also fails, redirect the user to the login screen. Never prompt the user manually to re-enter credentials on a token expiry.

Never store the JWT token in a place accessible to third-party scripts. On mobile, use secure device storage. On web, prefer memory or HttpOnly cookies over localStorage.

Idempotency Key

The `idempotencyKey` field protects against duplicate top-up requests caused by network retries. Without it, a user tapping the pay button while the network is slow could trigger two charges.

How to generate it correctly:

Generate the key exactly once when the user initiates the action — for example, when they tap the "Top Up" button — and store it in memory for the duration of that action. If the request fails due to a network error and your app retries, send the exact same key. The server will detect the duplicate and return the existing request instead of creating a new one.

Do not generate a new key on every retry. Do not use a random value that changes between attempts. A good key combines something unique to the user and something unique to the moment they initiated the action, for example combining the user ID with a timestamp captured at the moment the button was tapped.

The key must be unique per top-up attempt. Once a top-up completes or fails, do not reuse the same key for a future top-up — generate a fresh one for the next action.

Polling Strategy (USSD & Card)

After calling `/initiate`, the server returns immediately with status `AWAITING_CUSTOMER_ACTION`. The actual payment happens asynchronously — the user either enters their PIN on their phone (USSD) or pays on the card page. Your app must poll `/status/{id}` to know when the wallet has been credited.

Recommended polling behavior:

Poll every 5 seconds for the first 2 minutes. If still not completed after 2 minutes, slow down to every 15 seconds. Stop polling after 30 minutes total — this matches the server-side expiry window. If you reach 30 minutes without a `COMPLETED` status, show the user a message informing them the request has expired and they should try again.

Stop polling immediately when the status becomes `COMPLETED`, `FAILED`, or `EXPIRED`. Do not continue polling terminal states.

Card Payment Flow

When the channel is `CARD`, the response includes a `paymentUrl`. Open this URL in the device browser or a webview so the user can complete the card payment on Selcom's hosted page. Do not attempt to embed or replicate the card form in your app.

After the user returns from the payment page, resume polling `/status/{id}` — do not assume success just because the user returned. The wallet is only credited after Selcom sends a confirmation webhook to the server, which may take a few seconds after the card payment completes.

Network Error Handling

A network timeout or connection error on `/initiate` does not mean the request failed on the server. The server may have received and processed the request before the connection dropped. Always retry with the same `idempotencyKey` — the server will return the existing request if it was already created, preventing a duplicate.

On a timeout for `/status`, simply retry the poll on the next interval. Status polling is a read-only operation and is always safe to retry.

HTTP Status	What it means	Client action
`400`	Invalid request or business rule violation	Show `message` to user, do not retry automatically
`401`	Token expired or invalid	Refresh token silently, retry once
`403`	Forbidden — permission issue	Show error, do not retry
`422`	Validation error	Show field errors to user, fix and resubmit
`500`	Server error	Show generic error, allow user to retry manually
Network timeout	No response received	Retry with same idempotency key

---

Standard Response Format

Success Response

```
{
  "success": true,
  "httpStatus": "OK",
  "message": "Operation completed successfully",
  "action_time": "2026-03-06T10:30:45",
  "data": {}
}
```

Error Response

```
{
  "success": false,
  "httpStatus": "BAD_REQUEST",
  "message": "Error description",
  "action_time": "2026-03-06T10:30:45",
  "data": "Error description"
}
```

Standard Response Fields

Field	Type	Description
`success`	boolean	`true` for success, `false` for errors
`httpStatus`	string	HTTP status name
`message`	string	Human-readable result description
`action_time`	string	ISO 8601 timestamp
`data`	object/string	Response payload or error details

Flow Diagram

```
User
 |
 |--- POST /collection/initiate --------> CollectionController
 |       (channel, amount, msisdn)                |
 |                                      Validate request
 |                                      Check idempotency
 |                                      Save CollectionRequest (PENDING)
 |                                                |
 |                                .----------------.
 |                                |                |
 |                            USSD             CARD
 |                                |                |
 |                        Push USSD to      Create Selcom
 |                        Selcom API        Checkout Order
 |                                |                |
```

```
|                        Status: AWAITING  Status: AWAITING
|                        CUSTOMER_ACTION   CUSTOMER_ACTION
|                                |                |
|<-- Response (requestId) ------'----------------'
|
|   [USSD: User enters PIN on phone]
|   [CARD: User pays on paymentUrl ]
|
|                   Selcom Webhook ---------> /api/selcom/webhook
|                   (payment confirmed)               |
|                                          creditWallet()
|                                          Status: COMPLETED
|
|--- GET /collection/status/{id} -------> Status: COMPLETED
|<-- { status, transactionRef, ... }
```

Endpoints

---

1. Initiate Collection

Purpose: Initiates a wallet top-up request via mobile money (USSD) or card payment.

Endpoint: `POST` `/collection/initiate`

Access Level: Protected — Requires Bearer Token

Authentication: `Authorization: Bearer <token>`

Request Headers:

Header	Type	Required	Description
Authorization	string	Yes	Bearer JWT token
Content-Type	string	Yes	`application/json`

Request JSON Sample (USSD):

```
{
  "channel": "MPESA",
  "amount": 50000,
  "msisdn": "255712345678",
  "idempotencyKey": "usr-123-topup-1741234567"
}
```

Request JSON Sample (CARD):

```
{
  "channel": "CARD",
  "amount": 50000,
  "idempotencyKey": "usr-123-topup-1741234568"
}
```

Request Body Parameters:

Parameter	Type	Required	Description	Validation
`channel`	string	Yes	Payment channel	enum: `MPESA`, `AIRTEL`, `TIGO`, `HALOPESA`, `SELCOM_PESA`, `CARD`
`amount`	number	Yes	Amount to top up in TZS	Min: 1000
`msisdn`	string	Conditional	Mobile phone number	Required for all channels except CARD. Format: `255XXXXXXXXX` (12 digits)
`idempotencyKey`	string	Yes	Unique key to prevent duplicate requests	Max 200 chars, unique per request

Success Response JSON Sample (USSD):

```
{
  "success": true,
  "httpStatus": "OK",
  "message": "Collection initiated successfully",
  "action_time": "2026-03-06T10:30:45",
  "data": {
    "collectionRequestId": "a1b2c3d4-e5f6-7890-abcd-ef1234567890",
    "channel": "MPESA",
```

```
    "amount": 50000,
    "currency": "TZS",
    "status": "AWAITING_CUSTOMER_ACTION",
    "msisdnDisplay": "2557****678",
    "paymentUrl": null,
    "message": "Please enter your PIN on your phone to complete payment."
  }
}
```


Success Response JSON Sample (CARD):


```
{
  "success": true,
  "httpStatus": "OK",
  "message": "Collection initiated successfully",
  "action_time": "2026-03-06T10:30:45",
  "data": {
    "collectionRequestId": "a1b2c3d4-e5f6-7890-abcd-ef1234567890",
    "channel": "CARD",
    "amount": 50000,
    "currency": "TZS",
    "status": "AWAITING_CUSTOMER_ACTION",
    "msisdnDisplay": null,
    "paymentUrl": "https://checkout.selcom.net/pay/abc123",
    "message": "Redirect user to payment URL."
  }
}
```


Success Response Fields:

Field	Description
`collectionRequestId`	UUID — use this to poll `/status/{id}`
`channel`	Channel used for this collection
`amount`	Amount in TZS
`currency`	Always `TZS`
`status`	Current status — see status table below
`msisdnDisplay`	Masked phone number e.g. `2557****678`
`paymentUrl`	Card payment URL — redirect user here (null for USSD)
`message`	Human-readable instruction for the user

Collection Status Values:

Status	Description
`PENDING`	Request created, not yet sent to Selcom
`AWAITING_CUSTOMER_ACTION`	Sent to Selcom, waiting for user PIN or card payment
`COMPLETED`	Payment confirmed, wallet credited
`FAILED`	Payment failed or rejected
`EXPIRED`	Request expired before user acted (30 min window)

Error Responses:

Missing phone for USSD (400):


```
{
  "success": false,
  "httpStatus": "BAD_REQUEST",
  "message": "Phone number is required for MPESA payments.",
  "action_time": "2026-03-06T10:30:45",
  "data": "Phone number is required for MPESA payments."
}
```


Invalid phone format (400):


```
{
  "success": false,
  "httpStatus": "BAD_REQUEST",
  "message": "Invalid phone number format.",
  "action_time": "2026-03-06T10:30:45",
  "data": "Invalid phone number format."
}
```


Selcom rejection (400):


```
{
  "success": false,
  "httpStatus": "BAD_REQUEST",
  "message": "Payment initiation failed: Subscriber not found",
  "action_time": "2026-03-06T10:30:45",
  "data": "Payment initiation failed: Subscriber not found"
}
```

2. Get Collection Status

Purpose: Returns the current status of a collection request. Poll this after initiating to know when the wallet has been credited.

Endpoint: `GET` `/collection/status/{collectionRequestId}`

Access Level: □□ Protected — Requires Bearer Token

Authentication: `Authorization: Bearer <token>`

Path Parameters:

Parameter	Type	Required	Description
`collectionRequestId`	UUID	Yes	ID returned from `/initiate`

Success Response JSON Sample:

```
{
  "success": true,
  "httpStatus": "OK",
  "message": "Collection status retrieved",
  "action_time": "2026-03-06T10:30:45",
  "data": {
    "collectionRequestId": "a1b2c3d4-e5f6-7890-abcd-ef1234567890",
    "channel": "MPESA",
    "amount": 50000,
    "currency": "TZS",
    "status": "COMPLETED",
    "msisdnDisplay": "2557****678",
    "failureReason": null,
    "transactionRef": "NXT-TXN-20260306-ABCD1234",
    "createdAt": "2026-03-06T10:30:00",
    "completedAt": "2026-03-06T10:31:45"
  }
}
```

Success Response Fields:

Field	Description

`collectionRequestId`	UUID of this collection request
`channel`	Channel used
`amount`	Amount in TZS
`currency`	Always `TZS`
`status`	Current status — see status table above
`msisdnDisplay`	Masked phone number
`failureReason`	Populated if status is `FAILED`
`transactionRef`	Wallet transaction reference — available when `COMPLETED`
`createdAt`	When the request was created
`completedAt`	When the wallet was credited — null if not yet completed

Error Responses:

Not found or not owned by user (400):

```
{
  "success": false,
  "httpStatus": "BAD_REQUEST",
  "message": "Collection request not found",
  "action_time": "2026-03-06T10:30:45",
  "data": "Collection request not found"
}
```

Quick Reference

Method	Endpoint	Description
POST	`/collection/initiate`	Start a top-up via USSD or card
GET	`/collection/status/{id}`	Check top-up status

Revision #8
Created 2 October 2025 05:29:01 by Admin Qbit
Updated 6 March 2026 11:27:15 by Admin Qbit