

Location Service API

Author: Josh S. Sakweli, Backend Lead Team **Last Updated:** 2026-10-24 **Version:** v1.0

Base URL: `https://api.nexgate.co/api/v1`

Short Description: The location service turns map pins into Tanzanian places (ward, district, region), keeps every shop's location and every user's saved addresses, and hands checkout and shipping a frozen copy of a place. Everything in shipping — who can deliver, at what price, which campaign applies — starts from what this service knows about a place.

Hints:

- **Pins only.** Nobody types a street, city or region. The app sends `lat` / `lng`; the backend works out the ward. Show our `areaLabel` once a pin is confirmed (Google's text is only for the live map).
- **Places are frozen on orders.** Editing or deleting a saved address or moving a shop never changes an order already placed — the order keeps its own copy (a *place snapshot*, §1.5).
- All endpoints need a logged-in user, except reading a shop's location.
- Coded refusals come back as HTTP **409** with `data.code` (and sometimes `data.details`) — switch on the code, not the message.

Standard Response Format

Success

```
{
  "success": true,
  "httpStatus": "OK",
  "message": "Location resolved",
  "action_time": "2026-10-24T10:30:45",
  "data": { }
}
```

Error (coded refusal)

```
{
  "success": false,
  "httpStatus": "CONFLICT",
  "message": "You can save up to 10 addresses",
  "action_time": "2026-10-24T10:30:45",
  "data": { "code": "ADDRESS_LIMIT_REACHED", "reason": "You can save up to 10 addresses" }
}
```

Some refusals add `data.details` (e.g. `NEAR_DUPLICATE` returns the address it collides with).

Field	Type	Description
<code>success</code>	boolean	<code>true</code> for success, <code>false</code> for errors
<code>httpStatus</code>	string	HTTP status name
<code>message</code>	string	Human-readable result, safe to show
<code>action_time</code>	string	When the response was made
<code>data</code>	object/string	Payload, or error details

HTTP Method Badge Standards

- **GET** - **GET** - read only
- **POST** - **POST** - create / act
- **PUT** - **PUT** - replace
- **DELETE** - **DELETE** - remove

Part 1 — How the location service works

1.1 The map of Tanzania inside the backend

The backend holds the official **NBS 2022 ward boundaries**: about 4,300 wards, each inside a **district**, each district inside a **region**.

Region (e.g. 07 Dar es Salaam)

└ District (07-01 Kinondoni)

repeat across regions

└ Ward (1131 Kijitonyama)

← a district is regionCode + districtCode: district codes

← the smallest area; every pin lands in exactly one

The ward data is a file loaded by the backend at start-up (only when the file changes). If it is not loaded yet, calls that need it answer `WARD_DATA_UNAVAILABLE`.

1.2 A pin becomes a place — "resolve"

App map stops moving → GET /locations/resolve?lat&lng → covered?

└ yes: ward + district + region +

areaLabel

└ no: covered:false (sea, lake,

outside Tanzania)

- **CONTAINED** — the pin is inside a ward.
- **NEAREST** — the pin is just outside every ward (a beach, a river bank, a border line) but within **500 m** of one: that ward is used and `distanceM` says how far.
- Further than that → `covered: false`. The app disables *Confirm* and says "We don't deliver here yet".

Resolve stores nothing; it is safe to call on every map stop (debounce ~400 ms).

1.3 Areas — the picker for coverage and campaigns

Shippers tick **where they work** and campaigns choose **where they apply** with an *area*: a whole region, a district, or a single ward. The area endpoints give the lists (regions → districts → wards) and a search box (`/areas/search?q=sin` → "Sinza, Ubungo, Dar es Salaam"). An area is always one of:

level	identified by
REGION	regionCode
DISTRICT	regionCode + districtCode
WARD	wardId

1.4 Two kinds of places people own

	Shop location	Saved address
Who	the shop (owner sets it)	any user (buyer, seller, anyone)
How many	exactly one per shop	up to 10 per user, plus up to 5 recent pins
Who can see it	public — buyers need to find the shop	private — only the owner
Used for	pickup point, where riders collect, product page "Directions"	where to deliver
Extra	landmark, building, official address, the shop's phone	label (Home / Work / named), directions

Recent pins: when an order is delivered to a new pin that the buyer did not save, the pin becomes a *recent* (up to 5, kept 60 days; a pin within 50 m of an existing recent just refreshes it). A recent can later be saved as an address.

1.5 Place snapshots — frozen copies on orders

Every order, quotation and shipping leg keeps its own copy of the places it uses:

```
{
  "source": "SAVED_ADDRESS",           // SHOP_LOCATION | SAVED_ADDRESS | RECENT | NEW_PIN
| STATION
  "sourceId": "9b1d...",               // the address / recent / shop / station it came
from
  "label": "Home",
  "lat": -6.7735, "lng": 39.2187,
  "ward": { "wardId": 1131, "wardName": "Kijitonyama", "districtCode": "01", "districtName":
"Kinondoni",
            "regionCode": "07", "regionName": "Dar es Salaam", "nbsCode": "070113062",
"councilName": "..."},
  "areaLabel": "Kijitonyama, Kinondoni, Dar es Salaam",
  "directions": "Black gate, blue house",
  "building": null,                   // shops only
  "phone": "+255744742944",          // shops and stations only
  "capturedAt": "2026-10-24T10:30:45"
```

```
}
```

Because of this, editing or deleting an address, or moving a shop, **never changes an order already placed**.

1.6 Choosing where to deliver — the drop-off request

Checkout, the cart and shipping all accept the same small object to say *where*:

```
{ "addressId": "..." } // a saved address
{ "recentId": "..." } // a recent pin
{ "lat": -6.7735, "lng": 39.2187, // a new pin
  "directions": "Opposite the petrol station",
  "saveAs": "OTHER", "customLabel": "Office" } // optional: also save it to the address book
```

Give **exactly one** of `addressId`, `recentId` or `lat` + `lng` → otherwise `DROPOFF_INVALID`; none at all where one is needed → `DROPOFF_REQUIRED`. Who receives the parcel and the note for the rider belong to the **order**, not to the place.

1.7 Event venues

An in-person or hybrid event has one venue: name, optional typed address, and a **pin**. The pin is resolved like a shop's (`OUTSIDE_COVERAGE` when not covered) and the event's `venue` shows `ward`, `areaLabel` and `matchType`. A pin is required to publish (or reveal the location of) an in-person / hybrid event. The venue is part of the event (no separate endpoint): set it with `PATCH /e-events/drafts/{draftId}/location` or `PATCH /e-events/{eventId}/published/reveal-location`.

1.8 How shipping uses all this

- **Who can deliver:** a shipper covers a trip only when both the shop's ward and the buyer's ward are inside its ticked areas.
 - **Price:** road distance = straight line between the two pins × 1.3.
 - **Campaigns:** "free delivery in Dar" means the buyer's ward is in region 07.
 - **Reports** are by region / district / ward, never per person.
-

Part 2 — Endpoints

Pins and areas

1. Resolve a pin

Purpose: Turn a map pin into ward / district / region, or say it is not covered.

Endpoint: **GET** `{base_url}/locations/resolve`

Access Level: ☐ Protected (any logged-in user)

Authentication: Bearer token

Query Parameters:

Parameter	Type	Required	Description	Validation	Default
lat	number	Yes	Latitude of the pin	-90 ... 90, not 0,0	—
lng	number	Yes	Longitude of the pin	-180 ... 180	—

Success Response JSON Sample (covered):

```
{
  "success": true,
  "httpStatus": "OK",
  "message": "Location resolved",
  "action_time": "2026-10-24T10:30:45",
  "data": {
    "covered": true,
    "latitude": -6.819, "longitude": 39.2768,
    "matchType": "CONTAINED",
    "distanceM": 0,
    "ward": {
      "wardId": 1146, "nbsCode": "070214112", "wardName": "Kariakoo",
      "districtCode": "02", "districtName": "Ilala",
      "councilName": "Halmashauri ya Jiji la Dar es Salaam",
```

```
    "regionCode": "07", "regionName": "Dar es Salaam"
  },
  "areaLabel": "Kariakoo, Ilala, Dar es Salaam"
}
}
```

Not covered (sea, lake, outside Tanzania): `data = { "covered": false, "latitude": -6.8, "longitude": 39.4, "reason": "OUTSIDE_COVERAGE" }`.

Success Response Fields:

Field	Description
<code>covered</code>	<code>false</code> = no ward within 500 m: disable <i>Confirm</i>
<code>matchType</code>	<code>CONTAINED</code> inside a ward, <code>NEAREST</code> just outside one (≤ 500 m)
<code>distanceM</code>	metres to the ward when <code>NEAREST</code>
<code>ward</code>	the ward with its district and region
<code>areaLabel</code>	the name to show: "Ward, District, Region"
<code>reason</code>	only when not covered

Errors: `INVALID_COORDINATES`, `WARD_DATA_UNAVAILABLE`.

2. List regions

Purpose: First level of the area picker.

Endpoint: `GET` `{base_url}/locations/regions`

Access Level: ☐ Protected

Success Response JSON Sample:

```
{
  "success": true, "httpStatus": "OK", "message": "Regions", "action_time": "2026-10-24T10:30:45",
  "data": [
    { "regionCode": "07", "regionName": "Dar es Salaam", "districtCount": 5, "wardCount": 102 },
    { "regionCode": "12", "regionName": "Mbeya", "districtCount": 7, "wardCount": 199 }
```

```
1  
}
```

3. List a region's districts

Endpoint: **GET** `{base_url}/locations/regions/{regionCode}/districts`

Access Level: ☐ Protected

Path Parameters:

Parameter	Type	Required	Description
regionCode	string	Yes	e.g. <code>07</code>

Success Response `data`: `[{ "regionCode": "07", "regionName": "Dar es Salaam", "districtCode": "01", "districtName": "Kinondoni", "wardCount": 20 }]`

4. List a district's wards

Endpoint: **GET** `{base_url}/locations/regions/{regionCode}/districts/{districtCode}/wards`

Access Level: ☐ Protected

Success Response `data`: a list of ward objects, the same shape as `ward` in [Resolve](#).

5. Search areas

Purpose: Type-ahead for the area picker ("sin" → Sinza).

Endpoint: **GET** `{base_url}/locations/areas/search?q=sin`

Access Level: ☐ Protected

Query Parameters:

Parameter	Type	Required	Description	Validation
-----------	------	----------	-------------	------------

q	string	Yes	Part of a region, district or ward name	at least 2 letters
---	--------	-----	---	--------------------

Success Response `data` (up to 20, regions first, prefix matches first):

```
[
  { "level": "WARD", "regionCode": "07", "districtCode": "04", "wardId": 1212, "name":
    "Sinza", "label": "Sinza, Ubungo, Dar es Salaam" }
]
```

The returned item is exactly what coverage and campaigns accept as an area (`level` + codes).

Errors: `SEARCH_TOO_SHORT`.

Shop location

6. Set or replace the shop location

Purpose: The shop owner pins the shop. Used for pickups, riders and the product page.

Endpoint: `PUT` `{base_url}/e-commerce/shops/{shopId}/location`

Access Level: `[[` Protected (shop owner, or platform admin)

Request JSON Sample:

```
{
  "lat": -6.8190, "lng": 39.2768,
  "landmark": "Kariakoo market, opposite the clock tower",
  "building": "Shop 12",
  "officialAddress": null,
  "deviceLat": -6.7700, "deviceLng": 39.2400
}
```

Request Body Parameters:

Parameter	Type	Required	Description	Validation
lat / lng	number	Yes	The pin	must be inside a ward (or $\leq$ 500 m)

Parameter	Type	Required	Description	Validation
landmark	string	Yes	How to find the shop	max 300
building	string	No	Building / shop number	max 100
officialAddress	string	No	Anwani za Makazi	max 100
deviceLat / deviceLng	number	No	The phone's own position, only to warn when the pin is far	—

Success Response (message "Shop location saved"): the shop location (as in [Get](#)), plus `warnings` when the pin is more than 1 km from the phone: `"warnings": [{ "code": "FAR_FROM_DEVICE", "distanceM": 6797 }]` → ask "Your pin is 6.8 km from where you are. Is this your shop?" It is saved either way.

Rules: one location per shop; editing = the same `PUT` with the whole location; the ward is always worked out from the pin. Orders already placed keep their pickup place. Buyers are not notified when a shop moves.

Errors: `OUTSIDE_COVERAGE`, `INVALID_COORDINATES`, 403 "Only the shop owner can set the shop location".

7. Get a shop location

Endpoint: `GET` `{base_url}/e-commerce/shops/{shopId}/location`

Access Level: `Public` (no login)

Success Response JSON Sample:

```
{
  "success": true, "httpStatus": "OK", "message": "Shop location retrieved", "action_time":
"2026-10-24T10:30:45",
  "data": {
    "shopId": "aaaa0000-0000-4000-8000-000000000001",
    "lat": -6.819, "lng": 39.2768,
    "ward": { "wardId": 1146, "wardName": "Kariakoo", "districtName": "Ilala", "regionName":
"Dar es Salaam" },
    "matchType": "CONTAINED",
    "areaLabel": "Kariakoo, Ilala, Dar es Salaam",
    "landmark": "Kariakoo market, opposite the clock tower",
  }
}
```

```
"building": "Shop 12",
"phone": "+255744742944",
"officialAddress": null,
"updatedAt": "2026-10-12T11:49:59"
}
}
```

`phone` is the shop's own phone (one phone per shop). **App:** "Directions" opens maps with `lat, lng`; "Call" uses `phone`. 404 when the shop has no location yet. A shop without a location cannot sell physical products: checkout answers `SHOP_LOCATION_REQUIRED`.

Saved addresses and recent pins

8. My addresses

Endpoint: `GET` `{base_url}/locations/addresses`

Access Level: ☐ Protected (own addresses only)

Success Response `data`: default first, then most recently used:

```
[
  {
    "id": "9b1d...", "labelType": "HOME", "customLabel": null,
    "lat": -6.7745, "lng": 39.2419,
    "ward": { "wardName": "Kijitonyama", "districtName": "Kinondoni", "regionName": "Dar es Salaam" },
    "matchType": "CONTAINED", "areaLabel": "Kijitonyama, Kinondoni, Dar es Salaam",
    "directions": "Black gate, blue house",
    "isDefault": true, "lastUsedAt": "2026-10-20T09:00:00", "createdAt": "...", "updatedAt": "..."
  }
]
```

9. Save an address

Endpoint: `POST` `{base_url}/locations/addresses`

Access Level: ☐ Protected

Request JSON Sample:

```
{
  "lat": -6.7745, "lng": 39.2419,
  "labelType": "OTHER",
  "customLabel": "Mama's place",
  "directions": "Black gate, blue house",
  "makeDefault": false,
  "allowNearDuplicate": false
}
```

Request Body Parameters:

Parameter	Type	Required	Description	Validation
lat / lng	number	Yes	The pin	inside a ward
labelType	enum	Yes	HOME, WORK, OTHER	HOME / WORK once each
customLabel	string	for OTHER	The name for an OTHER address	max 40
directions	string	No	How to find the place	max 300
makeDefault	boolean	No	Make it the default	—
allowNearDuplicate	boolean	No	Save even if within 50 m of another address	—

Success Response: the saved address (shape as in [My addresses](#)).

Rules:

- Up to **10** addresses. The **first** one saved becomes the default; deleting the default leaves none (the app pre-selects the most recently used).
- Within **50 m** of another saved address → NEAR_DUPLICATE with that address in `data.details`. Ask "This looks like ☐ Home. Use it?"; to save anyway send again with `allowNearDuplicate: true`.

Errors: ADDRESS_LIMIT_REACHED, LABEL_ALREADY_USED, CUSTOM_LABEL_REQUIRED, NEAR_DUPLICATE, OUTSIDE_COVERAGE.

10. Edit an address

Endpoint: **PUT** `{base_url}/locations/addresses/{addressId}`

Access Level: ☐ Protected (own address)

Same body and rules as [Save](#) — send the whole address. Orders already placed are not affected.

11. Delete an address

Endpoint: **DELETE** `{base_url}/locations/addresses/{addressId}`

Access Level: ☐ Protected (own address)

12. Make an address the default

Endpoint: **POST** `{base_url}/locations/addresses/{addressId}/default`

Access Level: ☐ Protected (own address)

Success Response: the address, now `isDefault: true` (the old default is cleared).

13. My recent pins

Endpoint: **GET** `{base_url}/locations/recent`

Access Level: ☐ Protected

Success Response `data: [{ "id": "...", "lat": -6.7735, "lng": 39.2187, "ward": {...}, "matchType": "CONTAINED", "areaLabel": "...", "lastUsedAt": "..."}]` — up to 5, newest first. Added by checkout when an order is delivered to a new, unsaved pin.

14. Remove a recent pin / clear all

Endpoints:

- **DELETE** {base_url}/locations/recent/{recentId}
- **DELETE** {base_url}/locations/recent

Access Level: ☐ Protected

Part 3 — Codes and values

Error codes (HTTP 409, data.code)

Code	When
INVALID_COORDINATES	lat/lng out of range, or 0,0
OUTSIDE_COVERAGE	saving a shop location or address whose pin is not in Tanzania's wards (resolve answers covered: false instead)
WARD_DATA_UNAVAILABLE	the ward data is not loaded on this server yet
SEARCH_TOO_SHORT	area search with fewer than 2 letters
ADDRESS_LIMIT_REACHED	an 11th saved address
LABEL_ALREADY_USED	a second HOME or WORK
CUSTOM_LABEL_REQUIRED	OTHER without a name
NEAR_DUPLICATE	within 50 m of a saved address (data.details = that address)
DROPOFF_REQUIRED	a drop-off is needed and none was given
DROPOFF_INVALID	not exactly one of address / recent / pin, or the address / recent is not yours
SHOP_LOCATION_REQUIRED	checkout of a physical product from a shop without a location

Values

Value	Meaning
MatchType CONTAINED / NEAREST	pin inside a ward / within 500 m of one
AddressLabel HOME / WORK / OTHER	HOME and WORK once each; OTHER needs customLabel
PlaceSource	where a snapshot came from: SHOP_LOCATION, SAVED_ADDRESS, RECENT, NEW_PIN, STATION

Value	Meaning
AreaLevel REGION / DISTRICT / WARD	the three kinds of area

Limits

Limit	Value
Nearest-ward distance	500 m
Saved addresses per user	10
Recent pins per user / kept	5 / 60 days
Same place (duplicate / recent refresh)	50 m
Shop pin far from the phone (warning)	1 km
Area search results	20

Revision #1
Created 24 October 2026 19:26:27 by Admin Qbit
Updated 24 October 2026 19:27:11 by Admin Qbit