StroWallet API – Comprehensive Documentation

Last updated: August 16, 2025

Base URLs

External (StroWallet): https://strowallet.com/api/bitvcard
Your Backend Proxy (recommended): https://<your-domain>/api/strowallet
Webhook Receiver: https://workspace.wogisec814.replit.app/api/webhook/strowallet

Authentication

- Every request to StroWallet requires your public_key.
- Do NOT expose your public_key to the frontend. Call StroWallet only from your backend.
- For testing, pass mode: "sandbox" (where supported).

Conventions

- Content-Type: application/json unless otherwise stated.
- All dates are strings. dateOfBirth uses MM/DD/YYYY.
- Phone number must be international format without '+' (e.g., 2348012345678).

Errors

- 200 OK – Successful request.
- 400 Bad Request – Validation or request error.
- Your backend should normalize errors to: { ok: false, error: "message" }.

CUSTOMERS

---

1) Create Customer

POST /create-user/

Description: Registers a new customer in StroWallet (KYC).

Body (JSON):

```
{
"public_key": "YOUR_PUBLIC_KEY",
"houseNumber": "12B",
"firstName": "John",
"lastName": "Doe",
"idNumber": "AB123456",
"customerEmail": "john@example.com",
"phoneNumber": "2348012345678",
"dateOfBirth": "01/15/1990",
"idImage": "https://example.com/id.jpg",
"userPhoto": "https://example.com/photo.jpg",
"line1": "123 Main Street",
"state": "Lagos",
"zipCode": "100001",
"city": "Ikeja",
"country": "NG",
```

"idType": "PASSPORT" // BVN, NIN, PASSPORT
}

Response:
• 200 – Customer created
• 400 – Missing/invalid data

Example (curl):
curl -X POST https://strowallet.com/api/bitvcard/create-user/ -H "Content-Type: application/json" -d '{...}'

2) Get Customer

GET /getcardholder/

Query Parameters:
• public_key (required)
• customerId (optional)
• customerEmail (optional)

Response:
• 200 – Customer data
• 400 – Error

3) Update Customer

PUT /updateCardCustomer/

Body (JSON):
{
"public_key": "YOUR_PUBLIC_KEY",
"customerId": "CUST_123",
"firstName": "John",
"lastName": "Doe",
"idImage": "https://example.com/id.jpg",
"userPhoto": "https://example.com/photo.jpg",
"phoneNumber": "2348012345678",
"country": "NG",
"city": "Ikeja",
"state": "Lagos",
"zipCode": "100001",
"line1": "123 Main Street",
"houseNumber": "12B"
}

Response:
• 200 – Updated
• 400 – Error

CARDS
---

4) Create Card

POST /create-card/

Body (JSON):

{
"name_on_card": "My Name",
"card_type": "visa",
"public_key": "YOUR_PUBLIC_KEY",
"amount": "5",
"customerEmail": "mydemo@gmail.com",
"mode": "sandbox", // optional; pass for sandbox testing
"developer_code": "OPTIONAL"
}

Response:
• 200 – Card created
• 400 – Error

5) Fund Card

POST /fund-card/

Body (JSON):
{
"card_id": "jn564bjn33jdgf",
"amount": "3",
"public_key": "YOUR_PUBLIC_KEY",
"mode": "sandbox"
}

Response:
• 200 – Funded
• 400 – Error

6) Get Card Details

POST /fetch-card-detail/

Body (JSON):
{
"card_id": "CARD_ID",
"public_key": "YOUR_PUBLIC_KEY",
"mode": "sandbox"
}

Response:
• 200 – Card details
• 400 – Error

7) Card Transactions (recent)

POST /card-transactions/

Body (JSON):
{
"card_id": "CARD_ID",
"public_key": "YOUR_PUBLIC_KEY",
"mode": "sandbox"
}

Response:

• 200 – Transactions
• 400 – Error

8) Freeze / Unfreeze Card

POST /action/status/

Body (JSON):
{
"action": "freeze", // or "unfreeze"
"card_id": "CARD_ID",
"public_key": "YOUR_PUBLIC_KEY"
}

Response:
• 200 – Success
• 400 – Error

9) Full Card History (paginated)

GET /apicard-transactions/

Query:
• card_id (required)
• page (required, default 1)
• take (required, default 50, max 50)
• public_key (required)

Response:
• 200 – Paginated history
• 400 – Error

WEBHOOKS

Endpoint (Your server):
POST /api/webhook/strowallet

Example Hosted URL (given):
https://workspace.wogisec814.replit.app/api/webhook/strowallet

Headers:
• x-strowallet-signature: <HMAC SHA256 signature> (if provided)

Body: Raw JSON event payload. Example fields (may vary):
{
"id": "evt_123",
"type": "card.created", // or card.funded, card.frozen, transaction.posted, etc.
"data": { ... },
"created": 1690000000
}

Recommendations:
• Verify signature with your STROWALLET_WEBHOOK_SECRET if available.
• Store every event with eventId and eventType; de-duplicate on retry.
• Update local Card and Transaction records accordingly.

Minimal Express Handler:

```
app.post("/api/webhook/strowallet", express.raw({ type: "*/*" }), (req, res) => {
const raw = req.body.toString("utf8");
const payload = JSON.parse(raw || "{}");
// verify signature if header present
// persist payload
res.status(200).json({ ok: true });
});
```

SAMPLE CODE SNIPPETS

Node.js (axios) – Create Customer:

```
import axios from "axios";
await axios.post("https://strowallet.com/api/bitvcard/create-user/", {
public_key: process.env.STROWALLET_PUBLIC_KEY,
houseNumber: "12B",
firstName: "John",
lastName: "Doe",
idNumber: "AB123456",
customerEmail: "john@example.com",
phoneNumber: "2348012345678",
dateOfBirth: "01/15/1990",
idImage: "https://example.com/id.jpg",
userPhoto: "https://example.com/photo.jpg",
line1: "123 Main Street",
state: "Lagos",
zipCode: "100001",
city: "Ikeja",
country: "NG",
idType: "PASSPORT"
});
```

Node.js (axios) – Create Card:

```
import axios from "axios";
await axios.post("https://strowallet.com/api/bitvcard/create-card/", {
name_on_card: "My Name",
card_type: "visa",
public_key: process.env.STROWALLET_PUBLIC_KEY,
amount: "5",
customerEmail: "mydemo@gmail.com",
mode: "sandbox"
});
```

Python (requests) – Fund Card:

```
import requests
requests.post("https://strowallet.com/api/bitvcard/fund-card/", json={
"card_id": "jn564bjn33jdgf",
"amount": "3",
"public_key": "YOUR_PUBLIC_KEY",
"mode": "sandbox"
}).json()
```

Python (requests) – Full Card History:

```
import requests
requests.get("https://strowallet.com/api/apicard-transactions/", params={
"card_id": "CARD_ID",
"page": "1",
"take": "50",
"public_key": "YOUR_PUBLIC_KEY"
}).json()
```

BEST PRACTICES

• Backend-only StroWallet calls; never from frontend.
• Validate all input (e.g., Zod) before calling StroWallet.
• Persist external responses locally (MongoDB) for audit and caching.
• Normalize errors and log details server-side.
• Secure secrets in environment variables (.env).

Changelog (from source dump)

• Create Customer – updated 11 months ago
• Get Customer – updated 10 months ago
• Update Customer – updated 6 months ago
• Create/Fund/Details/Transactions – about 1 year ago
• Freeze/Unfreeze – almost 2 years ago
• Full Card History – over 1 year ago