Rewritten from the V0.36 manual for wallet V0.51. Terminology updated: the chain renamed game/games to app/apps, so endpoints are /api/v1/apps and keys are app_type.

ZooBC Wallet — User Manual

Version V0.36 · July 2026

This manual documents every screen, tab, field and transaction type in the ZooBC Wallet. The wallet is a single HTML file that runs entirely in your browser: your keys never leave your device, all transactions are built and signed locally, and only the finished signed bytes are sent to the network.

Table of contents

---

1. Core concepts

1.1 The coin and its units

The native coin is **ZBC**. On-chain, every amount is stored in **atomic units**: 1 ZBC = 100,000,000 atomic (10^8). The wallet always shows human amounts and converts at the edges. Issued tokens have their own decimals (0–8) and are likewise stored in base units.

1.2 Accounts and address formats

One wallet can hold accounts on many key schemes. Every account is stored on-chain as a **typed account**: a 4-byte type prefix + the key material. Supported formats:

Format	Type	Key scheme	Address looks like
ZooBC	0	ed25519	`ZBC_XXXX_…` (base32 + checksum)
Ethereum / BNB	4	secp256k1 + keccak	`0x…` (same bytes, different label)
Bitcoin legacy (P2PKH)	5	secp256k1	`1…`
Bitcoin P2SH	6	— (receive only)	`3…`
Bitcoin SegWit (P2WPKH)	7	secp256k1	`bc1q…`
Bitcoin P2WSH	8	— (receive only)	`bc1q…` (long)
Bitcoin Taproot	9	BIP-340 schnorr	`bc1p…`
Dataset deposit	10	— (object rent account)	`ZBS_…`
Solana	11	ed25519	base58 pubkey
Polkadot	12	sr25519 (Schnorrkel)	SS58
Cardano	13	ed25519	`addr1…`
Ripple	14	secp256k1	`r…`
Tron	15	secp256k1 + keccak	`T…`
Tezos	16	ed25519	`tz1…`

Any format can **send and receive on the ZooBC chain** — a Bitcoin-format account signs ZooBC transactions with its Bitcoin key. The address is derived exactly as on the original chain, so a bridged deposit from that chain credits the mirrored account automatically (see Deposit).

Derivation paths (from a BIP-39 phrase): ZooBC `m/44'/883'/account'` (SLIP-0010 ed25519); ETH/BNB BIP-44 coin 60; BTC BIP-44/84/86 for legacy/SegWit/Taproot; Solana `m/44'/501'/account'`; Cardano coin 1815; Tezos coin 1729; Polkadot uses substrate mini-secret derivation.

1.3 Transactions, fees, escrow

Every action on the chain is a signed **transaction** with a type number, a fee in ZBC, an optional message, and an optional **escrow wrapper** (approver + deadline + commission + instruction). The default network fee is **0.05 ZBC** (base 0.025 + headroom for data-carrying transactions); the wallet pre-fills it and raises it automatically for large messages. The full type list is in section 12.

1.4 Gateways, nodes, relays

- A **node** validates the chain.
- A **gateway** is the web server the wallet talks to; it serves the wallet/explorer and fronts one or more nodes.
- A **relay** is the call/presence server a gateway runs; it carries call signaling and online status.
- An **archival node** additionally serves full history and the explorer API.

If the configured gateway dies, the wallet **fails over automatically**: current node → built-in fallbacks → your extra nodes (Settings → Node) → gateways learned from the on-chain registry.

1.5 Storage rent and survival objects

Data written on-chain (datasets, stored files, tokens) pays **storage rent**. Each object has a fundable **ZBS_ deposit address** derived from the transaction that created it; when the deposit runs out the object can be pruned. You can top up any object's longevity (see Files → Objects).

1.6 Security model

- Your recovery phrase / key **never leaves the page**; signing is local.

- On-device persistence is an **encrypted vault** locked by a **6-digit PIN** (5 wrong attempts wipe the vault — the phrase restores it).
- Optional **passkey** unlock (Face ID / Touch ID / Windows Hello).
- **Watch-only** accounts can see everything but can't sign; key-requiring pages are hidden.
- **Air-gap mode** signs offline and exports JSON/QR instead of broadcasting.

---

2. Getting started

2.1 Before you open it — check the file is genuine

This wallet is a single file you open in a browser and then type your recovery phrase into. A fake

copy looks identical and behaves normally, right up to the moment it sends your phrase somewhere

else. Nothing on screen can warn you, because by then the fake page is the thing doing the warning.

So check the file before you open it. It takes about ten seconds.

The easy way. Open `/verify` on the gateway you downloaded from, and drag the file onto the box.

The check runs inside your own browser — the file is never uploaded anywhere.

The way that needs nothing but a terminal. A hash is a fingerprint of a file: change one byte

and it changes completely, and nobody can work backwards to build a different file with the same

fingerprint.

```
sha256sum "ZooBC Wallet.html"
```

Compare what that prints, character for character, with the hash published for that file.

Then ask who told you the fingerprint. This is the part that matters, and the `/verify` page is

explicit about it:

what it says	what it means
on-chain	The hash was signed into a block by the release authority — one key, fixed in the chain's genesis, that is the only thing allowed to register a file hash. Every node holds that record. A website cannot change it, fake it, or remove it.
this gateway says so	The hash came from the website you are looking at. That catches a corrupted download or a careless fake. It does **not** help if that website is itself the attacker, because then the file and its fingerprint are both fake and agree perfectly.
REVOKED on-chain	The authority published this file and then **withdrew it** — usually because it turned out to be compromised or faulty. This is worse than an unrecognised file, not better: the chain has something specific to say about this one. Delete it and fetch a current file. A website still offering it is a reason for more suspicion, not less.

For anything involving money, an on-chain match is the one to insist on. A gateway vouching for its

own downloads is not proof.

How to check without trusting any website at all. Everything above still runs on a page someone

serves you. If you want a check that does not, ask a node directly — any node, including one you run

yourself. Two commands:

```
sha256sum "ZooBC Wallet.html"
curl -s http://NODE:8080/api/v1/release/list
```

If your file's hash appears in that list, it is a file the release authority published. If it does

not appear, no amount of reassurance from any website makes it safe to type a recovery phrase into.

You can also ask which key is allowed to publish at all:

```
curl -s http://NODE:8080/api/v1/release/authority
```

Every node answers the same thing, because it is written into the chain. Ask two nodes run by

different people and compare — that is the whole point of the design, and it costs you one extra

command.

If `/verify` warns that a change of release authority is pending, stop and find out why before

trusting anything you downloaded from that site. The authority is the one key behind every claim on

that page, and proposing to move it is precisely what someone who had just stolen it would do. It

may well be a planned rotation — but confirm that through a channel that does not run through the

website in question, because a compromised site will happily reassure you about its own handover.

> This matters more as ZooBC grows into other wallets and other languages. You should never have to take anyone's word for which file is genuine — the chain exists so that you do not have to.

2.2 First launch — risk notice

On the very first launch a **Software Interface Notice & User Risk Acknowledgment** is shown. You must accept it (or exit); the full Terms of Service stay available from the footer at any time.

2.3 Welcome screen

The onboarding card offers:

- **Recovery phrase or secret key** — paste a 12/15/18/21/24-word BIP-39 phrase or a 64-hex private key. The eye button reveals/masks it.
- **Wallet format** — ZooBC (default), Ethereum, BNB, Bitcoin, Solana, Polkadot, Cardano, Ripple, Tron, Tezos. For Bitcoin an extra selector picks SegWit (`bc1q…`), Legacy (`1…`) or Taproot (`bc1p…`). *The same phrase yields a different address in each format.*
- **Account #** — stepper to pick a derivation index (several accounts from one phrase).
- **Unlock wallet** — opens with the entered credentials.
- **Create a new wallet** — generates a fresh phrase (see below).
- **Restore from a backup file** — load an encrypted vault backup (`.json`); it asks for the PIN it was saved with.
- **Node settings** link — set the gateway URL before logging in.

2.4 Creating a new wallet

1. Pick a phrase length (12–24 words). Words are shown as numbered pills; **Copy phrase**, **Copy numbered list**, **New phrase** buttons.
2. A warning reminds you the phrase is the *only* backup. Tick *"I've written down my recovery phrase"*.
3. **Verify** — pick the right word for a few random positions.
4. **Set a PIN** — choose and confirm a 6-digit PIN. It encrypts the vault on this device so the wallet survives refreshes.

2.5 Returning — Unlock

Enter your PIN (or use the passkey button if set up). Links below: *Restore from a backup file* and *Reset wallet on this device* (wipes local data; the phrase is your restore).

2.6 Watch-only

Watch an address shows balance & activity for any address without a key. The hero shows a *watch-only* badge and all signing pages are hidden.

---

3. The main interface

3.1 Top bar

- **Logo + version badge** (e.g. V0.36).
- **Notifications bell** — client-derived alerts: incoming chat, signature requests, chain alerts; a red counter badge; entries deep-link to the right page (e.g. an invoice notification opens the pay page).
- **Theme** — toggle dark/light.
- **Lock** — immediately locks the wallet (PIN to reopen).
- **Settings** — opens the Settings page.
- **Calls relay indicator** — shows remaining relay data when relevant.
- **Network status pill** — dot + label: `block N` (connected; shows the *confirmed* height, tip − 3), `synchronizing…` (transient failure < 30 s), `chain unreachable` (continuous failure). **Clicking it opens the block explorer** (`explorer.<gateway-domain>`).

3.2 Balance hero (left column)

- **Balance** with a scope switch — **This wallet** / all wallets combined.
- The coin icon refreshes on tap; a chevron opens the **token panel** listing your token balances.
- **Spendable** — balance minus locked amounts (stakes, escrows, held offers).
- **Address chip** — account avatar (tap to **switch account**), editable wallet name, the address, and a copy button.
- Badges: *watch-only*, *multisig*.
- **Send** and **Receive** quick buttons.

3.3 Left menu

Collapsible groups (your open/closed state is remembered; whole sections can be hidden under Settings → Menu):

Group	Items
HOME	Dashboard · To-do · Active · Activity
WALLET	Send · Receive · Deposit · Withdraw · Scheduled · Multi-sig · Policy
ASSETS	Tokens · Files · Exchange · Marketplace
SOCIAL	Contacts · Online · Chat · Calls
BUSINESS	Invoices · Forms · Notary
APPS	Lobby · Single Player · 1 vs 1 · 3+ Players

Group	Items
ADVANCED	Keys & seeds · Paper wallet · Submit
SYSTEM	Settings · Node · Network · Gateways · Relays · Governance

On narrow screens the menu collapses to a top navbar / burger menu.

3.4 Footer

Legal & info pages: **Terms & Conditions**, **Privacy Policy**, **Help & Manual**, **Participate in the Project**.

---

4. HOME

4.1 Dashboard (Overview)

The landing page after unlock:

- **Your coins** — combined balances per coin across your wallets.
- **Your wallets** — a card per account (name, address, live balance with de-emphasized decimals, chain badge, multisig/watch badges). A status box surfaces items needing attention.
- **Your tokens** — tokens you hold, with an *Explore* shortcut to the Tokens page.
- **Your nodes** — registered nodes owned by your wallets (auto-refreshes every 10 min).

4.2 To-do

"Everything waiting on you — approve, settle, or reply." One list that aggregates:

- escrows where **you are the approver** (approve / reject inline),
- multisig transactions awaiting **your signature**,
- payment/escrow **requests** addressed to you,
- **invoices** awaiting your payment,
- app turns and expiring items.

A counter badge on the menu item shows how many. **Refresh** re-pulls everything. The page auto-refreshes every 60 s while open.

4.3 Active (Active transactions)

Transactions that are live right now, in four tabs:

- **All** — combined list with per-tab counters.
- **Approve / reject escrow** — as the named approver, enter (or pick) an escrow transaction ID and **Approve & release** (funds go to the recipient) or **Reject & return** (funds go back to the sender). Broadcasts *ApprovalEscrow* (type 4).
- **Stop a liquid payment** — halt a liquid payment you started; the unvested remainder returns to you (*LiquidPaymentStop*, type 262).
- **Stop a vested** — running schedules you created; revoking returns the unfired remainder (tranches already delivered stay with the recipient) (*CancelSchedule*, type 30).

4.4 Activity

Your transaction history.

- **Account filter** — one wallet or all.
- **Coin filter** — all coins or a specific one.
- **Source selector** — **Latest TX** (the connected node's recent window) or **All TX** (an archival node's full history).
- **Category chips** — All · Wallet · Trading · Apps · Tools · Coinbase · Fees, with per-category totals.
- Every transaction the wallet broadcasts is also kept in a local **journal** (with its body bytes), so it appears instantly, survives node pruning, and renders full per-type detail. Rows expand to show amounts, counterparties (contact names shown when known), fees, escrow details, messages (decrypted when they're for you), and a copy-hash button.

---

5. WALLET

5.1 Send

The main transfer page. Fields top to bottom:

- **Recipient** — any supported address format (`ZBC_…` , `0x…` , `1…` , `bc1…` , Solana, SS58, `addr1…` , `r…` , `T…` , `tz1…`) or a contact (picker button). A validity

indicator and the resolved contact name appear under the field.

- **Amount** + **Max** (fills the spendable balance minus fee).
- **Network fee (ZBC)** — pre-filled from the network; grows with message size; you can raise it.
- **Message** (optional) — stored **on-chain**. For ZBC → ZBC transfers you can tick **Encrypt message** (end-to-end; only the recipient can read it). The fee hint updates with the byte size.
- **Private note** (optional) — saved **only on this device**, never on-chain.
- **Longevity** (contextual) — appears when sending to a `ZBS_` object address, to label a rent top-up.
- **Liquid payment** checkbox — the amount **vests gradually** to the recipient over a chosen period (minutes). The recipient can claim proportionally as it vests; you can stop it early from *Active* (unvested part returns). Sends *LiquidPayment* (type 6) instead of a plain transfer.
- **Escrow** checkbox — hold the funds until an approver approves:
- *Approver address* (or contact),
- *Hold until* — preset chips (1 day … 6 months) or an exact date/time picker; shown as a block-height timeout with the note that block timing is an **estimate** and actual timing will drift,
- *Approver commission (ZBC)*,
- *Instruction* (free text, e.g. "release on delivery").
- A link: *Expecting to be paid instead? → Request a payment* (receiver-side escrow, on the Scheduled page).
- **Pay the network fee in this token** (contextual) — for fee-enabled tokens: burns fee-tokens and releases an equal slice of the ZBC backing (not available under escrow).
- **Create transaction only** — signs but doesn't broadcast: produces the signed JSON + QR to submit later or from another device (see Submit).
- **Preview** — amount, commission, fee, total.
- **Review & send** — confirmation modal, then broadcast. **Reset form** clears everything. After sending, the transaction ID is shown (kept even if you navigate away) with copy and explorer links.

What it broadcasts: SendZBC (type 1) — or TransferToken (11) when a token is selected in the hero, or LiquidPayment (6) when the liquid box is ticked. Escrow rides in the envelope of any of them.

5.2 Receive

Your address as a large QR code, the address in text, and a **Copy address** button.

5.3 Deposit

Bridge **into** ZooBC: bring BTC, ETH, USDC or USDT onto the chain as wrapper coins (e.g. BTC → **ZBTC**), minted 1:1 after confirmation. Two tabs:

- **Start** — pick a coin (unsupported ones show *Coming soon*). The wallet shows:
- the **custody deposit address** (QR + copy) for the origin chain,
- required **confirmations**,
- **Who gets credited**: the deposit credits the ZooBC account that **mirrors the address you send from** (same key) — the custody address is shared, so the sending address is the only proof of ownership. Send from a wallet whose key you control here.
- After confirmations, a 2/3 supermajority of ZooBC nodes attests, then the wrapper mints (minutes up to ~1 hour).
- **Track** — three sub-tabs: **In-flight** (auto-refreshes every 20 s; states *Detected → Confirming → Attesting → Minted*, with per-chain ETA), **Watch another account** (any origin-chain or ZBC_ address), **History** (completed & dropped).

5.4 Withdraw

Bridge **out of** ZooBC (wrapper coins back to their origin chains). **This screen is a mock-up — the outbound bridge is not functional yet**; it shows the planned coin list and flow.

5.5 Scheduled

Value that moves on a timer. Five tabs:

- **Vesting** — pre-locked schedule: the **full total is locked at creation** and released to the recipient in tranches. Fields: recipient, coin/token, per-tranche amount, interval, number of tranches (1–520), optional cliff, revocable/irrevocable, optional end time. Broadcasts *ScheduledTransfer* (29) with funding_mode 0.
- **Recurring** — pull-at-fire: nothing is locked; each tranche is pulled from your balance when it fires (fails gracefully if empty). Same fields; funding_mode 1.

- **At height** — a one-shot **trigger**: locks an amount now, pays the beneficiary when the chain reaches a block height. The height field has a date/time calendar helper (converts to an estimated height; actual timing drifts with block time). Broadcasts *CreateTrigger* (15); cancel refunds via *CancelTrigger* (16).
- **Request** — the receiver-side escrow: **you request a payment** — set the payer, amount (ZBC or token), approver, deadline, commission and instruction; the payer receives it in To-do and funds it. Broadcasts *EscrowRequest* (260).
- **List** — every schedule you created or receive, grouped; each row shows kind (Vesting/Recurring/One-shot), progress (delivered of total), next fire, role badge, and actions: **Revoke** (if revocable), **Reassign** the remaining tranches to a new recipient (*ReassignSchedule*, 31 — current recipient only).

5.6 Multi-sig

An account that needs several signatures for each spend. Two tabs:

- **Create multisig account** — participant addresses (one per line, any format; *Add from contacts*), **Min signatures**, **Nonce**. **Compute address** shows the deterministic ZBC_ address. Then:
- **Register on-chain** — records the participant set (*MultisigRegistration*, type 5),
- **Add to my wallet** — makes it selectable as an account (multisig badge) to receive and propose spends.
- **Sign multisig transaction** — pending items addressed to you appear on top. Paste a transaction from a previous signer (hex / base64 / JSON), **Sign with my key**, then pass the output to the next signer — signing never broadcasts. **Broadcast (Submit page)** opens Submit with the payload prefilled once enough signatures are collected.

5.7 Policy — "What I transact with"

Per-account, on-chain **category opt-out**, enforced by every node. Turning a category off is **bidirectional**: you can neither send nor receive it; anyone who tries is bounced (no funds move; they pay only the network fee). Nine toggles:

Bit	Category	Covers
0	Payments	Send ZBC, liquid / scheduled / trigger transfers

Bit	Category	Covers
1	Apps	Create, join, move, resign, timeout, settle
2	Tokens	Issue, transfer, mint, burn, finance colored coins
3	Exchange	Swap offers and order-book markets
4	Data & Storage	Datasets, prepaid storage, files (DFS), storage proofs
5	Escrow & Multisig	Escrow approval / request, multisignature
6	Node & Infrastructure	Node registration, gateway / archival / relay registry
7	Bridge	Cross-chain attestation / mint
8	Governance	Fee votes, signed-release governance

A summary chip shows *"Transacting with everything"* or *"Refusing: …"*. Switching off **Payments** or **Node & Infrastructure** asks for an extra confirmation (they can lock you out of everyday use). **Save policy** broadcasts *SetTransactPolicy* (type 50, mask u16 LE, ~0.05 ZBC fee); it applies once the block settles.

---

6. ASSETS

6.1 Tokens

Registry of every coin on the chain (genesis + user-issued "colored coins"). Two modes:

- **Browse** — searchable list of all registered tokens with icon, symbol, name, supply, backing, your balance; you can give a token a local nickname. Backed tokens keep a constant ZBC value.
- **Issue & manage** (also reachable via Multi-sig hub → Tokens):
- **Issue a token** — Symbol (validated against charset + reserved-symbol blocklist), Name, Total supply, Decimals (0–8), **Backing (ZBC locked)** — 0 for an unbacked token, flags **Redeemable** (burn → reclaim your share of backing) and **Mintable** (issuer can add more), optional **SVG logo** + description (ride in the tx message, ≤ 6,144 bytes). Broadcasts *IssueToken* (10).

- **Send a token** — token, amount, recipient (*TransferToken*, 11; supports escrow and fee-in-token).
- **Mint or burn** — *Mint* (issuer only, mintable tokens; adds supply **and** locks more backing so unit value stays constant — type 12), *Burn / redeem* (destroys your tokens; redeemable tokens return your share of the backing — type 13), *Finance* (tops up the token's survival rent so it isn't pruned — type 14).

6.2 Files

Decentralized file storage (DFS) plus all data-lifecycle tools. Six tabs:

- **My Files** — files you stored: name, size, root hash, replica health, rent status; download or copy a share link.
- **Upload file** — the node splits the file into content-addressed pieces and seeds them to its replica set; the wallet then anchors it on-chain with *StoreFile* (type 40): file root (32 B), size, piece size, **rent deposit (min 0.01 ZBC)** and the piece list. If your files list is empty this tab opens by default.
- **Download by root** — fetch any file by its 64-hex root (or ZTX_ id) from the storage network.
- **Datasets** — your raw on-chain entries. Every chat message, poll, invoice, form and profile field is a **dataset entry** on an account:
- **My entries** — list with one-tap Remove.
- **Add an entry** — account (blank = your own), property, value (max 4,096 bytes; live byte counter). Broadcasts *SetupAccountDataset* (type 3).
- **Remove an entry manually** — must match recipient + property (incl. the `#…` suffix shown in Chat) + value exactly. Broadcasts *RemoveAccountDataset* (259).
- **Fund storage** — prepay dataset rent for your account (*FundStorage*, type 9). Per-dataset targeting arrives with the next chain.
- **Objects** — survival-rent object management:
- list of your objects with rent balance and expiry estimate,
- **Transfer ownership** — two-step: you offer (*TransferDataset*, 42), the recipient accepts (*AcceptDataset*, 44),
- **Access policy** — allow/deny lists of accounts (*SetDatasetPolicy*, 43),
- **Delete** (*DeleteDataset*, 45),
- **Fund any object by creating-tx hash** — look up the object's **ZBS_ deposit** from its transaction ID and top it up with a normal (escrowable) transfer.

6.3 Exchange

An on-chain **order-book exchange** (CLOB). Header badge shows **CONNECTING… / LIVE / OFFLINE** honestly; nothing is simulated when the endpoints are down.

- **Modes** — **Simple** (form first, no book/trades) and **Pro** (depth chart, order book, recent trades, orders — trading-terminal layout).
- **Pair picker** — From / Into rows listing every registered coin; flip button; markets that exist show as pills (with your open-order count).
- **Stats bar** — last price, 24 h change/high/low/volume, sparkline.
- **Market depth** & **Order book** — live from the chain; empty-book states explain: *"No orders yet — be the first"* / node offline / read-API missing.
- **Order form** — Buy/Sell, order type **Market / Limit / Stop**, amount (+Max from your live balance), limit price, total, estimated fee (0.1%). Market orders on an empty book are blocked with an explanation (no price to match). Placing an order broadcasts *PlaceOrder* (type 22); if the market for the pair doesn't exist yet you're offered **Create market** (*CreateMarket*, 21, with a ZBC rent deposit).
- **My orders** — persisted per device; shows *Awaiting block* while pending, then open/filled/cancelled; **Cancel** broadcasts *CancelOrder* (23, refunds the unfilled remainder); **Clear filled** tidies the list.

6.4 Marketplace

A peer-to-peer **swap-offer board** (OTC): *"post what you give and what you want; anyone can accept."* Fully atomic — the node swaps both sides in one block or not at all. LIVE-only, with an honest OFFLINE state.

- **Browse offers** — filter by Give / Want asset; each offer card shows maker (avatar + contact name), give → want amounts, implied price, expiry and status (Open / Filled / Cancelled / Expired). **Accept** broadcasts *AcceptSwapOffer* (19) — the taker pays the want-side, receives the held give-side.
- **Create offer** — give asset + amount (locked from your balance while the offer is open), want asset + amount, optional **expiry** (0 = good-till-cancelled), optional **reserved counterparty** (only that address may accept). Shows a confirmation card, then broadcasts *CreateSwapOffer* (18).
- **My offers** — your open offers with **Cancel** (*CancelSwapOffer*, 20 — refunds the held amount) plus a device-local history of sold / accepted / cancelled /

expired offers.

---

7. SOCIAL

7.1 Contacts

Your address book. **Add contact** — name, optional photo and note, and any number of addresses across chains (each labeled with a chain badge). Contact names replace raw addresses throughout the wallet (Send, Activity, Chat, Marketplace…) — the raw address is always one tap away to copy. Contacts live in the encrypted vault and travel with backups.

7.2 Online

Live presence of your contacts. Each row shows status (online / away / offline), with quick actions to chat or call. Presence is **push-based**: wallets beacon every ~60 s through the relay, and the list repaints on a timer — two wallets see each other without any navigation. How *you* appear is controlled under Calls → Appearance.

7.3 Chat

On-chain messaging. Two layers:

- **Conversation view** — WhatsApp-style: conversation list with unread badges, avatars (contact photo or initials), search, local rename; the thread view shows bubbles, timestamps and a lock icon on end-to-end encrypted messages (ZBC ↔ ZBC). Incoming messages from any of your own wallets are recognized as "you".
- **Raw send form** — *To* (address or contact), *Topic* (optional; e.g. a app id for moves — default `chat`), *Message*. Each message is a **SetupAccountDataset** transaction (type 3) on the recipient's account and costs a normal network fee.

Messages can be deleted via Files → Datasets (removal must match the entry exactly).

7.4 Calls

Voice, video and screen-share, peer-to-peer. **Direct calls are free**; when a firewall blocks a direct link, a **relay gateway** carries the traffic for credit you top up. Five tabs:

- **Online contacts** — who's reachable now (*Online only / All* filter); tap to start voice/video.
- **Relays** — your **relay credit**, shown as remaining **MB of data** plus derived estimates (voice ≈ 0.36 MB/min, video ≈ 15 MB/min — credit is pure data, not minutes). List of relay gateways with health; **Add gateway**; top-ups are a ZBC payment to the relay operator and confirm on-chain (pending top-ups keep polling until credited).
- **Recent calls** — log with direction, duration, route (direct / relayed), data used and gateway.
- **Enable accounts** — choose which of your accounts are call-enabled (signed enrollment with the relay).
- **Appearance** — how you appear: **Everyone / All contacts / Selected contacts / Nobody** (with a per-contact picker for *Selected*). "Nobody" still lets you start calls.

During a call: mute, camera toggle, screen-share, and a live data meter. The top-bar relay indicator shows remaining data during relayed calls.

---

8. BUSINESS

8.1 Invoices

Send a payable invoice by link — in **ZBC or any registered coin/token** (ZUSD, ZBTC…). Payment is a normal transfer in that currency; status reconciles itself from the chain.

- **Create invoice** — amount + currency, description, optional due date; the invoice is written as a dataset on your account and you get a share link (see 8.4).
- Two tabs: **Issued by me** and **Received**, each grouped **Awaiting payment** vs **Paid**. Paid rows link to the settling transaction. Incoming invoices also appear in notifications and To-do, deep-linking to the pay page.
- **Pay page** — opened from a link: shows issuer, amount, currency and memo; one tap builds the exact transfer.

8.2 Forms

Publish a form, share a link, collect **signed & encrypted submissions** only you can read — with proof of who signed what. Define fields, publish (a dataset on your account), share the link; submissions are encrypted to your key and listed with the signer's address and timestamp.

8.3 Notary (Proof of Existence)

Timestamp any file on-chain — **the document never leaves your device.**

- **Notarize** — drop a file; it's hashed locally (SHA3-256). For images you can optionally attach a small **thumbnail** (with an encrypt-to-self choice); PDFs get an honest "no preview" note. The hash (+ optional thumbnail) is written to your account as a dataset, financed by its own storage rent.
- **My anchors** — your anchors with block height/time; pending ones (submitted, not yet on-chain) are tracked locally until confirmed.
- **Verify** — drop a file to re-hash and search for its anchor — on your own account by default, or paste any ZBC_ address / link to verify someone else's.

8.4 Share links

Links to invoices, forms, polls and pay pages adapt to how the wallet is served: from a real `https://` origin they're full URLs; from a local file they're **#fragment links** — the recipient pastes the fragment after `…wallet.html` in their own wallet (no `file://` path is ever baked in).

---

9. APPS

The Arcade: on-chain, stake-based turn apps. The lobby gates honestly — when the chain is unreachable an overlay blocks play until it's back.

9.1 Views

- **Lobby** — network **prize pool** (ZBC + tokens), your open matches and challenges (join buttons), stats, and the app library.
- **Single Player** — solo apps vs the House (dice, coinflip, roulette, slots, lottery, crash…).

- **1 vs 1** — turn apps: tic-tac-toe, chess, connect-4, checkers, reversi, gomoku, battleship, dots & boxes… Challenge a specific opponent (address/contact — the invite lands in *their* wallet) or post an **open challenge** anyone can join.
- **3+ Players** — party apps (ludo, pig, race, monopoly-style).

9.2 Starting a match

The new-match form: app, **stake** (amount + coin — ZBC or any token from the pool list; 0 = friendly), seats, opponent (1 v 1) and, where supported, **fast mode** (state channel). Creating broadcasts *CreateApp* (24), locking your stake; a joiner locks theirs with *JoinApp* (25).

9.3 Playing

The board view shows the status bar (whose turn, countdown), the board, players rail and move history. Move legality is enforced client-side; the node's app VM re-validates every move.

- **On-chain mode** — every move is a *AppMove* (26) transaction; the countdown floors at the measured block time. If the wallet ever desyncs it can **rebuild the whole app from the chain**.
- **Fast mode (channels)** — moves are exchanged instantly as signed **vouchers** (retransmitted every ~10 s until acknowledged); at the end either player submits *SettleApp* (39) and the chain replays the signed vouchers through its own rules to adjudicate and pay out.
- **Resign** (27) forfeits — the stake goes to the opponent. **Claim timeout** (28) claims the win if the opponent abandoned past the per-move deadline.
- Leaving is safe: unfinished rounds persist locally — reopen the same app to reconnect; the outcome is sealed on-chain.

---

10. ADVANCED

10.1 Keys & seeds

Manage every identity in the vault:

- **Seeds** — named seed phrases. From each seed you can derive accounts on any supported chain and account index without re-entering the phrase. Reveal (PIN-gated), rename, remove.

- **Import a key** — a single 64-hex private key in any format.
- **Watch an address** — add a watch-only entry.
- **Imported keys & watched addresses** list — with per-row address, live balance, and remove.

10.2 Paper wallet

Printable cold storage. Controls: **Source** — *My wallet* (an account you hold) or *New empty wallet* (fresh phrase generated on the spot, never stored); layout options; **Print**. The sheet contains the address QR + the phrase/key with a strong handling warning.

10.3 Submit

Broadcast a transaction that was **signed elsewhere** (air-gapped device, multisig co-signer, "create only" from Send). Paste the signed JSON or **Scan QR** with the camera. No key is needed. Reports the resulting transaction ID or the node's error verbatim.

---

11. SYSTEM

11.1 Settings

Grouped into sub-tabs:

- **Security** — change the 6-digit PIN (backups downloaded afterwards need the new PIN).
- **Passkey** — set up / remove platform-passkey unlock (Face ID / Touch ID / Windows Hello).
- **Auto-lock** — lock after a period of inactivity.
- **Menu** — show/hide whole sidebar sections (device preference). Only HOME is always shown; Settings stays reachable from the top bar.
- **Backup & Restore** — download the **encrypted vault** (needs your PIN to open; anyone with file + PIN can spend) / restore from a backup file.
- **Node** (also reachable pre-login) — the current gateway with health dot, **List gateways & check who's online** (from the on-chain registry), a **manual node URL entry** for extra fallback nodes, and advanced node settings.
- **Theme** — dark / light.

- **Air-gap mode** — when ON the wallet never broadcasts: every transaction is signed and exported (JSON + QR) for a connected device to submit. An indicator shows in the top bar.
- **Reset** — wipe this device (vault, caches); your phrase restores everything that matters.

11.2 Node

Everything for running a **validator node** (ZooBC accounts only):

- **My nodes** — every node owned by your wallets (all accounts or one); name them locally, copy their keys.
- **Register a node** — paste the node's 64-hex secret key + **locked balance** (min 1 ZBC, default 1000). The wallet fetches a recent block and builds a **proof of ownership (POOWN)** — the node key signs `owner ‖ block hash ‖ height` — then broadcasts *NodeRegistration* (type 2). The stake is released if you remove the node.
- **Remove a node** — deregister; locked balance released (*RemoveNode*, 514).
- **Update node stake** — new locked balance (can only stay the same or **increase**); fresh POOWN (*UpdateNode*, 258).
- **Claim a node** — re-assign a node's ownership to this wallet, proving control with its secret key (*ClaimNode*, 770).

11.3 Network

Your on-chain infrastructure records, with a live overview of the three registries:

- **Register a gateway** — jump to the Gateways form (locks a **10 ZBC stake**, refunded on unregister; the gateway must then heartbeat or it's auto-pruned with the stake refunded).
- **Register / update a relay** — publish a relay for **discovery**: relay key (ZBR_/64-hex), which gateway it serves, domain, URL (*RegisterRelay*, 48; *Unregister*, 49). The off-chain relay-bind certificate (below) still authenticates it — this record only makes it findable.
- **Register / update an archival node** — mark one of your registered nodes as serving full history + the explorer API: node key (ZNK_/64-hex), domain, URL (*RegisterArchival*, 46; *Unregister*, 47). Puts the **Archival** badge on it in the explorer.

Re-registering with the same key updates domain/URL; only the owner's re-announce/unregister has effect.

11.4 Gateways

- **Gateways list** — the gateway this wallet talks to, with a health check across all registered gateways (**List gateways & check who's online**); pick the one to connect through. *Advanced node settings…* opens the raw URL config.
- **Register a gateway** — **Generate gateway key** (or paste 64-hex; the ZBG_ address preview updates live), domain, URL → *RegisterGateway* (36), locking the 10 ZBC stake. Heartbeats (37) are emitted automatically by the running gateway.
- **Unregister a gateway** — by key; refunds the stake (*UnregisterGateway*, 38).

11.5 Relays

Register a relay (operator flow, **no transaction, no fee**): choose the gateway whose ZBG_ key signs, set certificate validity (days), **Generate relay key & certificate**. The wallet produces the relay's ZBR_ key and a certificate signed by your gateway key; paste the output into the relay server's `/etc/zoobc/relay.env` and restart — its log shows `[fed] ON` and it links to peer gateways. Re-run any time to rotate the key.

11.6 Governance

- **Polls** — a poll is an on-chain record (a dataset): question, options, description, voting window, eligibility rules (min ZBC balance, min account age, min transaction count). **New poll** opens the editor; **Open a poll link…** pastes a shared `#gov=poll/...` link. Each vote is a signed transaction from the voter's account; tallies read straight from the chain. Poll cards show status (upcoming / open / closed), vote counts, copy-link and Open.
- **Fee voting** — vote the network **fee scale** up or down (1.0× = normal). The **median** of all node votes moves it, clamped to 0.5×–2.0× per period. Two steps:

1. **Commit** — your vote is broadcast as a hash (*FeeVoteCommit*, type 7); the wallet remembers the underlying vote on this device.
2. **Reveal** — in the reveal window, broadcast the vote + your raw signature (*FeeVoteReveal*, 263).

---

12. Transaction reference

All integers little-endian; amounts in atomic units (×10⁸ for ZBC, ×10^decimals for tokens). Every transaction shares the same envelope: `type(u32) · version(1) · timestamp(u64) · sender(typed acct) · recipient(typed acct; 02000000 = none) · fee(u64) · body_len(u32) · body · escrow-or-marker · message_len(u32) · message`. The escrow block (any escrowable type) is `approver(typed) · commission(u64) · timeout(u64) · instruction(lp4) · multi_party(1)`; without escrow a 4-byte marker (=2) is written. ZBC accounts sign ed25519 over SHA3-256 of the tx bytes; other formats sign with their native scheme over the same digest.

Payments & escrow

Type	Name	Body	Notes
1	**SendZBC**	amount(8)	Recipient in envelope. Escrow + on-chain message (optionally encrypted ZBC→ZBC) supported.
6	**LiquidPayment**	amount(8) · complete_minutes(8) · [token_id(8)] · [fee_in_token(1)]	Vests linearly to the recipient over the period; recipient claims as it vests. Token streams append token_id last.
262	**LiquidPaymentStop**	tx_id(8)	Sender stops the stream; unvested remainder returns.
4	**ApprovalEscrow**	action(u32: 0=approve, 1=reject) · escrow_tx_id(8)	Signed by the named approver; approve releases to the recipient, reject returns to the sender. On timeout the chain auto-resolves.
260	**EscrowRequest**	token_id(8) · amount(8) · approver(typed) ·	Envelope recipient = the **payer**. Receiver-

Type	Name	Body	Notes
		timeout(8) · commission(8) · instruction(lp4)	side escrow: the payer funds it from To-do.
15	**CreateTrigger**	fire_height(8) · amount(8)	Envelope recipient = beneficiary. Locks now, pays at the block height.
16	**CancelTrigger**	trigger_id(8)	Refunds the locked amount to the owner.

Scheduler

Type	Name	Body	Notes
29	**ScheduledTransfer**	token_id(8) · per_fire(8) · interval_s(8) · fires(u32) · cliff_s(8) · funding_mode(1) · cancel_policy(1) · end_time(8) · reserved(1)	funding_mode 0 = Vesting (total pre-locked), 1 = Recurring (pull at fire). cancel_policy 0 = revocable. 1–520 tranches.
30	**CancelSchedule**	schedule_id(8)	Sender revoke (if revocable) or recipient decline; unfired remainder returns.
31	**ReassignSchedule**	schedule_id(8) · new_recipient(typed)	Current recipient only.

Messages & data

Type	Name	Body	Notes
3	**SetupAccountDataset**	prop_len(u32) · prop · val_len(u32) · val	Chat messages, invoices, forms, polls, notary anchors, profile fields. Max 4,096 bytes per entry;

Type	Name	Body	Notes
			storage rent applies.
259	**RemoveAccountDataset**	same layout	Must match account + property + value exactly.
9	**FundStorage**	amount(8)	Prepays your account's dataset rent.
40	**StoreFile**	file_root(32) · total_size(8) · piece_size(u32) · deposit(8) · piece_count(u32) · piece_ids(32×n)	Anchors a DFS upload; deposit ≥ 0.01 ZBC; root must match the pieces.
42	**TransferDataset**	object_id(32) · new_owner(typed 36)	Two-step ownership transfer (offer).
43	**SetDatasetPolicy**	object_id(32) · mode(1) · n_add(1)+adds · n_remove(1)+removes	Access-control lists (max 255 per change).
44	**AcceptDataset**	object_id(32)	Recipient accepts a transfer offer.
45	**DeleteDataset**	object_id(32)	Owner deletes the object.

object_id = the creating transaction's hash (ZTX_/64-hex). Longevity top-ups are plain SendZBC to the object's **ZBS_** deposit address (account type 10) and can be escrowed.

Tokens (colored coins)

Type	Name	Body	Notes
10	**IssueToken**	decimals(1) · flags(1) · supply(8) · backing(8) · sym(lp2) · name(lp2)	Flags: redeemable, mintable. SVG icon + description ride in the message (≤ 6,144 B).

Type	Name	Body	Notes
11	**TransferToken**	token_id(8) · amount(8) · fee_in_token(1)	Escrowable; under escrow the fee must be ZBC (fee_in_token forced 0).
12	**MintToken**	token_id(8) · add_supply(8) · add_backing(8)	Issuer only; keeps unit value constant.
13	**BurnToken**	token_id(8) · amount(8)	Redeemable tokens return your share of backing.
14	**FinanceToken**	token_id(8) · amount(8)	Tops up the token's survival rent.

Trading

Type	Name	Body	Notes
18	**CreateSwapOffer**	give_token(8) · give_amt(8) · want_token(8) · want_amt(8) · expiry(8; 0=GTC) · [counterparty(36)]	Locks the give-side. Counterparty makes it a reserved offer.
19	**AcceptSwapOffer**	offer_id(8)	Atomic swap: want-side taker→maker, held give-side→taker.
20	**CancelSwapOffer**	offer_id(8)	Maker only; refunds the hold.
21	**CreateMarket**	base_token(8) · quote_token(8) · deposit(8)	Order-book market; deposit is ZBC rent.
22	**PlaceOrder**	market_id(8) · side(1) · price(8, ×1e8) · amount(8, base) · flags(1: bit0 market, bit1 post-only) · expiry(8)	side 0 = buy base, 1 = sell.
23	**CancelOrder**	order_id(8)	Owner only; refunds unfilled remainder.

Apps

Type	Name	Body	Notes
24	**CreateApp**	app_type(1) · stake_token(8) · stake_amount(8) · seats(1) · params(lp2) · [opponent(36)] · [channel(1)]	Stake in **base units**. Seats: 1 solo, 2 duel, 3+ party. Opponent addresses the envelope so the invite shows in their stream. channel=1 = fast mode (2 seats only).
25	**JoinApp**	app_id(8)	Locks the joiner's stake.
26	**AppMove**	app_id(8) · move(lp2)	One move = one tx; the node's app VM re-validates.
27	**ResignApp**	app_id(8)	Stake to the opponent.
28	**ClaimTimeout**	app_id(8)	Win if the opponent abandoned past the per-move deadline.
39	**SettleApp**	app_id(8) · final_seq(u32) · move_count(u32) · [seat(1)·move(lp2)·sig(64)]*	Fast-mode settlement: the chain replays the signed vouchers and adjudicates.

Node & infrastructure

Type	Name	Body	Notes
2	**NodeRegistration**	node_pub(32) · owner(36) · locked(8) · POOWN(136)	Min 1 ZBC locked. POOWN = `owner‖blockhash‖height` (72) + node ed25519 sig(64) over it, from a fresh block.
514	**RemoveNode**	node_pub(32)	Releases the locked balance.
258	**UpdateNode**	node_pub(32) · locked(8) · POOWN(136)	Stake may only stay or increase.
770	**ClaimNode**	node_pub(32) · POOWN(136)	Re-assigns ownership to the signer.

Type	Name	Body	Notes
36	**RegisterGateway**	gw_key(32) · domain(lp4) · url(lp4)	Locks a protocol-fixed 10 ZBC stake.
37	**GatewayHeartbeat**	gw_key(32)	Liveness beat; keeps the record from pruning.
38	**UnregisterGateway**	gw_key(32)	Refunds the 10 ZBC stake.
46 / 47	**Register / UnregisterArchival**	node_pub(32) [· domain(lp4) · url(lp4)]	No stake, fee only. Owner-signed.
48 / 49	**Register / UnregisterRelay**	relay_key(32) [· gw_key(32) · domain(lp4) · url(lp4)]	Discovery record; the off-chain certificate still authenticates.

Governance & policy

Type	Name	Body	Notes
7	**FeeVoteCommit**	vote_hash(32)	SHA3-256 of FeeVoteInfo (blockhash‖height‖feeVote); vote kept locally for reveal. 1.0× ↔ feeVote 10000.
263	**FeeVoteReveal**	FeeVoteInfo(44) · sig_len(u32=64) · voter_sig(64)	Raw ed25519 over the info. Median applied, clamped 0.5×–2.0×/period.
5	**MultisigRegistration**	ver(u32) · min_sigs(u32) · nonce(u64) · n(u32) · sorted participants · 0 · 0	Address = ZBC encoding of SHA3-256 over the canonical participant set.
50	**SetTransactPolicy**	mask(u16)	Bit i set = account opts **out** of category i (bits 0–8, see §5.7). Self-signed only; enforced bidirectionally by every node.

13. Troubleshooting

- **"synchronizing…" in the top bar** — a transient network blip; the wallet waits 30 s of continuous failure before declaring *chain unreachable*, and meanwhile tries fallback nodes and known gateways automatically.
- **"New chain detected"** — the node's tip is far below the highest block this wallet has seen: the network was restarted from genesis. The wallet offers a one-click cleanup of old-chain caches (activity, tokens, apps, notifications); **keys, seeds and contacts are kept**.
- **Forgot the PIN** — after 5 wrong attempts the vault is wiped. Restore with your recovery phrase (or a backup file + its PIN).
- **A transaction is refused with only the fee charged** — the counterparty (or you) has a **Transact policy** refusing that category (§5.7).
- **Exchange/Marketplace/Apps show OFFLINE** — the node lacks those endpoints or is unreachable; the tabs keep probing and flip to LIVE by themselves.
- **App froze / browser closed mid-app** — reopen the same app from the Lobby to reconnect; on-chain apps rebuild from the chain, fast-mode apps retransmit vouchers automatically.
- **Sent a message/transaction that isn't in Activity from another device** — Activity's *Latest TX* source shows the connected node's recent window; switch the source to **All TX** (archival) for full history.
- **Share link starts with #** — you're running from a local file; the recipient pastes the fragment after `…wallet.html` on their own copy or any gateway-hosted wallet.

---

14. Glossary

- **Atomic unit** — smallest on-chain unit; 10^{-8} ZBC.
- **Backed token** — token with ZBC locked behind it; burning redeems the pro-rata backing.
- **Dataset** — a property→value record on an account; the chain's general data store.
- **Escrow** — funds held until a named approver approves, rejects, or the deadline auto-resolves.

- **Fast mode / channel** — off-chain signed move vouchers settled by one final transaction.
- **Gateway** — the web server the wallet connects through (ZBG_ key).
- **Liquid payment** — a transfer that vests continuously over time.
- **POOWN** — proof of node ownership: the node key signs owner + fresh block hash + height.
- **Relay** — call/presence server run by a gateway (ZBR_ key).
- **Spendable** — balance minus everything locked (stakes, escrows, offers, schedules).
- **Vault** — the PIN-encrypted local store holding seeds, keys and contacts.
- **Watch-only** — an address added without its key; view but never sign.
- **ZBS_ address** — an object's rent-deposit account, fundable like any address.
- **ZTX_** — display form of a transaction hash.