Hydra Point-of-Sale (PoS) & Invoicing - Requirements Specification Document

Purpose

Define the complete functional and non-functional requirements for an open-source, self-hosted, non-custodial PoS and invoicing toolkit that accepts ada and cardano native tokens (cnts), with hydra layer-2 settlement and automatic layer-1 fallback.

Goals

1. Enable merchants to accept ada/cnts without custody, without platform fees, and without kyc.
2. Provide a retail-friendly pos UI and an invoicing module (qr-based payments).
3. Provide an API and webhooks for integrations (e-commerce, ERP, custom apps).
4. Support hydra for fast confirmation; fall back to cardano layer 1 seamlessly.

Scope

In scope

- Web-based pos UI (tablet/kiosk-friendly)
- Invoicing (create/send/export/track)
- ada + selected cnt support (merchant-configurable allowlist)
- qr payment requests
- Payment detection and status updates (hydra and/or layer 1)
- Hydra head integration (layer 2) and hybrid routing and layer-1 fallback

- Merchant wallet connection (non-custodial)
- Roles/permissions (merchant owner vs staff)
- Reporting and exports (csv/excel + pdf invoices/receipts)
- Rest API + webhooks

Out of scope (v1.0)

- A hosted custodial service
- Built-in fiat onramp/offramp
- Hardware-specific pos terminal firmware

5. Personas and user stories

Personas

- **Merchant owner:** configures store, tokens, staff, settlement preferences.
- **Cashier/staff:** creates carts, requests payment, confirms received, prints receipts.
- **Customer:** scans qr and pays using a cardano wallet.
- **Developer/integrator:** uses API and webhooks to integrate with systems.

Sample user stories

- As a cashier, I want to ring up items quickly and show a qr so the customer can pay.
- As a merchant, I want to accept a cnt, not every token.
- As a merchant, I want instant hydra confirmation, but if hydra is down, still take payment on layer 1.
- As an integrator, I want webhooks when invoices are paid so I can fulfill orders.

Assumptions and constraints

- Merchant runs the software in an environment they control (self-hosted).
- Customer uses a cardano wallet capable of scanning/reading the payment qr format.

Feature set (high-level)

- Invoice creation + tracking + export (pdf + csv/excel)
- Accept ada
- Payment status lifecycle + notifications
- hydra layer-2 settlement + layer-1 fallback
- API + webhooks
- Basic staff roles
- Audit trail for transactions

Functional requirements

Merchant onboarding and configuration

- **fr-001:** the system will allow creating a store profile (name, address, contact, timezone, currency display).
- **fr-002:** the system will support connecting a merchant wallet in a non-custodial manner.
- **fr-003:** the system will allow the merchant to set default settlement mode: hydra-first, layer-1-only, or hydra-only.
- **fr-004:** the system will support setting invoice expiry defaults.

Checkout and payment request (qr)

- **fr-020:** the system will generate a payment request qr code containing:
  - asset
  - amount

    - expiry timestamp
- **fr-021:** the system will display a real-time payment status screen.

Invoicing module

- **fr-030:** the system will allow creating invoices with:
    - asset + amount
    - expiry
    - unique invoice id/reference
- **fr-031:** the system will provide invoice statuses: draft, issued, pending_payment, paid, expired, cancelled, failed.
- **fr-032:** the system will export invoices to pdf and csv/excel.
- **fr-033:** the system will support sending an invoice link/qr.

Payment processing

- **fr-040:** the system will support layer-1 payment requests using standard cardano addresses.
- **fr-042:** the system will mark an invoice/order as paid only after a merchant-defined confirmation rule.
- **fr-043:** the system will store the resulting transaction hash for reconciliation.

Payment processing

- **fr-050:** the system will support hydra head connection parameters (head endpoint, participants, network).
- **fr-051:** the system will create hydra-mode payment requests when hydra is active and the merchant selects hydra-first/hydra-only.
- **fr-052:** the system will detect and confirm hydra payments with near-real-time settlement semantics for the head.
- **fr-053:** the system will record hydra transaction/reference identifiers for audit.

Hybrid routing and fallback (hydra → layer 1)

- **fr-060:** the system will automatically select the payment rail using this logic (default):
    1. if hydra is configured + head is reachable + merchant mode allows hydra, use hydra
    2. otherwise, fall back to layer 1
- **fr-061:** if a request is created in hydra mode and hydra becomes unavailable before payment, the system will:
    - mark the request as fallback_available

Token support

- **fr-070:** the system will support pricing and accepting cnts by unit amount.
- **fr-071:** the system will validate cnt identity using policy id + asset name.

10. Non-functional requirements

Security and privacy

- **nfr-001:** private keys are not transmitted to any third-party service.
- **nfr-002:** the system includes safe cookie/session defaults.

Reliability and availability

- **nfr-010:** hydra head downtime is handled gracefully and layer-1 fallback remains available.
- **nfr-011:** invoice/payment state persists across restarts.

Performance

- **nfr-020:** common pos UI actions respond quickly.
- **nfr-021:** payment status updates arrive within 1–20 seconds in hydra mode when available.

Maintainability and observability

- **nfr-030:** structured logs exist for payment lifecycle events.

- **nfr-031:** health endpoints exist for dependencies (hydra connector, layer-1 connector, db).

Integration plan

Payment rails overview

- **Hydra rail:** used when merchant has an active, reachable hydra head and mode allows hydra.
- **Layer-1 rail:** available fallback using standard cardano payments to a merchant address.

Rail selection rules

- Use hydra when:
  - hydra is configured, and
  - head endpoint is reachable, and
  - head reports a ready/active state, and
  - merchant mode is hydra-first or hydra-only
- Otherwise use layer 1.