Architecture Blueprint & Feasibility Documentation: Hydra Mobile SDK for Android & iOS

The Hydra Mobile SDK enables **native mobile applications** (Android, iOS, Flutter) to interact with **Hydra Heads** for real-time, low-fee Layer-2 payments on Cardano. This document validates feasibility, defines the client workflow, specifies APIs and integration points, documents security and scalability strategies, and provides a complete system architecture blueprint.

Client Workflow (End-to-End)

Normal Operation Flow

Connect → Open/Join Head → Commit → In-Head Payments → Close → Fan-out

1. **Connect**
   - Mobile SDK establishes a secure WebSocket (TLS) connection to `hydra-node`.
   - Connection state and last known snapshot are restored if resuming.
2. **Open / Join Head**
   - Client issues `openHead()` or `joinHead()` via HTTP API.
   - SDK subscribes to Head lifecycle events.
3. **Commit**
   - User commits UTxOs from L1 into the Head.
   - SDK tracks commit confirmation via snapshot events.
4. **In-Head Payments**

- L2 transactions created and signed locally.
- Transactions submitted via WebSocket.
- Validated via `TxValid` / `TxInvalid` events.

5. **Close Head**
   - Client initiates Head closure.
   - SDK monitors contestation and final snapshot.
6. **Fan-out**
   - Final balances settled on L1.
   - SDK confirms completion and clears local session state.

Failure & Recovery Paths

- **App Backgrounding / Kill** → Snapshot-based resume
- **Network Switch (WiFi ↔ Mobile)** → Reconnect + resync
- **Missed Events** → Snapshot replay
- **Partial Signing Failure** → Retry or user prompt

These flows ensure **crash safety and session continuity** under mobile constraints.

API & Integration Points

3.1 Hydra-Node Interfaces

WebSocket (Real-time):

- `TxValid`
- `TxInvalid`
- `Snapshot`
- `HeadOpened`
- `HeadClosed`

HTTP (Commands):

- `/open`
- `/join`
- `/commit`
- `/close`
- `/fanout`
- `/status`

Wallet Bridge Options

To support signing on mobile:

- **Deep Links** – external wallet i
- **WebView Injection** – embedded wallet context

Supports:

- Hydra L2 signing keys
- Cardano L1 signing via compatible wallets

Transaction Build Adapters

- **Dart (Flutter)** tx builder
- Deterministic transaction encoding
- Abstracted signing interface
- Pluggable signer backends

Security & Scalability

Security Controls

- TLS with certificate pinning
- Hardware-backed key storage (Secure Enclave / Keystore)

- No private key material leaves device
- Strict message schema validation
- Replay protection at SDK and protocol layers

Scalability & Back-Pressure

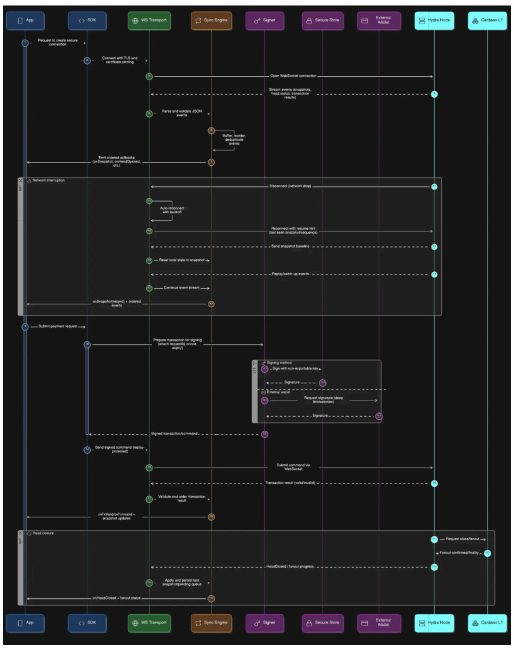
- Buffered event pipeline
- Ordered message processing
- Back-pressure handling for snapshot floods
- Non-blocking WebSocket design

Hydra throughput is preserved by ensuring the SDK never becomes the bottleneck.

Risk Evaluation & Mitigation

Risk	Impact	Mitigation
App backgrounding	Missed events	Snapshot-based recovery
Network instability	Disconnections	Auto-reconnect & resync
Key compromise	Loss of funds	Hardware-backed storage
Hydra online requirement	Session drops	Provider-managed Heads
Event ordering issues	Invalid state	Sequenced processing

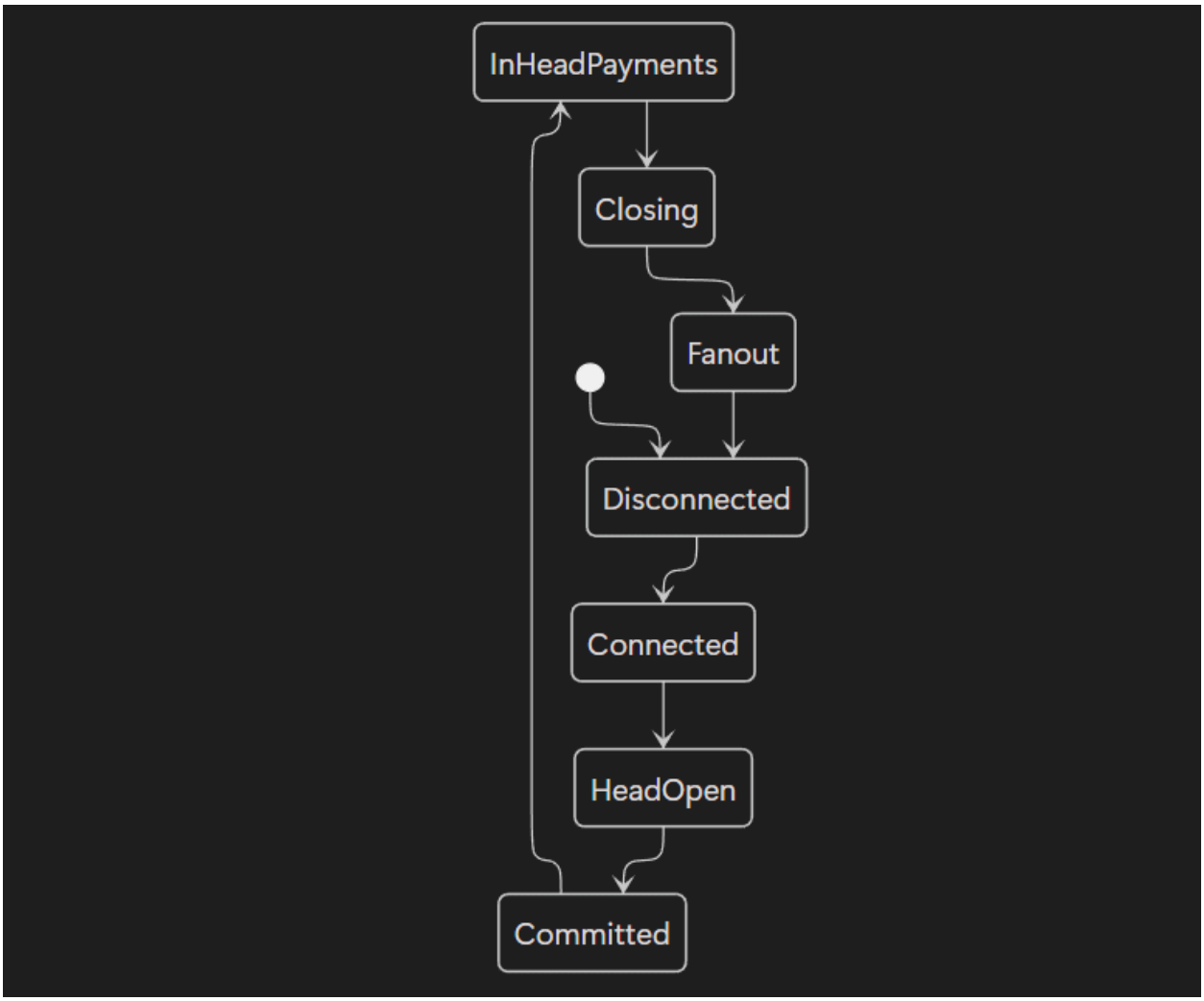
System Architecture Blueprint



Module Layout

- **Transport Layer** – WebSocket + HTTP clients
- **Sync Engine** – Snapshot recovery & ordering
- **Signer** – Hydra + L1 signing abstractions
- **Storage** – Persistent state & keys
- **Public SDK API** – Developer-facing interface

State Machine



Message Pipeline