

THE UNIVERSAL TRADE WINDOW

Customer application / technical design

IMPLEMENTATION SPECIFICATION / v0.1 / 11 OCTOBER 2026

A device-agnostic commerce prototype with a foldable-first interaction model. This document separates the working demo from a proposed production architecture.

What ships in this kit

A self-contained HTML application, a TypeScript transaction core, original SVG illustrations and a zero-runtime-dependency Node server. The customer and employee clients share the same model and command vocabulary. The local server synchronizes clients through server-sent events (SSE).

There are three customer surfaces: the trade desk, the wanted board with a simulated second buyer, and Asset Passports. The desk supports adaptive, Duo/open, closed-phone and tabletop presentation modes. These are layout previews, not readings from a hinge sensor.

The recommended portfolio demo is one vertical slice: fund a used PS5 Pro, match the Jordans to an existing wanted ad, collect inspection evidence, pass specialist review, settle once, and produce an honest value receipt. The advisor's alternative keeps the Charizard and avoids a credit-card charge.

What does not ship

There is no live payment processing, crypto conversion, marketplace search, real authentication, external account linking, production identity system, AI inference or NFT minting. The optional SwiftUI/WKWebView shell is source for hosting the demo; it is not a native SwiftUI implementation and has not been compiled or simulator-tested in this environment.

Architecture at a glance

Layer	Working implementation	Production direction
Presentation	HTML/CSS and JavaScript; responsive panels	SwiftUI on Apple; equivalent web/other clients
Domain	Typed state and deterministic command reducer	Same behavioral contract; server authority

Synchronization	Local HTTP commands and SSE snapshots	Authenticated APIs and resumable event stream
Storage	In-memory server; local browser fallback	Durable transaction store and accounting ledger
Integration	Seeded fixtures	Partner-approved quote, payment, vault and custody adapters

Design rule: a pleasant interface can simplify the transaction without hiding its uncertainty.

01 / Domain model and money

One contribution contract; no imaginary universal cash balance.

TradeCore.State stores the selected asset IDs, cash limit, Pro eligibility, quote expiry, offer revision, client-visible version, approvals, inspections, phase, settlement and event history. TradeCore.apply(state, command, now) returns a new state or rejects an invalid transition. The injectable time supports deterministic expiry tests.

All amounts are integer USD cents. Formatting happens at the UI boundary. The demo's fixed tax calculation is illustrative; production needs a jurisdiction- and transaction-aware tax service rather than reusing the fixture.

Proposed production entities

Entity	Required responsibility
Asset	Stable identity; owner claim; category; location; restrictions
Quote	Gross and net amount; item and condition; costs; expiry; provider; certainty
Contribution	Authorized asset or payment source; amount; reservation and settlement route
Wanted intent	Product/variant constraints; condition; price cap; fulfillment preferences
Trade session	Versioned basket and independent legs; approvals; reservation references
Attestation	Procedure version; evidence references; authorized reviewer and decision

Passport	References to item, custody, attestations and rights; disclosure policy
Settlement	Immutable accounting and fulfillment instructions; subsequent adjustments

Numbers that must not be merged

Estimated asset value may be conditional. Eligible value has passed the required route checks. Settled value has actually been posted to the relevant transaction. Available credit is a payment capability, not customer-owned wealth.

In the demo, the initial cash/controller/Jordans basket estimates \$578.00. Only \$300.00 is immediately eligible. Buck adds a \$50.00 eligible GameStop gift card to cover the \$628.00 order without selling the Charizard. The \$245.00 shoe estimate does not become eligible until specialist approval; the \$33.00 controller contribution needs its store check.

An eBay gift card can enter the trade window but has no eligible route in the fixture. Ryan Cohen's graded underwear has a fictional \$69.69 quote and \$0.00 eligible value. Both must be removed before settlement. This proves that "anything can be considered" does not mean "everything is accepted."

The credit-card route charges only the remaining gap, capped by a demo limit. Cash usage is bounded by the customer's limit. Excess noncash contribution is identified as remaining credit, not silently forfeited or labeled as cash savings.

02 / The secure-trade state machine

Accept a version, not a moving target.

The working session has three top-level phases: building, settled, and collected. Within building, a proposal may exist, parties may have accepted its revision, physical goods may be pending/review/verified/failed, and quotes may be valid or expired.

version changes on every accepted command. offerRevision changes when the economic agreement changes. This distinction allows new inspection evidence without gratuitously resetting an unchanged offer, while preventing approval from surviving a change to the deal.

Required invariants

Condition	Enforcement
-----------	-------------

Contribution, cash limit, rewards or quote changes	Increment offer revision; clear both approvals and proposal
Discrepancy or reopened inspection	Block affected value; reset offer approvals
Insufficient authorized funding	Reject customer acceptance and settlement
Sneaker leg selected	Require Player B to accept the current revision
Physical route incomplete	Block settlement until the required review passes
Crypto quote expired	Block settlement; refreshed terms require acceptance again
Unsupported route still selected	Show explicit blocker; do not silently drop the item
Trade already settled	Reject edits and a second settlement

Proposal and conditional acceptance are not custody transfers or payment captures. The demo can accept an economically sufficient but inspection-pending offer, then wait for evidence. The final "complete" action re-evaluates all blockers using current time and state.

Real-world settlement is not a database swap

The production orchestrator would reserve inventory, authorize or reserve each permitted funding source, validate required attestations, recheck expiry and approvals, and commit coordinated settlement. External payment and logistics operations cannot be assumed to be instant or atomic.

Use a durable saga: explicit steps, idempotent provider requests, reconciliation, retry rules and compensating actions. If one leg fails after another commits, release unused reservations and route any committed obligation through a defined recovery policy. Never claim that a database rollback automatically reverses a shipped item, blockchain transfer or captured card payment.

A return or voluntary buyback creates a new post-sale process. It must not rewrite the original value receipt. The demo records the request and return intake, but does not implement a real refund, buyback payment or re-listing engine.

03 / API and synchronization contract

Thin clients; one authoritative writer in server mode.

Run `npm start` in app. The server binds to `127.0.0.1:4173` by default. `/trade-anything.html` and `/employee.html` then share one in-memory session. Opening two unrelated file copies does not

provide reliable cross-device synchronization.

Endpoint	Purpose	Important behavior
GET /api/state	Current complete snapshot	Demo has one session; no authentication
GET /api/events	SSE snapshots and heartbeat	Reconnection obtains current state; no durable replay log
POST /api/commands	Apply a typed command	Requires expected version and idempotency key

A command envelope contains `expectedVersion`, `commandId` (the idempotency key), and `command`. The command includes a type, demo role and relevant bounded parameters. Example: a customer TOGGLE for asset gift, or an employee EVIDENCE command identifying an asset, step, reference, mode and note. See the server and integration test for the exact executable contract.

Concurrent edits with an outdated expected version receive HTTP 409. Repeating the same retained idempotency key and payload returns current state without replaying the mutation. Reusing a key with a different payload is rejected. The bounded in-memory cache and event history are demo conveniences, not durable production idempotency guarantees.

The UI subscribes to snapshots and rerenders the relevant surface. The reducer is shared by browser fallback and Node, preventing two independent implementations of the basic accounting rules. A production offline client may draft edits, but the server must validate them before they affect financial state.

Local and failure behavior

The HTML bundles its scripts, styles and illustrations and can run without external network resources. When the local API is unavailable, it uses browser-local storage where supported, otherwise memory. The visible mode indicator distinguishes local/server behavior. Restarting the Node server resets the session.

Production service ports

Define narrow adapters for quote discovery, balance eligibility, vault entitlement, identity/permissions, payment authorization, specialist attestations, custody, tax, fulfillment and notifications. Implement mock and real adapters behind the same contracts.

Marketplace sold-comparable access, vault transfers and rewards entitlements require appropriate partner access and data rights. This kit does not assume a public API supplies every sold listing or authorizes cross-brand gift-card redemption. An AI estimate is advisory evidence, not a committed buyer bid.

04 / Foldable-first, not foldable-only

The hardware changes the composition, not the rules.

Apple's developer guidance provides an official entry point for iPhone Duo layouts and capabilities. The implementation should follow available public SDK contracts, not guessed hinge-angle APIs. The included web prototype deliberately uses viewport and explicit preview modes. [S9]

Presentation states

Available space / mode	Customer experience	Must remain invariant
Compact phone	Inventory, goal and funding summary stack; sticky trade window access	Every fee and blocker remains reachable
Open / wide	Inventory beside target and settlement panel	Same contributions, approvals and receipt
Tabletop preview	Review/goal area and controls composed for a supported surface	No hidden posture requirement
Desktop or tablet	Wide trading workspace with role/navigation links	All operations available by keyboard and click

Dropping an inventory card into the trade window and tapping its add control invoke the same command. Dragging is an enhancement, never the only way to fund a trade. Modes must not create a second transaction or clear approvals merely because a viewport changed.

In a future native app, use SwiftUI's adaptive composition, explicit navigation state and platform-supported scene handling. Keep domain state outside views, model asynchronous work with structured concurrency, and put credentials in platform-protected storage. NavigationSplitView is a relevant container to evaluate, not a substitute for designing the transaction itself. [S12]

Accessibility and interaction

The prototype includes labeled controls, focus-visible styling, native dialog semantics, reduced-motion handling and tap alternatives to dragging. The 390-pixel layouts were checked for horizontal overflow. This is not a completed WCAG or VoiceOver certification.

Before a native pilot: audit Dynamic Type and large accessibility sizes, VoiceOver reading order, switch/keyboard access, contrast, non-color status labels, touch targets, focus

restoration, error announcements and motion preferences. Ensure the receipt can be reviewed without an animation, sound effect or haptic cue.

Buck is not a financial pressure mechanic

The advisor shows a deterministic alternative route. It is not an LLM recommendation or a mathematical global optimum. Its claim is specific: the displayed fixture can cover the order while retaining named assets. No countdown except a real quote-expiry concept; no "free" reward that hides exchanged property or borrowing.

05 / Security, delivery and acceptance

A prototype that demonstrates controls without claiming production safety.

Roles in command payloads are demo labels, not authenticated identities. Anyone with access to the local server can attempt role commands or reset the session. The server has no TLS, persistent account system, signing keys, encryption-at-rest policy or financial integration. Keep it on loopback or a trusted test network and never enter real payment credentials, private keys or identity documents.

Local photo selection supports bounded image types/sizes and a temporary preview. The current app persists only an evidence reference and note, not an uploaded or verified image. The browser can lose the preview on reload. A production evidence service needs authenticated upload authorization, malware/content validation, access controls, retention/deletion policy, integrity checks and secure object storage.

Production controls before a pilot

Server-derived actor identity and scoped permissions; session isolation; strong account recovery; approved payment providers; quote reservations; duplicate-spend prevention; durable idempotency; reconciled accounting; signed specialist attestations; privacy-preserving audit access; abuse detection; incident recovery and operational support. Client UI state must never be the authority for ownership, credit eligibility or authentication.

Do not put identity records or sensitive inspection photographs on a public chain. Any NFT integration needs defined ownership/redemption semantics, transfer restrictions while disputed, key recovery and a physical-asset re-entry process. A portable hash cannot correct a false inspection.

Verified in this delivery

The Node test suite passes 28 tests: 27 domain tests plus one HTTP/SSE integration test. Nine browser scenarios exercise the assembled interface, including the advisor, evidence gates, specialist review, settlement, return request, discrepancy, quote expiry and responsive layouts. The browser harness injects the bundled page into Chromium because this environment restricts direct file/local navigation. Backend transport was tested separately.

No Safari, physical iPhone Duo, native build, real camera, payment-provider sandbox, screen reader or production load test was performed. Browser checks are useful interaction validation, not evidence of those unperformed tests.

Delivery gates

Prototype: reproducible fixture; safe failure; readable receipt; inspectable source. Native spike: compile the chosen SDK, test actual layout transitions and establish a pure-Swift domain contract. Partner sandbox: exercise real permissions and payment/custody failures without customer assets. Limited pilot: approved categories, trained operators, support/claims process, measured loss and discrepancy rates.

References: S9 Apple iPhone Duo developer resources; S11 WKWebView; S12 SwiftUI NavigationSplitView. The complete primary-source register and research boundaries are in `research/sources.md`.