Kindred Table — Supper Club Marketplace

Preliminary Requirements and System Specification

Document Type: Pre-Pipeline Functional Specification (M0 Deliverable / Sprint 0 output) **Version:** 0.2 (Post-M0-audit revision) **Changelog:** v0.1 initial intake synthesis → M0 Completeness Audit (score 88, findings issued) → **v0.2**: six audit accepts incorporated (host approval criteria; zone definitions; auth model resolved; promo schema constraints; concrete performance/concurrency thresholds; reference stack declared); four production-grade findings recorded as DEMO-WAIVED in §10 (food-safety depth, sales-tax fallback, DR/retention, dispute process) — waived for the demonstration build, required before any production launch. Audit report published alongside this document. **Date:** 2026-07-06 **Prepared For:** AI-Native Software Delivery Pipeline (VH M0–M7) — Epic/Feature Decomposition **Classification:** Published Demonstration Artifact

> **Demonstration notice.** Kindred Table and its founder are fictional; the scenario is a composite. The requirements, specification structure, provenance discipline, and all build artifacts generated from this document are real and independently verifiable. No fictional entity herein is intended to resemble any actual company.

Provenance legend (per-requirement tags):

- **[E]** Extracted — derived from artifacts (waitlist responses, host interest form, benchmark screenshots)
- **[S]** SME-stated — stated by the founder (the sole SME; interviews 1–2 / Sprint 0 session)
- **[I]** Inferred — synthesized by the audit; **flagged**. *Note: a pre-launch, single-founder venture carries a higher [I] density than an operating company — the volumes in §1.3 are targets, not history, and are flagged accordingly.*

---

1. System Overview

1.1 What Is Kindred Table?

Kindred Table is a two-sided marketplace for hosted supper clubs in Austin: hosts publish ticketed dinners in their homes or venues; guests discover, book seats, and pay; the platform confirms bookings, moves money via its payment seam, and builds trust through post-event

reviews. **[S — Founder, Sprint 0]** The founder is solo and non-technical; the platform must be operable by one person end-to-end (§7 admin console) and launch web-first in one city.

1.2 Problem Statement

[S — Founder] Supper-club hosts in Austin currently sell seats through Instagram DMs and payment apps: no discoverability, manual seat tracking, awkward money collection, no cancellation structure, and no accumulated reputation. Guests have no way to find dinners or trust unknown hosts. **[E — 240-response waitlist; 31 host interest-form responses]** Demand signals exist on both sides; the missing piece is the transactional venue between them.

1.3 Operating Environment (targets, pre-launch)

- **Launch market:** Austin, TX only (v0.1). **[S]**
- **Six-month targets (founder assumptions — [I], flagged as plan not history):** ~30 active hosts; ~150 published dinners; average 8 seats/dinner at ~$65/seat; platform fee 12% of the seat price. All four numbers are the founder's plan and are configuration, not constants.
- **Actors:** hosts, guests, the founder (admin), and the payment/notification providers behind seams.
- **Devices:** responsive web (mobile-primary); no native apps v0.1 (§8).

1.4 Key Technical Approach

Four-system decomposition (§3): pure-software core (Host Experience, Guest Booking, Marketplace Core) with money movement and messaging isolated behind contracted seams (System X). **The platform is not a payments company**: it never holds funds — Stripe Connect moves money directly with a platform application fee (§8's load-bearing exclusion) — and it is not a communications provider (notifications ride a provider seam).

2. Stakeholders and System Roles

Role	Person (fictional)	Interest	Provenance
Founder / Admin	L. Serrano	One-person operability; host quality; clean money flow; milestone visibility	Sprint 0 + Interviews 1–2 **[S]**
Hosts (~31 waitlisted)	—	Easy listing, seat management, reliable payout, protection from no-shows	Host interest form **[E]**

Role	Person (fictional)	Interest	Provenance
Guests (~240 waitlisted)	—	Discovery, trustworthy booking, clear cancellation terms	Waitlist responses **[E]**

3. Architecture & Decomposition Model

System H — Host Experience: host onboarding, dinner listings, seat/date management, host dashboard. **System G — Guest Experience:** discovery/browse, booking flow, guest account, reviews. **System B — Marketplace Core:** seat inventory, booking lifecycle, pricing/fee computation, cancellation policy engine, payout ledger view. **System X — External Seams:** X1 Stripe Connect (payments, payouts, refunds); X2 notification provider (email/SMS). **System P — Platform:** founder admin console; waitlist/landing; **commercial telemetry**; audit.

Reference stack (v0.2, per M0 finding): TypeScript/Node service, Postgres, React responsive web client, Stripe Connect **Express** onboarding (OQ-2 resolved), provider-adapter notifications (X2), single region. Demonstration-scale deployment; no multi-region or HA topology implied (see OQ-10).

Constraint C-1 — Monetary representation (hard constraint)

All monetary amounts are integer cents (`cents: i64`); floats prohibited in schemas, interfaces, persistence. Fee computation rounds half-even once, at fee calculation; `seat_price_cents × qty = booking_subtotal_cents` and `platform_fee_cents + host_net_cents = booking_total_cents` must hold exactly (AT-1). *Pipeline note: enforced by deterministic validation.*

Constraint C-2 — The platform never holds funds

All charges are Stripe Connect destination charges to the host's connected account with an application fee to the platform; refunds execute through Stripe against the original charge. No platform-side balance, wallet, or stored value exists (money-transmission avoidance; §8). **[S — Founder, on counsel's preliminary advice]**

Constraint C-3 — One-person operability

Every operational workflow (host approval, dispute note, refund exception, listing takedown) is executable from the admin console by a single non-technical operator; no workflow requires database or code access. **[S — Founder]**

4. System H — Host Experience

H1 — Host Onboarding

- REQ-H1.1 **[S]** The system shall register hosts via application (profile, venue description, sample menu, photos) entering `APPLIED` status; hosts become `ACTIVE` only on founder approval (C-3 console). **Approval checklist (v0.2, per M0 finding):** complete profile; ≥3 venue and food photos; sample menu; Stripe onboarding initiated; founder phone/video screen completed; no unresolved safety flags. Checklist is console-editable configuration.
- REQ-H1.2 The system shall require completion of Stripe Connect onboarding (X1) before a host may publish; publishing attempts without a payable account are rejected (`ERR_PAYOUT_NOT_READY`).
- REQ-H1.3 **[E — host form: 9 of 31 asked about co-hosts]** The system shall support one co-host per dinner (display + access), with payout to the primary host only in v0.1 (split payouts: OQ-5).

H2 — Dinner Listings

- REQ-H2.1 **[S]** The system shall create dinner listings: title, narrative, menu, dietary tags (enum: `VEGETARIAN, VEGAN, GF, DAIRY_FREE, NUT_FREE, NONE` — full coverage in filters), date/time, address (revealed per REQ-G2.4), seat count (2–20), seat price.
- REQ-H2.2 Listing lifecycle: `DRAFT → PUBLISHED → SOLD_OUT ⇄ PUBLISHED (on cancellation freeing seats) → COMPLETED → CLOSED`; `DRAFT|PUBLISHED → CANCELLED_BY_HOST`. Illegal transitions rejected (`ERR_ILLEGAL_TRANSITION`).
- REQ-H2.3 **[S]** Host cancellation of a published dinner shall trigger full refunds to all bookings (X1), guest notifications (X2), and a host-reliability mark visible to the founder (repeat cancellations flag for review).
- REQ-H2.4 The host dashboard shall show per-dinner seat sales, guest list (first name + dietary tags), expected payout in cents, and review summary.

5. System G — Guest Experience

G1 — Discovery

- REQ-G1.1 **[S]** The system shall present published dinners with search and filters (date range, price range, dietary tags, neighborhood zone — **launch zone list (v0.2, config):** Downtown, East Austin, South Congress, Hyde Park, Mueller, Zilker/Barton Hills, North Loop, Other; console-editable); sold-out dinners remain visible marked SOLD_OUT.
- REQ-G1.2 **[I — flagged; founder undecided]** Default ordering: soonest-first; ranking beyond that is out of v0.1 (OQ-6).

G2 — Booking

- REQ-G2.1 The system shall book 1–4 seats per booking against live seat inventory with overbooking prevented under concurrency (AT-4: no oversell under concurrent purchase of the last seat).
- REQ-G2.2 Booking lifecycle: `PENDING_PAYMENT → CONFIRMED → ATTENDED | NO_SHOW`; `CONFIRMED → CANCELLED_BY_GUEST | CANCELLED_BY_HOST → REFUNDED (per policy)`. `PENDING_PAYMENT` expires in 10 minutes, releasing seats.
- REQ-G2.3 **[S — Founder, benchmarked against comparable platforms; E — screenshots]** The cancellation policy engine shall compute refunds by rule: >72h before start = 100%; 24–72h = 50%; <24h = 0%; host cancellation = 100% always. Policy is versioned configuration; the booking stores its policy version (historic bookings refund under their original terms).
- REQ-G2.4 **[S — host safety concern, Interviews 1–2]** The exact address shall be revealed only to CONFIRMED guests, 24 hours before start (neighborhood zone shown publicly).
- REQ-G2.5 Confirmations, reminders (T-48h, T-4h with address), and cancellation notices ride X2.

G3 — Reviews

- REQ-G3.1 **[S]** Post-COMPLETED, guests may review the dinner (1–5 + text) and hosts may review guests (attendance/conduct flags); reviews publish double-blind (visible when both submit or at T+7 days).
- REQ-G3.2 Host aggregate rating displays on listings after ≥3 reviews; founder console can suppress a review pending dispute (C-3), with audit trail.

6. System B — Marketplace Core

- REQ-B1.1 The system shall compute per-booking money as: `subtotal = seat_price_cents × qty`; `platform_fee = round_half_even(subtotal × fee_rate)`; `host_net = subtotal - platform_fee` — fee rate is configuration (launch 12% **[S]**), stored per-booking at booking time (fee changes never restate history).
- REQ-B1.2 The system shall maintain a derived payout ledger view per host (booking-level: gross, fee, net, Stripe transfer reference, status) reconciling exactly to X1 events (AT-1/AT-8); the ledger is a view of Stripe truth, never an independent balance (C-2).
- REQ-B1.3 Refund execution shall follow the policy engine's computation through X1, updating booking and ledger states idempotently by Stripe event id.

- REQ-B1.4 **[S]** Promo codes: flat or percentage discounts, founder-created, usage-capped; discount reduces subtotal before fee computation; full accounting in the ledger view. **Schema constraints (v0.2):** code 4–16 alphanumeric, unique; percentage 5–50% or flat $5–$50 (cents-exact); single use per guest; total usage cap 1–500; expiry date required; no stacking (one code per booking).

7. System X — External Seams

Contracted stubs: typed payloads, versioned APIs, idempotent event handling; counterparty behavior specified at the boundary with provenance.

X1 — Stripe Connect

- REQ-X1.1 The system shall implement Connect Standard/Express onboarding (choice: OQ-2), destination charges with application fees, refunds against original charges, and webhook consumption (`payment_intent.succeeded`, `charge.refunded`, `payout.paid`, `account.updated`) — all handlers idempotent by event id (C-2 discipline).
- REQ-X1.2 **[I — flagged]** Stripe handles tax documentation to hosts (1099-K thresholds) and card compliance (PCI via Stripe elements — card data never touches the platform); assumption to be confirmed at counsel-lite review.
- REQ-X1.3 Webhook outage tolerance: the system shall reconcile against Stripe's API on a schedule such that missed webhooks converge within 1 hour (AT-9).

X2 — Notification Provider

- REQ-X2.1 Email/SMS via a provider adapter (selection: OQ-3) with templated messages (confirmation, reminders, cancellations, review prompts), delivery-state tracking, and guest SMS opt-in/out compliance.

8. System P — Platform

P1 — Founder Admin Console (C-3)

- REQ-P1.1 The console shall provide: host application review/approval; listing takedown; booking lookup with refund-exception execution (reason-coded: `enum { HOST_FAULT, GUEST_HARDSHIP, SAFETY, ERROR, OTHER }`, full coverage, OTHER requires note); review suppression; promo management; policy/fee configuration (versioned).

P2 — Commercial Telemetry

- REQ-P2.1 **[S — Founder: "I want to see the milestones happen"]** The system shall emit first-class business milestone events — `FIRST_HOST_ACTIVATED`, `FIRST_DINNER_PUBLISHED`, `FIRST_BOOKING_CONFIRMED`,

`FIRST_REVENUE_PROCESSED` (first `payment_intent.succeeded` with nonzero platform fee), plus running GMV, fee revenue, bookings, and repeat-guest rate — to the founder dashboard and to an exportable event log with timestamps and source references.

- REQ-P2.2 Milestone events are immutable facts derived from X1/booking records (never manually creatable) and are independently verifiable against Stripe records. *Note: where a build of this platform is financed under a lease with revenue-triggered terms, `FIRST_REVENUE_PROCESSED` is the machine-observable commencement event — the same telemetry serving product analytics serves payment-trigger observation, by construction rather than by audit.*

P3 — Waitlist & Launch Page

- REQ-P3.1 The public landing shall present the concept, capture waitlist signups (both sides), and convert to live discovery at launch toggle (C-3 console).

P4 — Audit

- REQ-P4.1 All founder-console actions, policy/fee changes, refund exceptions, and suppressions are recorded append-only with actor and timestamp.

9. Constraints and Exclusions

9.1 Scope Boundary (published coverage map)

In scope: the two-sided marketplace slice of §§4–8, Austin, web. **Explicitly out of scope, v0.1: money transmission** — the platform never holds funds; Stripe Connect is the money mover and compliance surface (C-2, the load-bearing exclusion); native mobile apps; host payout tax documentation (Stripe's 1099-K per REQ-X1.2); recurring/subscription dinners and series passes (OQ-4); whole-dinner private buyouts (**OQ-7 — deferred**); split co-host payouts (OQ-5); ranking/recommendation beyond soonest-first (OQ-6); content moderation at scale (founder-manual v0.1 per C-3); multi-city (config-ready, launch-locked to Austin); food-safety/permitting compliance (host-attested at onboarding; the platform is a venue, not a regulator — disclosure text at booking **[l — flagged for counsel-lite]**).

9.2 Technical Constraints

C-1 integer-cents with exact fee identities; C-2 no platform-held funds, Stripe as truth, idempotent event handling; C-3 one-person operability; policy versioning with historic-terms preservation (REQ-G2.3, B1.1); milestone-event immutability (P2.2); append-only audit.

10. Open Questions and Assumptions

#	Status	Question / Ambiguity	Assumption / Resolution path
OQ-1	**RESOLVED (v0.2)**	Guest identity: account-required vs. guest checkout	Lightweight account: email + SMS verification, passwordless (magic link / OTP); 30-day rolling sessions. Chosen per M0 finding to unblock decomposition
OQ-2	**RESOLVED (v0.2)**	Stripe Connect Standard vs. Express onboarding for hosts	**Express** chosen (fastest host onboarding; adequate for demonstration scope)
OQ-3	OPEN	Notification provider selection	Adapter-confined; selection does not block build
OQ-4	OPEN	Recurring dinner series / season passes (4 hosts asked **[E]**)	Out of v0.1; listing model admits `series_id` without rework
OQ-5	OPEN	Co-host split payouts	Out of v0.1 (REQ-H1.3); Stripe multi-destination evaluated later
OQ-6	OPEN	Discovery ranking beyond soonest-first	Deferred; requires data that doesn't exist yet **[S — Founder, correctly]**
OQ-7	**DEFERRED**	**Whole-dinner private buyouts.** Two waitlisted hosts	Revisit post-launch as a scope-change candidate; the

#	Status	Question / Ambiguity	Assumption / Resolution path
		requested corporate/private full-table bookings at a premium. **[E — host interest form]** Explicitly deferred from v0.1 to protect launch scope.	booking model (qty = full seat count, single payer) and policy engine were designed to admit it without structural rework
OQ-8	**DEMO-WAIVED (v0.2)**	Texas sales tax on ticketed dining experiences	**[I — genuinely uncertain]** Waived for the demonstration build (no real transactions). REQUIRED before any production launch: counsel-lite review; likely Stripe Tax integration on the X1 seam
OQ-9	**DEMO-WAIVED (v0.2)**	Food-safety safeguards beyond host attestation + disclosure text (per M0 finding)	Waived for demonstration. Production launch requires: host food-handler attestation flow, incident reporting path, and counsel-reviewed disclosure language
OQ-10	**DEMO-WAIVED (v0.2)**	Disaster recovery, backup, and data-retention policy (per M0 finding)	Waived for demonstration (single-region reference deployment). Production requires: backup/restore RPO-RTO targets,

#	Status	Question / Ambiguity	Assumption / Resolution path
			retention schedule, deletion policy
OQ-11	**DEMO-WAIVED (v0.2)**	Guest–host dispute process depth (per M0 finding)	Waived for demonstration; refund-exception reason codes (P1.1) are the v0.2 surface. Production requires: dispute workflow, evidence capture, and resolution SLAs

11. Acceptance & Performance

11.1 Business Acceptance Tests (selected)

- **AT-1 (property):** for every booking over the generated corpus: `subtotal = price × qty`, `fee + host_net = subtotal` (post-promo), exactly, in integer cents; ledger view reconciles to Stripe event totals.
- **AT-2:** cancellation refunds compute per the booking's stored policy version, including after a policy change.
- **AT-4 (concurrency):** simultaneous purchase attempts on the final seat produce exactly one CONFIRMED booking; no oversell across 1,000 trials.
- **AT-5:** address revelation occurs only to CONFIRMED guests at T-24h; no earlier path exists.
- **AT-7:** all lifecycle FSMs reject illegal transitions; enum consumers demonstrate full value coverage (dietary tags, refund reasons, booking states).
- **AT-9:** webhook replay produces zero duplicate state changes; a simulated 45-minute webhook outage converges via reconciliation within the hour.
- **AT-11:** every founder-console operational workflow completes without code/database access (C-3 walkthrough).
- **AT-13:** `FIRST_REVENUE_PROCESSED` emits exactly once, on the first nonzero-fee successful payment, is immutable, and matches the Stripe record (the milestone-integrity test).

11.2 Performance Targets

Booking flow ≤ 1.5 s p95 server-side per step at 50 concurrent users; discovery page ≤ 2 s p95 at 500 published listings; AT-4 concurrency gate: 200 simultaneous purchase attempts on a final

seat, 1,000 trials, zero oversell; webhook processing ≤ 5 s p95 with 1-hour reconciliation convergence (X1.3); admin console actions ≤ 2 s p95. Demonstration scale: 100 concurrent users sustained. **(v0.2: thresholds concretized per M0 finding; [I] — targets, validated at build)**

Appendix A (provenance source register): waitlist responses (240); host interest form (31 responses); benchmark screenshots (3 comparable platforms); founder Sprint 0 session + interviews 1–2. Appendix B (acceptance traceability) and the M7-generated build package are published alongside this document on the case-study page, together with the lease term sheet under which builds of this class are financed.