soleX Round 2 — Prototype Note

A working, frontend-only prototype of the soleX operating system. It follows one transaction from a channel partner's enquiry to the developer's report, with real business rules, role-scoped data and a live audit trail. This note answers the engineering questions in section 7 of the brief and states plainly what is simulated.

Prepared for **Team soleX** · ShahX Buildcon Pvt. Ltd. By **Equitysoft Technologies**, Ahmedabad **November 2026** All data fictional

01 What we built & how to explore it

A cinematic landing page (a 3D night skyline where every lit window is a real unit, a plain-language "What is soleX?" section and five everyday mandate arguments the system ends) → login with one-click role cards → six role workspaces that read and write **one shared data set**. A booking made by Sales shows the same unit, amount and receipt reference in the Management, Finance, Broker, Developer and Customer views. A floating **Guided tour** walks the story below, and the in-app **How it works** page mirrors this note.

ROLE	DEMO PERSONA · LOGIN	LANDS ON	WHAT TO LOOK FOR
Management	Arjun Shah, Director · password	Command Centre	KPIs across 4 projects, approvals inbox, drill-down from a tile to the project, unit or booking
Sales	Riya Mehta, Sales Executive · password	Sales workspace	Pipeline, lead timeline, broker attribution, unit hold, booking wizard
Finance	Hetal Parikh, Finance Head · password	Collections	Record receipts into the developer's escrow; commission ledger
Channel partner	Mehul Patel, Skyline Realty · OTP	Broker portal	Lead registration with duplicate check and 60-day protection; own commission statement
Developer	Kunal Agarwal, Aaryan Infrabuild · password	Developer portal	Own projects only: sales, inventory, escrow collections, MIS; "Not shared with developers" panel
Customer	Nirav Desai or Kavya Iyer · OTP	Customer app	Booking, payment schedule, receipts and relationship manager in a phone frame

The one-transaction demo story

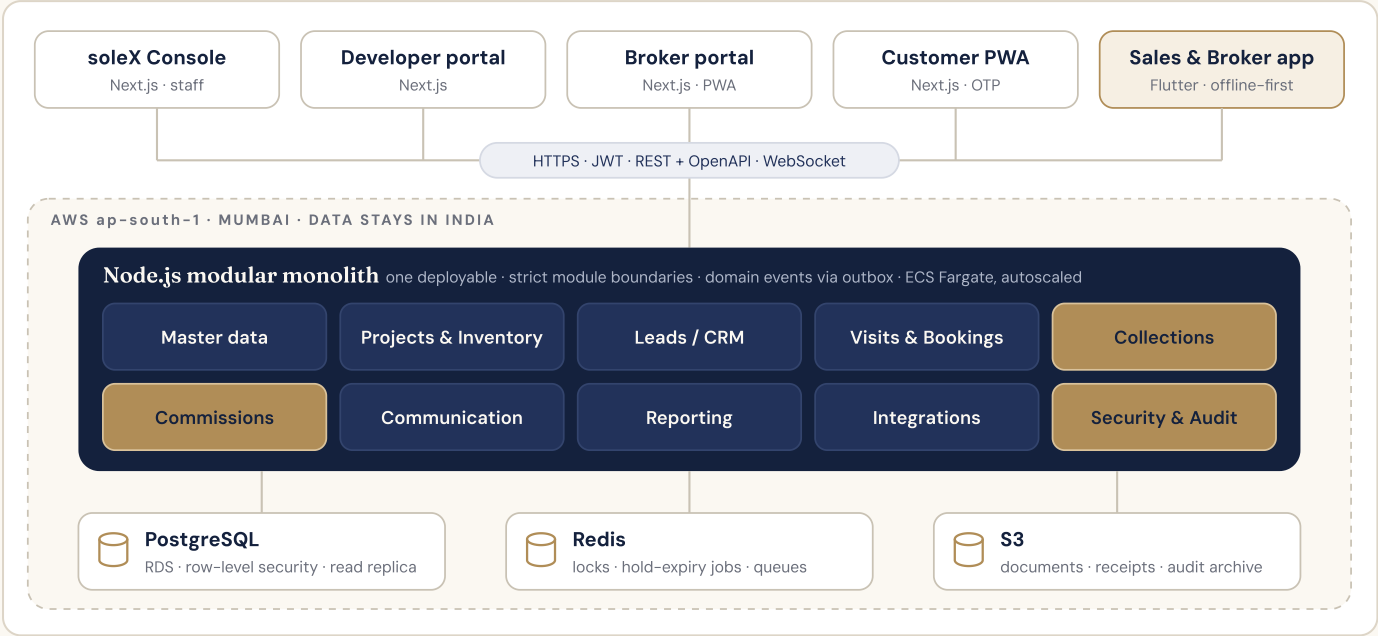
Nirav & Pooja Desai, introduced by Skyline Realty, buy corner 3 BHK **B-1204** at **Aaryan Aura, Shela**. Suggested order (about 15 minutes):

1. START **Landing page → Enter the demo.** Pick a role card; any 6-digit OTP works for partner and buyer logins.
2. PARTNER **Register the buyer.** Re-register 98250 12312 and the duplicate is blocked; the lead is protected to Skyline for 60 days.
3. SALES **Open the Desai lead.** Timeline, completed site visit, Skyline attribution and a rule-based next best action.
4. SALES **Inventory → Tower B → B-1204 → Hold (30 min).** Try to hold or book it in a second tab as another user: refused.
5. SALES **Convert to booking:** CLP plan with a 2.5% discount. Above 2% needs approval, so it waits as "pending approval".
6. MGMT **Approve from the inbox.** The unit becomes booked, the payment schedule and expected commission lines are created, and sold value updates.
7. FINANCE **Record the 10% booking amount** by NEFT with a UTR, credited to Aaryan Infrabuild's RERA escrow account.
8. FINANCE **Commissions:** 10% collected moves Skyline's payout, soleX's fee and Riya's incentive from expected to earned; approve and pay with TDS.
9. DEV · BUYER **Each sees its slice.** The developer sees the sale and fee payable; the buyer sees the same receipt, amount and reference.
10. ALL **How it works.** Switch roles and watch the visibility panel re-scope; the live audit log lists every action you just took. Optional: cancel a booking and watch the unit return to inventory and its commission lines claw back.

> **Press "Reset demo" (top bar) before each run.** It restores the seeded data set; demo data lives in your browser only. Use the **Switch role** menu to move between personas mid-story without logging out.

02 Architecture & stack

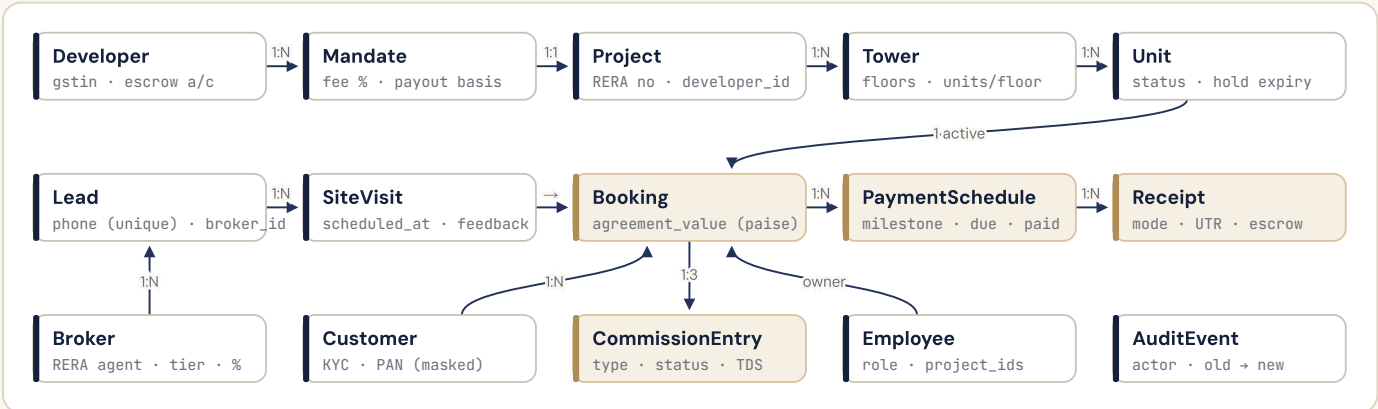
As in our proposal: a **Node.js modular monolith** behind role-based front-ends. One booking touches inventory, schedule, commissions and audit at once; in one service that is a single database transaction. Modules keep hard boundaries and talk through interfaces and outbox events, so any one can later become its own service without a rewrite.



LAYER	PRODUCTION CHOICE	WHY
Web	Next.js, React, TypeScript: four role-based web apps on one component library	Dashboards, grids and approvals need a large screen; one design system
Mobile	One Flutter app with Sales and Broker modes, offline-first	Brokers and site staff live on phones; one app halves store upkeep
API	Node.js (TypeScript, NestJS framework), REST + OpenAPI, WebSocket for live inventory and approvals	Documented contracts; real-time unit status across users
Data	PostgreSQL, Redis, S3 on AWS Mumbai (ECS Fargate), accounts in soleX's name	Transactions and row-level security; data stays in India; no lock-in
This prototype	Next.js 15 + TypeScript + Tailwind, Zustand store, Recharts, Motion; rules in `src/domain`	Same front-end stack as production; rules written once as pure functions

03 Core entities & relationships

The **mandate is the spine**: every rupee traces back to a unit and a mandate. Gold entities carry money, stored as integer paise.



Supply: Developer → Mandate (fee %, booking- or collection-linked payout) → Project → Tower → Unit. **Demand:** Broker → Lead → Site visit → Booking, attribution kept. **Money:** Booking → schedule → receipts, plus up to three commission lines (soleX fee, partner payout, staff incentive).

04 Keeping inventory, bookings & money consistent

RISK	CONTROL IN PRODUCTION	IN THIS PROTOTYPE
Same flat sold twice	Unit row locked (`SELECT … FOR UPDATE`) inside the hold and booking transaction, plus a unique "one active booking per unit" constraint	REAL Unit state machine and hold-owner check; a second tab is refused
Holds that never end	30-minute holds (up to 24 h with a Sales Manager–approved extension) expire through a Redis-scheduled job; a hold awaiting discount approval does not expire	REAL Same rule; expired holds read as available
Half-made bookings	Unit status, booking, payment schedule, commission lines and audit events written in **one transaction**: all or nothing	REAL One command updates the store in one step
Rounding drift	Amounts are BIGINT paise, never floats; the last milestone absorbs the remainder so instalments sum exactly	REAL Integer paise throughout
Duplicate receipts	UTR / cheque number unique; cheques count only once cleared; a bounce raises a critical alert	REAL Duplicate reference blocked
Wrong commission	Rule engine driven by mandate and partner terms; lines move expected → earned → approved → paid; TDS 2% (194H) on partner payouts; clawback on cancellation	REAL Same engine, unit-tested; cancel-booking flow releases the unit and claws back lines
Lost notifications	Side effects written to an outbox table in the same transaction and delivered by a worker	SIMULATED Toasts
Untraceable edits	Append-only audit (actor, time, old → new); the app role has no UPDATE / DELETE on it; archived to S3	REAL Live audit log

> **soleX tracks money; it never holds it.** Buyer payments go straight to the developer's RERA escrow account. soleX records each receipt, reconciles it against the schedule, and uses collection progress to decide when commissions become payable.

05 Confidential data separation

One platform, but no party sees another's commercial terms. Enforced **twice**, with sensitive fields redacted on top.

1 · CASL policies in the API

Every request is checked against the user's role and scope (projects, developer, partner) before any query runs.

```
cannot('read','CommissionEntry',
{type:'broker_payout'})
```

2 · Postgres row-level security

Policies on `developer_id`, `broker_id` and `project_id`: even a faulty query cannot return another party's rows.

```
USING (developer_id =
current_setting('app.developer_id'))
```

WHAT	MGMT & FINANCE	SALES	CHANNEL PARTNER	DEVELOPER	CUSTOMER
Projects & inventory	All	Allocated projects	Empanelled projects	Own projects	Own unit
Leads & buyer contact	All	Own leads	Own leads only	Counts; contact details redacted	—
soleX mandate fee	Visible	Hidden	Hidden	What they owe	Hidden
Partner payouts	Visible	Hidden	Own statement	Hidden	Hidden
Staff incentives	Visible	Own only	Hidden	Hidden	Hidden
soleX margin	Visible	Hidden	Hidden	Hidden	Hidden
Audit trail	All	Own actions	Hidden	Own units & bookings	Hidden

> **Try it:** as Kunal (developer) you see the B-1204 sale and the soleX fee payable, never Skyline's payout or soleX's margin. As Mehul (Skyline) you see only Skyline's leads and statement, never another partner's. In the prototype the matrix is one tested function, `scopeFor(role)`.

06 Room to grow

- **Configuration over code.** Mandates, fee %, payout basis, payment plans, incentive slabs and approval limits are data: a new developer needs no deployment.
- **Multi-project, multi-city.** Every record is keyed to project and developer, so scoping holds as the portfolio grows.
- **Capacity by configuration.** Autoscaled containers; a read replica serves MIS so reports never slow a live booking. ~1,500 users at launch, 5,000 without code changes.
- **Extract when needed.** A busy module (e.g. Communication, Reporting) moves to its own service behind the same contracts.
- **Integrations when justified.** WhatsApp Business API, SMS, email, bank-statement matching, HRMS and accounting connect through one Integrations module.
- **AI on real data.** Rules now; lead scoring and suggestions once 3+ months of live data exist. AI suggests with a reason and never decides money or ownership.

07 Simulated vs real

CAPABILITY	IN THIS PROTOTYPE	IN PRODUCTION
Business rules	REAL Hold, booking, schedule, receipts, commissions and scoping run as tested TypeScript	Same rules in Node.js services inside DB transactions
Audit trail	REAL Every action appends actor and old → new values	Append-only table, archived to S3
Data & backend	SIMULATED No backend; seeded data in browser storage, synced across tabs	PostgreSQL on AWS Mumbai, backups, point-in-time recovery
Login & OTP	SIMULATED One-click personas; any 6-digit OTP	Password + 2FA for staff, SMS OTP for partners and buyers
Messaging	SIMULATED WhatsApp, SMS and email buttons show toasts	WhatsApp Business API, SMS gateway, email service
Payments	SIMULATED Receipts recorded manually; no money moves; no gateway	Same, plus bank-statement matching; gateway optional
Documents	SIMULATED Upload previews; downloads and MIS exports are toasts	S3 with signed URLs; generated PDFs
AI insights	ILLUSTRATIVE Rule-based next best action, labelled "AI-ready"	Scoring model on CRM history, behind a feature flag
HRMS	PREVIEW Attendance and incentive snapshot from seed data	Integrated HRMS / payroll, in a later phase

Assumptions

- Buyer payments go to the developer's RERA escrow; soleX tracks collections and does not collect them.
- Illustrative rules, to confirm in discovery: Director approval above 2% discount; payouts and incentives earn at 10% collected; 30-min holds (24 h with approved extension); 60-day lead protection.
- One mandate per project; one active booking per unit.

Limitations

- Data lives in one browser; devices do not share state.
- Role scoping runs in the browser to show the rules; it is not a security boundary (production: API + database).
- No live messaging or payment integrations; no load or penetration testing.
- All companies, people, phone and RERA numbers are fictional.