Wallet Connect

- Secure login and onboarding flow using Freighter or other Stellar-compatible wallets, allowing users to connect their existing wallet, view connected public address, and confirm network details before proceeding to any purchase flow.
- Session management layer on the frontend using React Context and secure HTTP-only tokens so that once a wallet is connected, users can move between dashboard, plans, and referral tree without repeated reconnect prompts.
- Validation and error-handling screens for failed connections, unsupported networks, or missing extensions, with clear retry options and basic help text to guide non-technical users.
- Basic profile section displaying linked wallet address, last login time, and option to disconnect wallet, ensuring transparency and user control similar to account sections in the sample quotation.

Purchase & Lock

- Plan selection screen listing all active admin-created investment plans with key details such as minimum USDT amount, lock-in duration, expected token allocation, and plan status (active/closed).
- Guided purchase flow that connects to the Soroban smart contract, calculates token quantity from USDT based on the selected plan, and prepares a Stellar transaction for signing via the user's wallet.
- Automatic lock-in logic where the contract stores purchase timestamp, lock duration, and status flags so funds cannot be withdrawn or swapped back before the lock period ends.
- Post-purchase confirmation screen and history list showing each purchase, lock start and end date, token amount, and current status (locked, completed, cancelled).

Admin Plan Management (CRUD)

- Admin-only web panel to create new investment plans with configurable fields such as plan name, description, minimum/maximum USDT, lock-in days, rate or bonus, and visibility (active/inactive).
- Plan listing grid with search and filters, allowing admins to quickly view, edit, or deactivate existing plans, following the structured management style used for “Commission Slab Management” and “Insurance Order Management” in the sample quotation.
- Role-Based Access Control (RBAC) to restrict plan creation and updates to authorized admin roles while allowing view-only access for moderators or support staff, with all changes stored in MongoDB.
- Version-safe updates so that edits to a plan do not break existing active investments: when a plan is updated, new purchases use the latest parameters while older purchases retain original terms.

Sell & Withdraw

- Post-lock sell request flow where, once the lock period is completed, users can initiate a “Sell Token” action that calculates eligible token balance and corresponding USDT output based on current contract rules.
- Backend queue mechanism that signs and submits Stellar transactions asynchronously, ensuring user requests are recorded immediately while actual blockchain settlement is processed in the background for better UX.
- Withdrawal status tracking screen showing each sell/withdraw request with amount, requested time, processing state (queued, in progress, completed, failed), and transaction hash when available, mirroring the structured reporting style found in the sample quotation.
- Basic failure and retry handling where any failed transaction is logged, surfaced to the user with a descriptive message, and optionally retried by the admin through the backend tools.

Binary Referral Tree

- Binary-tree referral model where every user has at most two direct sub-users (left and right), with MongoDB storing parent-child references optimized for quick traversal up to at least three levels.
- Referral link generation so each user can share a unique URL; new signups through that link automatically attach to the correct parent node and fill the left or right leg based on business rules.
- Tree visualization page on the web dashboard showing a graphical left/right structure up to three levels, displaying key data per node such as user ID, join date, and cumulative volume, similar in clarity to the hierarchical views described for agents and customers in the sample quotation.
- API endpoints to fetch tree data, enabling future extensions like deeper-level views, performance analytics, and export of referral structures for reporting or audits.