Crypto Portfolio Dashboard — Mobile App

Claude Code Project Plan

Owner: Curtis **Purpose:** A unified mobile dashboard showing total crypto exposure across Crypto.com, Hyperliquid, Blur (NFT trading), Ethereum, and Solana — with 2025 tax export and manual entry support for events like the Blur NFT liquidation loss.

---

1. Tech Stack

Layer	Choice	Why
Framework	React Native (Expo)	Cross-platform iOS/Android, fast dev cycle, OTA updates
Navigation	React Navigation v6	Tab + stack nav, well-supported
State	Zustand	Lightweight, no boilerplate, persists easily
Storage	expo-secure-store (API keys), AsyncStorage (app data)	Secure storage for secrets, fast local storage for everything else
Charts	react-native-chart-kit or Victory Native	Portfolio visualizations
Excel export	xlsx (SheetJS) + expo-sharing	Generate .xlsx in-app, share via native sheet
Styling	NativeWind (Tailwind for RN) or StyleSheet	Consistent design system

---

2. App Architecture

```
src/
├── app/                          # Expo Router screens
```

```
│   ├── (tabs)/
│   │   ├── dashboard.tsx   # Main portfolio overview
│   │   ├── trades.tsx      # Transaction history + filters
│   │   ├── manual.tsx      # Custom entry form
│   │   └── settings.tsx    # API key management
│   └── _layout.tsx
├── services/
│   ├── crypto-com.ts       # Crypto.com API integration
│   ├── hyperliquid.ts      # Hyperliquid API integration
│   ├── blur.ts             # Blur API integration
│   ├── ethereum.ts         # Etherscan / Alchemy for ETH wallet
│   ├── solana.ts           # Helius / Solscan for SOL wallet
│   └── aggregator.ts       # Normalizes all sources into unified format
├── stores/
│   ├── portfolio.ts        # Zustand store — holdings, balances, prices
│   ├── trades.ts           # Zustand store — all transactions
│   ├── manual-entries.ts   # Zustand store — user-added custom entries
│   └── settings.ts         # Zustand store — API keys, preferences
├── utils/
│   ├── excel-export.ts     # Build .xlsx from trades + manual entries
│   ├── tax-helpers.ts      # Cost basis, gain/loss calc helpers
│   └── formatters.ts       # Currency, date, address formatting
├── components/
│   ├── PortfolioCard.tsx
│   ├── AssetRow.tsx
│   ├── TradeRow.tsx
│   ├── PlatformBadge.tsx
│   ├── ExposureChart.tsx
│   └── ManualEntryForm.tsx
└── types/
    └── index.ts            # Shared TypeScript types
```

---

3. Screen-by-Screen Spec

3.1 Dashboard (Main Screen)

This is the home screen. It answers one question: **“What’s my total crypto exposure right now?”**

Header section:

- Total portfolio value in USD (large, prominent)
- 24h change ($ and %)

- Last refreshed timestamp + pull-to-refresh

Exposure breakdown:

- Pie or donut chart showing allocation by platform (Crypto.com, Hyperliquid, Blur, ETH wallet, SOL wallet)
- Secondary breakdown by asset (BTC, ETH, SOL, NFTs, stablecoins, etc.)
- Tap any segment to drill into that platform's holdings

Asset list:

- Scrollable list below the chart
- Each row: asset icon, name, amount held, current value, 24h %, platform badge
- Sort by: value (default), name, 24h change
- Group by: platform or asset (toggle)

Data sources for prices:

- CoinGecko free API for spot prices (no key needed, 30 calls/min)
- Platform APIs for balances/positions

3.2 Trades Screen

Filters (top bar, horizontal scroll):

- Platform: All / Crypto.com / Hyperliquid / Blur / ETH / SOL
- Type: All / Buy / Sell / Swap / Transfer / Liquidation / Custom
- Date range: Quick picks (YTD, Q1, Q2, etc.) + custom date picker
- Search by asset name or tx hash

Trade list:

- Each row: date, type badge, asset, amount, price at time, total value, platform badge, gain/loss
- Manual entries show a distinct "Custom" badge so they're clearly flagged
- Tap to expand with full details (tx hash link, fees, notes)

Export button (top right):

- "Export 2025 Trades" — generates .xlsx filtered to Jan 1 – Dec 31, 2025

- Opens native share sheet (save to Files, email, AirDrop, etc.)

3.3 Manual Entry Screen

This exists because not everything shows up via API — especially the Blur NFT liquidation loss from early 2025.

Form fields:

- Date (date picker, defaults to today)
- Platform (dropdown: Crypto.com / Hyperliquid / Blur / Ethereum / Solana / Other)
- Type (dropdown: Buy / Sell / Swap / Transfer / Liquidation / Airdrop / Other)
- Asset name (text input with autocomplete from CoinGecko)
- Amount (number)
- Price per unit at time of transaction (USD)
- Total value (auto-calculated, or override manually)
- Fees (optional, USD)
- Cost basis (optional — for tax calc)
- Gain/Loss (optional — auto-calc if cost basis provided, or manual override)
- Notes (free text — e.g., “Blur NFT liquidation, collection X”)
- TX hash / reference (optional, for record keeping)

Actions:

- Save — adds to manual entries store, appears in trades list
- Edit — tap any saved manual entry to modify
- Delete — swipe to delete with confirmation

Pre-populated template for Curtis:

- On first launch or via a “Quick add liquidation” shortcut, pre-fill: Platform = Blur, Type = Liquidation, so it’s fast to log the NFT loss details

3.4 Settings Screen

API Keys section: Each platform gets its own card with:

- Platform name + logo

- API Key field (masked, tap to reveal)
- API Secret field (if applicable)
- "Save" button per platform
- Connection status indicator (green dot = connected, red = error, gray = not configured)
- "Test Connection" button — makes a lightweight API call to verify credentials

Platforms requiring keys:

Platform	Keys Needed	How to Get
Crypto.com	API Key + Secret	Crypto.com Exchange → Settings → API
Hyperliquid	Wallet address (no key needed for read)	Just the 0x address
Blur	Wallet address	0x address used on Blur
Ethereum	Wallet address + Etherscan API key (optional)	etherscan.io for API key
Solana	Wallet address + Helius API key (optional)	helius.dev for API key

Storage:

- All API keys stored via `expo-secure-store` (encrypted, never in plain AsyncStorage)
- Wallet addresses can be in AsyncStorage (not sensitive)

Other settings:

- Default currency (USD / CAD — Curtis is in Kelowna, BC so CAD option matters)
- Auto-refresh interval (5min / 15min / 30min / manual only)
- Export format preferences
- Clear all data / reset

---

4. API Integration Details

4.1 Crypto.com Exchange API

```
Base URL: https://api.crypto.com/exchange/v1
Auth: HMAC-SHA256 signed requests (API key + secret)
```

Endpoints needed:

- `private/get-account-summary` — balances per coin
- `private/get-order-history` — filled orders (trades)
- `private/get-deposit-list` / `get-withdrawal-list` — transfers

Rate limits: 3 requests/100ms per method. Batch where possible.

Normalization: Map their trade format to unified TradeRecord type.

4.2 Hyperliquid API

```
Base URL: https://api.hyperliquid.xyz/info
Auth: None for read — just POST with wallet address
```

Endpoints needed:

- `POST /info` with `{"type": "clearinghouseState", "user": "0x..."}` — open positions, margin, PnL
- `POST /info` with `{"type": "userFills", "user": "0x..."}` — trade history
- `POST /info` with `{"type": "spotClearinghouseState", "user": "0x..."}` — spot balances

Rate limits: Generous, no key needed. Just don't hammer it.

Note: Hyperliquid is perps-heavy, so positions include unrealized PnL. Dashboard should show both realized and unrealized.

4.3 Blur

```
Blur doesn't have an official public API with key-based auth.
```

Approach options (in priority order):

1. **Reservoir API** (reservoir.tools) — aggregates NFT marketplace data including Blur. Free tier available. Needs API key.

    - `GET /users/{address}/activity` — sales, purchases, listings
    - `GET /users/{address}/tokens` — current NFT holdings

2. **Blur's internal API** — undocumented, may break. Not recommended for production.
3. **Etherscan** — trace Blur contract interactions for the wallet address. Reliable but requires parsing contract calls.
4. **Manual entries as fallback** — for the liquidation and any trades the API misses.

Recommendation: Use Reservoir as primary, manual entries as safety net. Curtis's Blur liquidation may not surface cleanly via any API, so the manual entry feature is critical here.

4.4 Ethereum (Wallet)

```
Provider: Etherscan API or Alchemy
Auth: API key
```

Endpoints needed:

- ETH balance: `module=account&action=balance`
- ERC-20 token balances: `module=account&action=tokentx` (then aggregate)
- Transaction history: `module=account&action=txlist`
- Token transfers: `module=account&action=tokentx`

Price enrichment: Cross-reference token addresses with CoinGecko for current prices.

4.5 Solana (Wallet)

```
Provider: Helius API (helius.dev) or Solana RPC
Auth: API key for Helius, none for public RPC
```

Endpoints needed:

- `GET /v0/addresses/{address}/balances` — SOL + SPL token balances
- `GET /v0/addresses/{address}/transactions` — parsed transaction history
- Helius's enhanced API returns human-readable tx types (swap, transfer, NFT sale, etc.)

Alternative: Solscan API (solscan.io) for simpler queries.

---

5. Excel Export Spec

File: `crypto-trades-2025.xlsx`

Sheet 1: “All Trades”

Column	Description
Date	Transaction date (YYYY-MM-DD)
Platform	Crypto.com / Hyperliquid / Blur / Ethereum / Solana / Manual
Type	Buy / Sell / Swap / Transfer / Liquidation / Airdrop / Other
Asset	BTC, ETH, SOL, NFT collection name, etc.
Amount	Quantity
Price (USD)	Per-unit price at time of transaction
Total Value (USD)	Amount × Price
Fees (USD)	Transaction fees
Cost Basis (USD)	Original purchase cost (if available)
Gain/Loss (USD)	Proceeds minus cost basis
TX Hash	On-chain reference (if applicable)
Source	“API” or “Manual” — flags which entries were user-added
Notes	Free text (manual entry notes)

Sheet 2: “Summary”

- Total trades count
- Total volume
- Total realized gains
- Total realized losses
- **Net gain/loss for 2025**
- Breakdown by platform

- Breakdown by asset
- Flagged: Blur liquidation loss (since this is the big tax event)

Sheet 3: "Manual Entries Only"

- Same columns as Sheet 1, filtered to manual entries
- Makes it easy for an accountant to see what was self-reported vs. API-sourced

Export flow:

1. User taps "Export 2025 Trades" on Trades screen
2. App filters all trades (API + manual) to date range Jan 1 – Dec 31, 2025
3. SheetJS builds the workbook with 3 sheets
4. File saved to temp directory
5. expo-sharing opens native share sheet
6. User saves to Files, emails to accountant, etc.

---

6. Unified Data Types

```
// Every trade from every source gets normalized to this
interface TradeRecord {
  id: string;
  date: string;                  // ISO 8601
  platform: Platform;
  type: TradeType;
  asset: string;
  amount: number;
  priceUsd: number;
  totalValueUsd: number;
  feesUsd: number;
  costBasisUsd?: number;
  gainLossUsd?: number;
  txHash?: string;
  source: "api" | "manual";
  notes?: string;
  raw?: any;                     // Original API response for debugging
}

type Platform =
  | "crypto.com"
  | "hyperliquid"
  | "blur"
  | "ethereum"
  | "solana"
  | "other";

type TradeType =
  | "buy"
  | "sell"
  | "swap"
  | "transfer"
  | "liquidation"
  | "airdrop"
  | "other";

interface PortfolioHolding {
  asset: string;
  platform: Platform;
  amount: number;
  currentPriceUsd: number;
  currentValueUsd: number;
  change24hPercent: number;
  unrealizedPnlUsd?: number;    // For Hyperliquid perps
}

interface ManualEntry extends TradeRecord {
  source: "manual";
  createdAt: string;
  updatedAt: string;
}
```

---

7. Build Order for Claude Code

Work through these phases in order. Each phase should be a working app you can run.

Phase 1: Shell + Navigation + Settings

Goal: App runs, you can navigate tabs, save API keys.

- Initialize Expo project with TypeScript
- Set up React Navigation with 4 tabs (Dashboard, Trades, Manual Entry, Settings)
- Build Settings screen with secure key storage

- Placeholder screens for other tabs
- Basic dark theme (matches the crypto dashboard vibe)

Phase 2: Dashboard + Price Data

Goal: Dashboard shows portfolio value from a single source.

- Integrate CoinGecko for live prices
- Connect Ethereum wallet (simplest API, good first integration)
- Build PortfolioCard, AssetRow, ExposureChart components
- Pull-to-refresh + loading states

Phase 3: All Platform Integrations

Goal: All 5 data sources feeding the dashboard.

- Crypto.com authenticated API calls
- Hyperliquid read-only API
- Solana wallet via Helius
- Blur/NFT data via Reservoir
- Aggregator service that normalizes everything into unified types
- Error handling per platform (if one fails, others still show)

Phase 4: Trades Screen + Manual Entries

Goal: Full transaction history with filtering + ability to add custom entries.

- Trades list with platform/type/date filters
- Manual entry form with all fields
- CRUD for manual entries (create, edit, delete)
- “Custom” badge on manual entries in the trade list
- Pre-fill template for Blur liquidation entry

Phase 5: Excel Export + Polish

Goal: Tax-ready export, production polish.

- SheetJS integration to build 3-sheet workbook

- Export filtered to 2025 date range
- Native share sheet integration
- Summary calculations (total gains, losses, net)
- Loading skeletons, error boundaries, empty states
- App icon, splash screen

---

8. Claude Code Prompt Strategy

When working with Claude Code, give it one phase at a time. Here's how to prompt each:

Phase 1 example prompt:

> Initialize a new Expo project with TypeScript. Set up React Navigation with a bottom tab navigator containing 4 tabs: Dashboard, Trades, Manual Entry, and Settings. Build the Settings screen first — it should have cards for each platform (Crypto.com, Hyperliquid, Blur, Ethereum, Solana) where users can input and save API keys using expo-secure-store. Include a "Test Connection" button per platform (can be a stub for now). Use a dark theme. The other tabs can be placeholder screens.

Phase 3 example prompt:

> Add API integrations for all 5 platforms. Here are the details: [paste the relevant section from 4.1–4.5 above]. Create a services/ directory with one file per platform. Each service should fetch balances and trade history, then normalize results into the TradeRecord and PortfolioHolding types defined here: [paste types from Section 6]. Create an aggregator service that calls all platforms and merges results. Handle errors gracefully — if one platform fails, the others should still work.

---

9. Known Risks + Mitigations

Risk	Impact	Mitigation
Blur has no stable public API	Can't auto-fetch NFT trade data	Reservoir API as primary, manual entries as backup
Crypto.com API auth is complex	Integration takes longer	Follow their HMAC signing docs carefully, test with small requests first

CoinGecko rate limits on free tier	Price data gaps	Cache aggressively (5min TTL), only fetch what’s needed
NFT valuation is subjective	Tax export accuracy	Manual override fields for price/value, notes field for justification
Hyperliquid unrealized PnL fluctuates	Dashboard value jumps	Clearly label “unrealized” vs “realized” in UI
CAD/USD conversion for taxes	Wrong tax amounts	Add currency toggle, use Bank of Canada daily rate API for conversion

---

10. Post-MVP Ideas (Future)

- Push notifications for large price moves or liquidation warnings
- Portfolio performance chart over time (daily snapshots)
- Cost basis tracking methods (FIFO, LIFO, ACB for Canada)
- Direct CRA/IRS tax form generation
- DeFi protocol tracking (Aave, Uniswap positions)
- Wallet connect integration instead of manual address entry