

# Web3 Terminal – Unified Crypto Portfolio and Trading Dashboard

## 1. Project Title

*Web3 Terminal – A Unified Platform to Track and Manage Portfolios Across All Exchanges and*

*Wallets*

---

## 2. Problem Statement

*In the crypto trading ecosystem, users often maintain multiple wallets (MetaMask, Phantom, Trust Wallet) and accounts on various exchanges (Binance, Coinbase, KuCoin). Tracking overall portfolio performance, asset allocation, and profits across all platforms becomes complex and Time-consuming.*

**Problem:** *There is no single terminal that provides a complete, real-time view of both centralized (CEX) and decentralized (DEX/Web3) holdings, along with trade execution and analytics.*

**Goal:** *To build a Web3-enabled trading dashboard that aggregates wallet and exchange data into a single, secure platform with real-time analytics, AI-powered insights, and optional trade execution.*

---

## 3. System Architecture

### Overall Flow:

Frontend → Backend (API + Data Aggregation Layer) → Database → Blockchain / Exchange APIs

### Tech Stack:

- **Frontend:** React.js + Next.js + TailwindCSS
- **Backend:** Node.js + Express.js
- **Database:** MongoDB (portfolio data) + Redis (cache for prices)

- **Authentication:** JWT + OAuth 2.0 + WalletConnect
  - **Blockchain Interaction:** Ethers.js / Web3.js + Alchemy / Moralis APIs
  - **Hosting:**
    - Frontend → Vercel / Netlify
    - Backend → Render / Railway
    - Database → MongoDB Atlas
- 

## 4. Key Features

Category	Features
<b>Authentication &amp; Authorization</b>	Secure login with Google or wallet (MetaMask / WalletConnect). Encrypted storage of exchange API keys.
<b>Portfolio Aggregation</b>	Connect multiple CEX (Binance, Coinbase, KuCoin) and DEX wallets. Fetch balances, prices, and trade history.
<b>Unified Dashboard</b>	Real-time visualization of total net worth, asset allocation, and performance over time.
<b>DeFi &amp; NFT Tracking</b>	Track staking positions, liquidity pools, and NFT collections across chains.
<b>Trade Execution (Advanced)</b>	Integrate DEX aggregator APIs (1inch, Uniswap) for in-app trading.
<b>Analytics &amp; Insights</b>	Calculate PnL, volatility, Sharpe ratio, and suggest rebalancing actions.
<b>AI Assistant (Optional)</b>	Natural language queries like “Show my top performing assets this month.”
<b>Notifications &amp; Alerts</b>	Price movement, gas fee optimization, and liquidation risk alerts.
<b>Hosting &amp; Deployment</b>	Deploy frontend + backend publicly; connect to live APIs.
<b>Dynamic Data Fetching</b>	The system will dynamically fetch live crypto prices, wallet balances, and exchange portfolio data using APIs such as Binance

API, CoinGecko API, and Moralis/Alchemy Web3 APIs. All dashboard values will update in real time.  
Pages like dashboard, wallet view, portfolio distribution, asset details → **fetch data from real APIs**.

---

## 5. Tech Stack

Layer	Technologies
Frontend	React.js, Next.js, TailwindCSS, Web3Modal, Axios
Backend	Node.js, Express.js / FastAPI
Blockchain / Web3	Ethers.js, Web3.js, Moralis / Alchemy APIs
Database	MongoDB (Atlas) / PostgreSQL
Cache Layer	Redis
APIs	Binance API, CoinGecko API, 1inch API, Uniswap GraphQL
Authentication	JWT, OAuth 2.0, AES Encryption
Hosting	Vercel, Render/ Railway
AI Layer (optional)	OpenAI API / LangChain for portfolio assistant

---

## 6. API Overview

Endpoint	Method	Description	Access
/api/auth/signup	POST	Register a new user with email or wallet	Public
/api/auth/login	POST	Login and return JWT	Public
/api/portfolio/connect-exchange	POST	Connect exchange using API keys	Authenticated
/api/portfolio/connect-wallet	POST	Link Web3 wallet address	Authenticated
/api/portfolio/summary	GET	Fetch aggregated balances and performance	Authenticated

<code>/api/portfolio/insights</code>	GET	Generate analytics (PnL, ROI, allocation)	Authenticated
<code>/api/trade/execute</code>	POST	Execute a DEX trade via 1inch aggregator	Authenticated
<code>/api/alerts/create</code>	POST	Create a price or performance alert	Authenticated

## 7. Extended Functional Features

### CRUD Operations using APIs and Database

The system will include full **Create, Read, Update, and Delete (CRUD)** functionality through RESTful APIs integrated with the **MongoDB** database.

- Users can **add**, **view**, **edit**, and **delete** their assets, wallets, or alerts directly from the dashboard.
- CRUD operations ensure smooth data management, persistence, and real-time synchronization across frontend and backend.
- All APIs will be secured using JWT authentication and input validation.

### Searching, Sorting, Filtering, and Pagination

To enhance user experience and data accessibility, the platform will support:

- **Search:** Keyword-based search on assets, transactions, and wallets.
- **Sort:** Sorting by parameters like asset value, profit/loss, chain, and performance.
- **Filter:** Filtering data based on asset type, blockchain network, or time period.
- **Pagination:** Efficient data pagination for large datasets, ensuring optimized performance and faster load times.

### Routing for Multiple Pages

The application will include **multi-page routing** implemented through **Next.js** routing structure:

- Each feature (Dashboard, Portfolio, Wallets, Trades, Analytics, Alerts) will be accessible through separate routes.
- Dynamic routes will display detailed views for specific assets or transactions (e.g., `/assets/[id]`).
- Client-side routing will ensure smooth and fast navigation without page reloads.

## Dynamic Fetching of Data

The dashboard will dynamically fetch and update data in real-time using APIs such as:

- **CoinGecko API** – for live crypto prices
  - **Binance API** – for exchange balances and trades
  - **Moralis / Alchemy** – for Web3 wallet data
  - **1inch / Uniswap APIs** – for trade execution
- Dynamic fetching ensures that users always see the most up-to-date balances, prices, and analytics without manual refresh.