**FRONTEND DEVELOPMENT WITH REACT.JS PROJECT DOCUMENTATION**

1. **INTRODUCTION:-**

**PROJECT TITLE: CRYPTO CURRENCY DASHBOARD**

**TEAM MEMBERS: 4**

|  |  |
| --- | --- |
| **TEAM ID** | **SWTID1741150814** |
| **TEAM LEADER** | **H PAVITHRA** |
| **TEAM MEMBER** | **M YOGALAKSHNI** |
| **TEAM MEMBER** | **M SANGEETHA** |
| **TEAM MEMBER** | **A PRIYA** |

1. **PROJECT OVERVIEW:-**

A Cryptocurrency Dashboard is a tool designed to provide users with real-time information and analytics about various cryptocurrencies.

**Purpose:**

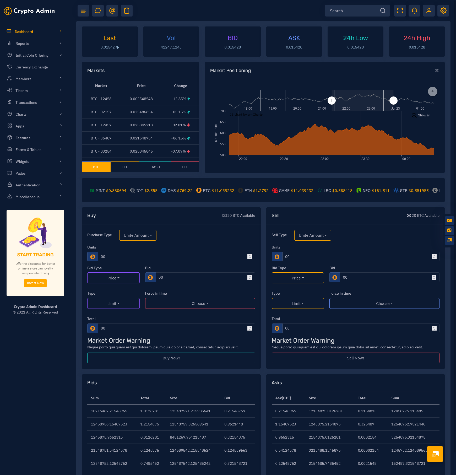
* To track the performance and price trends of different cryptocurrencies.
* To help users make informed decisions about buying, selling, or holding assets.
* To provide insights and analysis of the crypto market.

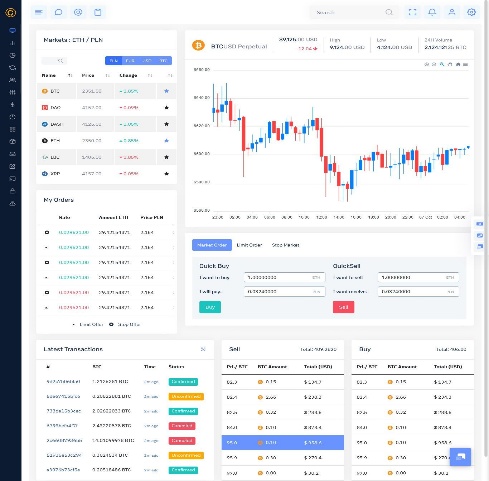
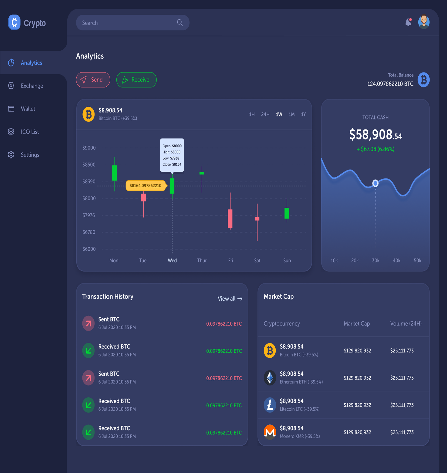
**Features:**

* Price Tracking: Real-time updates of cryptocurrency prices across multiple exchanges.
* Portfolio Management: Users can track their investments, monitor gains/losses, and portfolio performance.
* Market Analysis: Charts, graphs, and technical indicators to analyze trends.
* News Feed: Latest news and updates related to cryptocurrencies.
* Alerts/Notifications: Customizable alerts for price changes or market events.
* Wallet Integration: Allows users to connect their wallets to track balances.

1. **ARCHITECTURE:-**





**A Cryptocurrency Dashboard serves as a centralized platform to monitor and manage cryptocurrency investments, providing real-time data and analytics. Here's a concise overview of its architecture:**

**1. Component Structure:**

* **Frontend:**
  + **User Interface (UI):** Developed using frameworks like React or Angular, the UI presents real-time data, charts, and user interactions.
  + **State Management**: Utilizes tools such as Redux or React's Context API to manage the application's state, ensuring consistent data flow and responsiveness.

[**upgrad.com**](https://www.upgrad.com/blog/react-js-architecture/?utm_source=chatgpt.com)

* **Backend:**
  + **Server:** Built with technologies like Node.js or Django, it handles API requests, user authentication, and data processing.
  + Database: Stores user data, transaction histories, and other persistent information using systems like MongoDB or PostgreSQL.
* External Services:
  + Cryptocurrency APIs: Fetch real-time market data from sources like CoinGecko or Coinbase.

[coingecko.com](https://www.coingecko.com/learn/crypto-price-tracker-react?utm_source=chatgpt.com)

* + Authentication Services: Integrate with platforms like Web3Auth for secure user authentication.

[blog.web3auth.io](https://blog.web3auth.io/intents-blockchain-architecture/?utm_source=chatgpt.com)

**2. State Management:**

* **Frontend State:**
  + **Local State:** Managed within individual components for transient data.
  + **Global State:** Utilizes state management libraries to handle data shared across multiple components, ensuring synchronization and scalability.
* **Backend State:**
  + **Session Management:** Maintains user sessions and authentication tokens.
  + **Data Consistency:** Ensures reliable transactions and data integrity, especially during high-frequency data updates.

**3. Routing:**

* **Frontend Routing:**
  + Client-Side Routing: Employs libraries like React Router to navigate between different views (e.g., dashboard, portfolio, settings) without full page reloads, enhancing user experience**.**
* **Backend Routing:**
  + API Endpoints: Defines routes to handle various HTTP requests (GET, POST, etc.) for data retrieval, submission, and user actions.

1. **SET UP INSTRUCTIONS:**

**Clone the Repository**:  
 Start by cloning the repository using the command git clone <repository\_url>. Navigate to the project directory using cd <project\_directory>. Make sure you have Node.js installed, and check the project's README file for any specific prerequisites.

**Install Dependencies and Configure API Keys**:  
 Run npm install to install all required packages listed in the package.json. After installation, create a .env file in the root directory and add your API keys for real-time cryptocurrency data. Once set up, run npm start to launch the application locally and begin testing the dashboard.

1. **Installation**:

1. Clone repository: git clone <https://github.com/your-repo/cryptoverse.git>

2. Install dependencies: npm install

3. Configure environment variables

1. **Folder Structure**

/cryptoverse

??? /src ?

??? /components

? ??? /pages

? ??? /redux

? ??? /utils

? ??? /assets

??? package.json

??? README.md

1. **Running the Application:**

**Local Development**:

Clone the repository, install dependencies with npm install, and run the app using npm start to view the dashboard locally.

**API Integration**:

Set up API keys for real-time data fetching, configure environment variables, and ensure endpoints are correctly connected for live price and portfolio updates.

The application will be accessible at <https://github.com/pavithra240/Cryptoverse.git>

1. **Component Documentation Key Components:**

**Crypto Price Card**:

* 1. Displays current price, market cap, and 24-hour change for a specific cryptocurrency.
  2. Accepts dynamic props like coin symbol, price, and percentage change.
  3. Reusable for any coin with real-time data updates.
  4. Supports styling for positive and negative price changes.

**Portfolio Summary**:

* 1. Displays user's portfolio balance, individual coin holdings, and profit/loss.
  2. Accepts dynamic user data and recalculates total value in real-time.
  3. Supports multi-coin tracking with quick updates.
  4. Easily customizable to include additional metrics like ROI or historical performance.

1. **Reusable Components:**

- **Button**: Customizable

- **Loader**: Loading animations

- **Crypto Price Card**: Displays current price, 24-hour change, and market cap, reusable for any cryptocurrency with dynamic props.

**- Portfolio Summary**: Shows total balance and individual coin performance, reusable for user-specific data with real-time updates.

**10. State Management**

**Global State:** Managed using Redux Toolkit

**Local State**: Managed with React?s useState

**Transaction History Management**: Tracks all buy/sell transactions, supporting filtering by date, coin type, and transaction status.

**Balance Management**: Maintains an up-to-date view of user’s holdings across different cryptocurrencies, with automatic updates on price fluctuations.

**11.Styling**

Using Tailwind CSS with a custom theme for a modern UI

**Dark Mode Theme**: Use dark background with light text for better readability and a modern aesthetic, especially in low-light environments.

**Card-based Layout**: Organize data into responsive cards with subtle shadows, creating a clean and user-friendly interface for quick information access.

**12.Testing**

Testing Strategy:

- **Integration Testing**: Cypress for E2E tests

- **Unit Testing**: Test individual components like price cards and charts using Jest or Mocha to ensure correct data rendering and functionality.

- **End-to-End Testing**: Use Cypress or Selenium to simulate user interactions, ensuring the dashboard updates in real-time and handles transactions correctly.

**13.Known Issues**

API rate limits may affect updates.

Dark mode support in progress.

**Data Latency**: Real-time price updates may experience delays due to API rate limits or network issues, leading to outdated information.

**Volatility Handling**: Sudden market fluctuations can cause discrepancies in displayed prices, potentially confusing users if data isn't updated quickly enough.

14.**Future Enhancements Future Plans:**

- Push notifications for price alerts

- AI-driven price predictions

- Staking feature for rewards

-**AI-Powered Insights**: Integrate machine learning to provide personalized investment recommendations and market trend predictions based on user behavior.

**-Multi-Chain Support**: Expand the dashboard to support multiple blockchains and decentralized assets, offering a broader view of the entire crypto ecosystem.