Frontend Development with React.js

Project Documentation for Cryptoverse - Cryptocurrency

---

1. Introduction

- **Project Title**: Cryptoverse – Cryptocurrency
- **Team Members**:
  - SAKTHIVEL J (Team Leader) [Email Id: sakthivelsakthivel43838@gmail.com]
  - PARAMASIVAM S [Email Id: sivamsivam9797371940@gmail.com]
  - RAGHUL P [Email Id: rahulrahul637491@gmail.com]
  - RAKESH E [Email Id: rakeshagastine@gmail.com]
  - SANJAI KUMAR V [Email Id: sanjaikumarmass12@gmail.com]
  - **2. Project Overview**

- **Purpose**:

  Cryptoverse is a web application designed to provide users with real-time cryptocurrency data, including price tracking, market trends, and portfolio management. The goal is to create an intuitive and visually appealing platform for both novice and experienced cryptocurrency enthusiasts.

- **Features**:
  - Real-time cryptocurrency price tracking.
  - Interactive charts for market analysis.

- Portfolio management to track user investments.
- News feed for the latest updates in the crypto world.
- User authentication and profile management.

---

3. Architecture

- **Component Structure**:

  The application is built using a modular component structure. Major components include:

  - **Header**: Navigation bar with links to different sections.
  - **Dashboard**: Displays real-time cryptocurrency data and charts.
  - **Portfolio**: Allows users to manage their cryptocurrency investments.
  - **NewsFeed**: Fetches and displays the latest cryptocurrency news.
  - **Footer**: Contains links to legal information and social media.

- **State Management**:
  The application uses **Redux** for global state management. Redux handles the state for user authentication, cryptocurrency data, and portfolio information. Local state is managed using React's `useState` and `useReducer` hooks.

- **Routing**:
  The application uses **React Router** for navigation. Routes are defined for the Dashboard, Portfolio, NewsFeed, and User Profile pages.

---

4. Setup Instructions

- **Prerequisites**:
    - Node.js (v16 or higher)
    - npm (v8 or higher)
    - Git (for cloning the repository)
- **Installation**:

1. Clone the repository: gitclone https://github.com/asunm12911747/cryptocurrency.git
2. Navigate to the project directory: cd cryptoverse
3. Install dependencies: npm install
4. Configure environment variables:

   Create a .env file in the root directory and add the required API keys and configurations.
5. Start the development server: npm start

---

5. Folder Structure

- **Client**:
    - **Src/components**: #Resuable UI components (cryptoPriceTracker,etc.)
    - **Src/pages**: #page components (Dashborad, portfolio,etc.)
    - **Src/assets**: #Other static files (Image,icons,etc.)
    - **Src/utils**: #utility functions and helpers.
    - **Src/hooks**: #Custom React hooks
    - **Src/store**: #Redux store and slices
    - **Src/styles**: #Global and components - Specific styles
    - **Src/App.js**: #Main application components
    - **Src/index.js**: #Entry point

- **Utilities:**
    - **api.js**: Handles API requests to fetch cryptocurrency data.
    - **auth.js**: Manages user authentication and token storage.
    - **formatters.js**: Utility functions for formatting currency and dates.

---

6. Running the Application js

- **Frontend**:
- Run the following command in the project directory:
- npm start
- The application will be available at http://localhost:3000.

---

7. Component Documentation

- **Key Components**:
    - **Dashboard**: Displays real-time cryptocurrency prices and charts. Receives data as props from the Redux store.
    - **Portfolio**: Allows users to add, edit, and delete cryptocurrency holdings. Uses local state for form inputs.
    - **NewsFeed**: Fetches news articles from an external API and displays them in a list.
- **Reusable Components**:
    - **CryptoCard**: A card component that displays basic information about a cryptocurrency.
    - **Chart**: A reusable chart component that visualizes price trends.

---

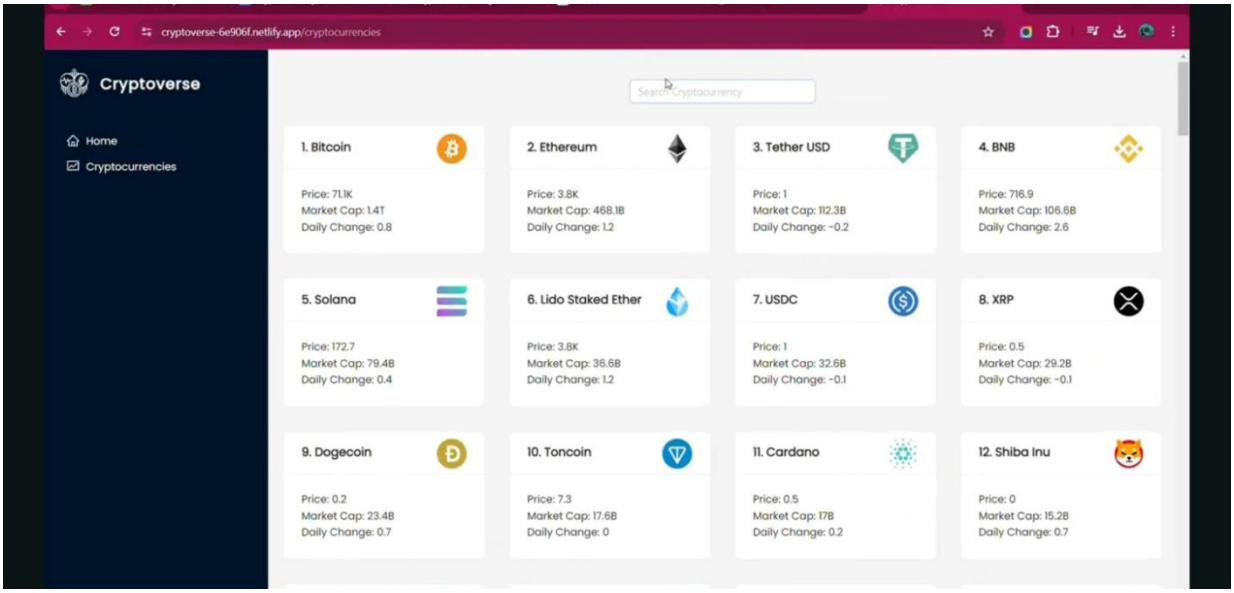
8. State Management

- **Global State**:
  The Redux store manages the following global states:
  - User authentication status.
  - Cryptocurrency data fetched from APIs.
  - Portfolio data for logged-in users.
- **Local State**:
  Local state is used within components for form inputs, toggles, and other UI interactions. For example, the `Portfolio` component uses local state to manage form inputs for adding new holdings.

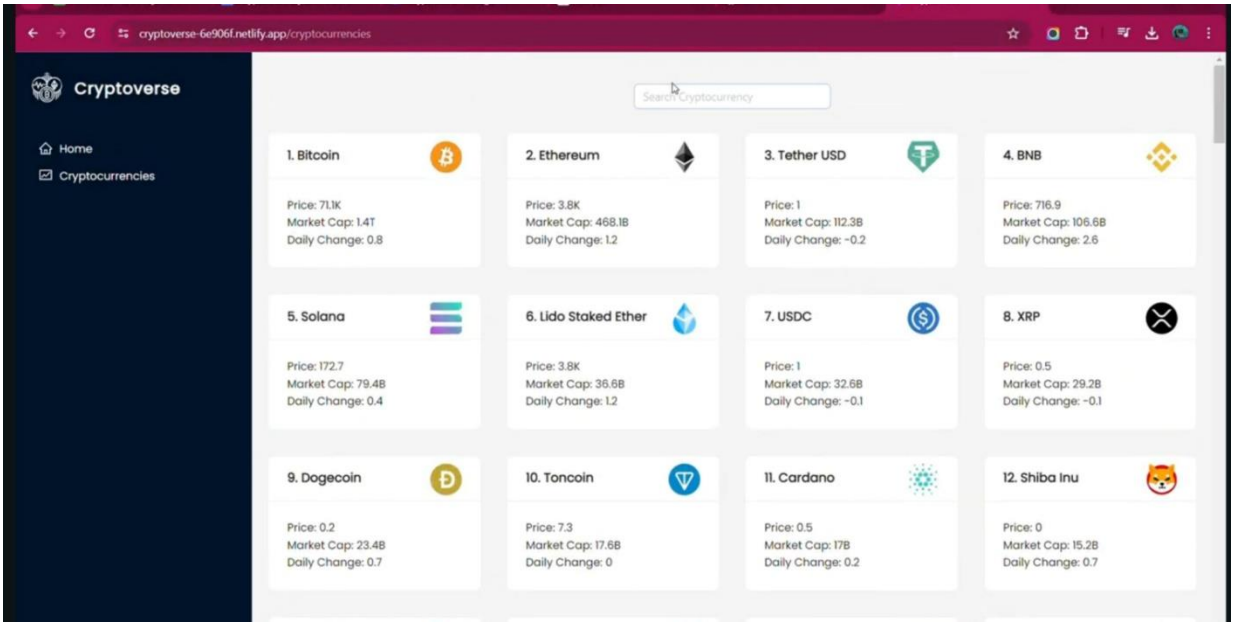
---

9. User Interface

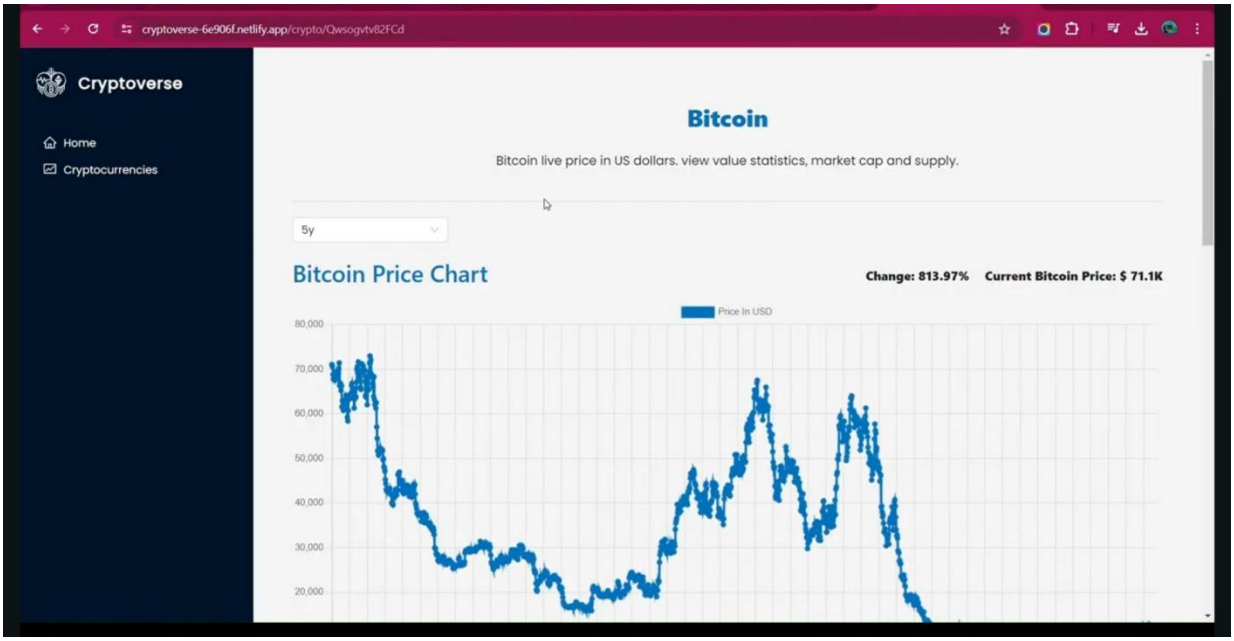
- **Screenshots**:
- **Home page:** Providing user with quick access to key statistics and information



Search page: Allows user to search for specific cryptocurrencies.



Bitcoin page: "Digital Gold" due to its finite supply and store-of-value properties.



---

10. Styling

- **CSS Frameworks/Libraries**:

  The application uses **Styled-Components** for component-specific styling and **Tailwind CSS** for utility classes.

- **Theming**:

  A custom theme is implemented using Styled-Components, allowing for easy switching between light and dark modes.

---

11. Testing

- **Testing Strategy**:
  - Unit testing is done using **Jest** and **React Testing Library**.
  - End-to-end testing is performed using **Cypress**.

- **Code Coverage**:
  The project maintains a code coverage of **85%** using Jest's coverage reporting.

---

12. Screenshots or Demo

- **Demo Link**:

  https://drive.google.com/drive/folders/12zXOI1yHgXCnKnejvPWL3-TWpypxYrga?usp=sharing

  - **Screenshots**: See Screenshot 9 for UI Screenshot

---

13. Known Issues

- **Issue 1**: The chart component occasionally fails to render on slow network connections.
- **Issue 2**: The news feed API has a rate limit, which may cause delays in fetching new articles.

---

14. Future Enhancements

- **Future Features**:
  - Add support for more cryptocurrencies.
  - Implement a mobile app version using React Native.
  - Enhance the charting library for better performance.
  - Add social features, such as user comments and sharing.

This documentation provides a comprehensive overview of the Cryptoverse - cryptocurrency project including architecture, setup, instructions, and future plans