***Frontend Development with React.js***

***Project Documentation for CryptoVerse***

**1. Introduction**

**Project title : Cryptoverse**

**Team members :**

**Priyadharshini S(Team leader)**[**priyadharshinis22cs064@gmail.com**](mailto:priyadharshinis22cs064@gmail.com)

**Priyadharshini R** [**priyadharshinir22cs063@gmail.com**](mailto:priyadharshinir22cs063@gmail.com)

**Rabiyathul Bassariya A** [**rabiyathulbassariyaa22cs065@gmail.com**](mailto:rabiyathulbassariyaa22cs065@gmail.com)

**Ragavarshini R** [**ragavarshinir22cs066@gmail.com**](mailto:ragavarshinir22cs066@gmail.com)

**2. Project Overview**

• Purpose:

Cryptoverse is a digital platform that provides real-time updates, insights, and information about cryptocurrencies. Users can track market trends, view historical data, and explore various digital assets.

• Features:

* Cryptocurrency Market Overview.
* Live Price Updates.
* User Authentication.
* Trending Coins and Market Insights.
* Historical Data Visualization.

**3. Architecture**

Component Structure:

The application is built using React.js with a component-based architecture. Major components include:

* Header: Contains the navigation bar and search bar.
* Sidebar: Displays trending cryptocurrencies and navigation links.
* HomePage: Shows Market Trends, Most Popular Cryptos.
* SearchPage: Allows users to search for specific cryptocurrencies and tokens.

• State Management:

The application uses Redux for global state management. The Redux store manages user authentication, market data, and search results.

• Routing:

The application uses React Router for navigation. Routes include.

* /: Home page
* /search: Search page
* /login: User login page

**4. Setup Instructions**

• Prerequisites:

* Node.js (v16 or higher)
* npm (v8 or higher
* Git

• Installation:

1. Clone the repository: git clone :

<https://github.com/sundaramoorthypriyadharshini/Cryptoverse_Priyadharshinis>

2. Install dependencies: npm install

3. Configure environment variables: Create a .env file in the client directory and add the necessary variables (e.g., API keys).

4. Start the development server: npm start

5. Folder Structure

• Client:

* src/components: # Reusable components (Header, MarketList, etc.)
* src/pages: # Page components (HomePage, SearchPage, etc.)
* src/assets: # Icons, images, and other static files.
* src/redux: # Redux store, actions, and reducers.
* src/utils: # Utility functions and helpers.
* App.js: # Main application component. o index.js: # Entry point

• Utilities:

* api.js: Handles API requests to the backend.
* auth.js: Manages user authentication and token storage.

**6. Running the Application**

Frontend:

* To start the frontend server, run the following command in the client directory: npm start
* npm install
* The application will be available at http://localhost:3000

**7. Component Documentation**

* Key Components:
* Header: Displays the navigation bar and search bar.
* Props: onSearch (function to handle search queries).
* MarketList: Displays a list of cryptocurrencies with prices.
* Props: cryptos (array containing market details), onSelect (function to handle selection).
* Reusable Components:
* Button: A customizable button component.
* Props: text, onClick, disabled.
* Input: A reusable input field for forms and search.
* Props: type, placeholder, value, onChange.

8. State Management

• Global State:

The Redux store manages the following global states:

* user: Current authenticated user.
* marketData: Live cryptocurrency market data and trends.search.
* Results: Results from the search functionality.

• Local State:

Local state is managed using React's useState hook within components. For example, the SearchPage component manages the search query input locally.

**9. User Interface**

• Screenshots

* Home Page: Displays trending cryptocurrencies and market trends.
* Popular Cryptocurrencies: Highlights the most traded and valuable coins.
* Market Insights: Displays market fluctuations and historical data charts.

**10. Styling**

• CSS Frameworks/Libraries:

The application uses Styled-Components for styling. This allows for modular and scoped CSS within components.

• Theming:

A custom theme is implemented using Styled-Components, with support for light and dark modes.

**11. Testing**

• Testing Strategy:

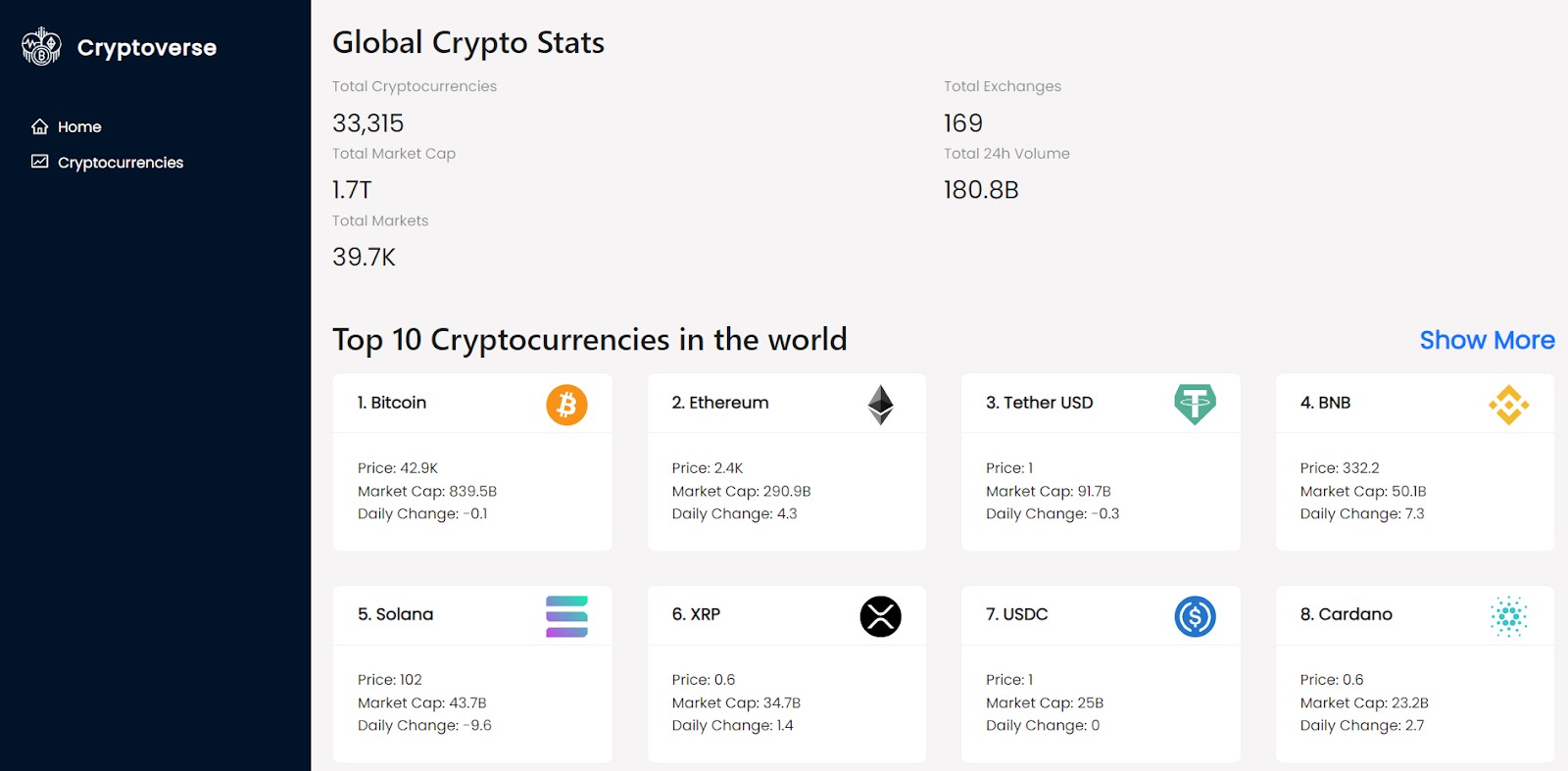
* Unit Testing: Using Jest and React Testing Library.
* Integration Testing: Ensures that components work together as expected.
* End-to-End Testing: Cypress is used for end-to-end testing of user flows.
* Code Coverage:

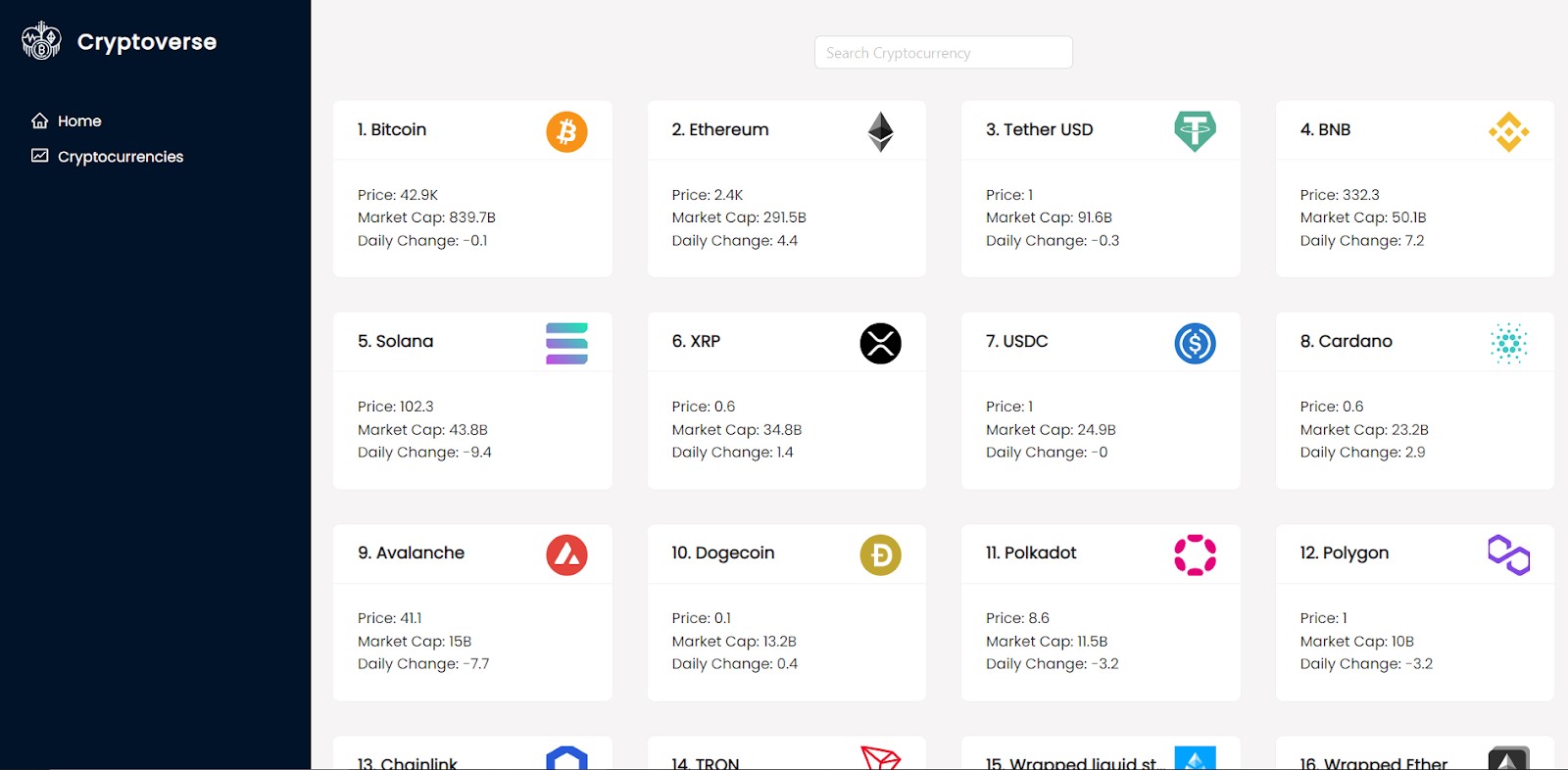
Code coverage is monitored using Jest’s built-in coverage tool. The current coverage is 85%.

**12. Screenshots or Demo**

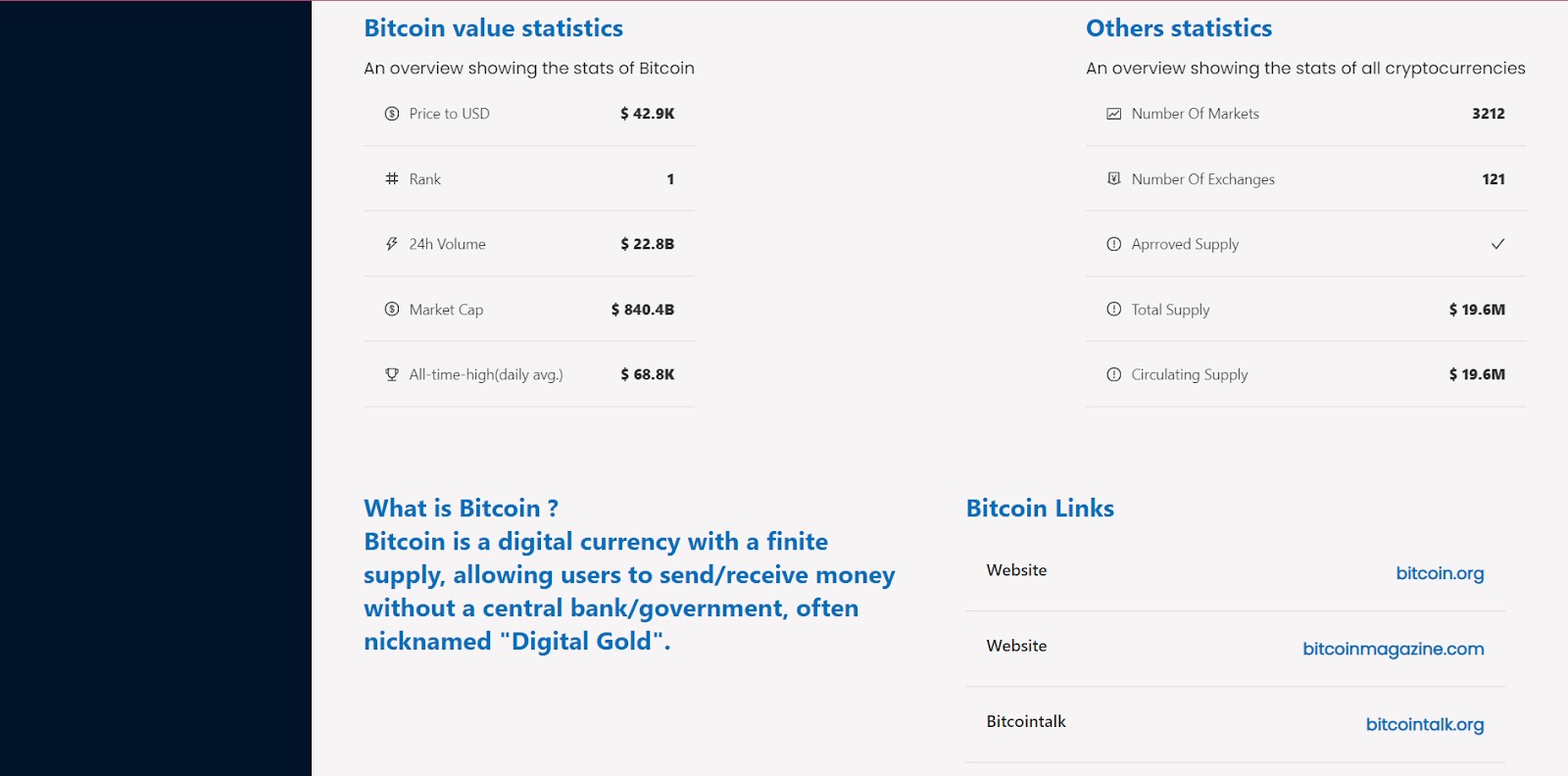
• Demo Link:

<https://drive.google.com/file/d/1E77S9Oc0id3LI_bQlMijU2JLxFAL_pIw/view?usp=drive_link>









**13. Known Issues**

• Issue 1: The search functionality is slow with large datasets.

**14. Future Enhancements**

• Future Features:

* Add support for user profiles and portfolio tracking.
* Implement a recommendation engine for personalized cryptocurrency insights.
* Add animations and transitions for a smoother user experience.

This documentation provides a comprehensive overview of the CryptoVerse project, including its architecture, setup instructions, and future plans.