Crypto Project Documentation

Project Overview

This project is a cryptocurrency tracking application built using React and Vite. It allows users to view cryptocurrency prices, trends, and other relevant data.

Architecture

The project follows a modular architecture with the following key components:

- React for UI
- Vite for fast development and build process
- State management using Context API
- Service layer for API interactions
- Component-based structure for UI elements

Setup Instructions

1. Clone the repository:
2. git clone <repository_url>

```
cd Crypto-project-main
```

3. Install dependencies:

```
npm install
```

4. Start the development server:

```
npm run dev
```

Folder Structure

```
Crypto-project-main/
│-- public/            # Static assets
│-- src/
│    │-- app/          # Business logic and app-related modules
│    │-- assets/       # Images and static resources
│    │-- components/    # Reusable React components
│    │-- services/      # API calls and utility functions
│    │-- App.jsx        # Root component
│    │-- main.jsx       # Entry point
│    │-- index.css      # Global styles
│-- package.json       # Project dependencies
│-- vite.config.js     # Vite configuration
│-- index.html         # Main HTML file
```

Running the Application

To run the application in development mode:

```
npm run dev
```

To create a production build:

```
npm run build
```

Components Documentation

The components/ folder contains reusable UI components. Key components include:

- Navbar: Displays the navigation bar.
- CryptoList: Shows a list of cryptocurrencies.
- CryptoDetail: Displays detailed information about a selected cryptocurrency.

State Management

The project uses **React's**React’s Context API for state management. The state logic is handled inside the app/ directory.

User Interface

The UI is built using React components and styled using CSS. The design focuses on usability and responsiveness.

Styling

Global styles are maintained in index.css, and component-specific styles are in App.css or individual component files.

Testing

To run tests (if available):

```
npm test
```

Known Issues

- API rate limits may cause some data to load slowly.
- The UI may need optimizations for mobile devices.

Future Enhancements

- Add user authentication.
- Implement dark mode.
- Improve error handling for API requests.