**Cryptoverse**

**Introduction:**

Welcome to the Crypto Dashboard, this platform is designed to provide users with powerful tools to analyse and understand the historical performance of cryptocurrencies. With data spanning the past years, Crypto Dashboard offers detailed insights into the price trends, volatility, and market shifts of various digital assets. Whether you're a seasoned investor or a newcomer, this dashboard equips you with the resources to track market dynamics and make well-informed decisions.

The Crypto Dashboard combines intuitive design with advanced analytical capabilities, featuring interactive charts and customizable timeframes. This allows users to compare multiple cryptocurrencies, explore long-term trends, and perform in-depth analyses of price movements. By leveraging historical data, users can uncover recurring patterns, identify key performance indicators, and assess market risks.

In addition to supporting strategic investment decisions, Crypto Dashboard also serves as an educational resource, helping users to better understand the evolving landscape of cryptocurrency markets. Through this documentation, we aim to guide you through the features, functionality, and best practices for making the most of Crypto Dashboard, ensuring you have all the tools you need to succeed in the dynamic world of cryptocurrency.

**Description:**

Crypto verse is an advanced cryptocurrency dashboard that offers investors in-depth insights into market trends through a comprehensive analysis of historical price data over the past five years. The platform features easy-to-understand charts, interactive tools, and smooth navigation, allowing users to identify the best-performing assets and make well-informed investment choices. With its powerful search capabilities, users can effortlessly explore a wide variety of cryptocurrencies and compare their performance over time. Crypto verse not only aids in optimizing investment portfolios but also serves as an educational platform, helping users grasp the constantly changing landscape of the cryptocurrency market.

**Requirements:**

Here are the key prerequisites for developing a frontend application using

React.js: ✔ Node.js and npm:

Node.js is a powerful JavaScript runtime environment that allows you to run JavaScript code on the local environment. It provides a scalable and efficient platform for building network applications.

Install Node.js and npm on your development machine, as they are required to run JavaScript on the server-side

●Download: <https://nodejs.org/en/download/>

●Installation instructions: <https://nodejs.org/en/download/package-manager/>

✔React.js:

React.js is a popular JavaScript library for building user interfaces. It enables developers to create interactive and reusable UI components, making it easier to build dynamic and responsive web applications.

Install React.js, a JavaScript library for building user interfaces.

●Create a new React app:

npm create-react-app my-react-app

Replace my-react-app with your preferred project name.

●Navigate to the project directory:

cd my-react-app

●Running the React App:

With the React app created, you can now start the development server and see your React application in action.

●Start the development server:

npm start

This command launches the development server, and you can access your React app at http://localhost:3000 in your web browser.

✔HTML, CSS & JavaScript: Basic knowledge of HTML for creating the structure of your app, CSS for styling, and JavaScript for client-side interactivity is essential.

✔ Version Control: Use Git for version control, enabling collaboration and tracking changes throughout the development process. Platforms like GitHub or Bitbucket can host your repository.

• Git: Download and installation instructions can be found at: <https://git-scm.com/downloads>

✔Development Environment: Choose a code editor or Integrated Development Environment (IDE) that suits your preferences, such as Visual Studio Code, Sublime Text, or WebStorm.

•Visual Studio Code: Download from https://code.visualstudio.com/download • Sublime Text: Download from https://www.sublimetext.com/download • WebStorm: Download from https://www.jetbrains.com/webstorm/download

✔Get the demo video from google drive:

• Execute the project by this steps:

Drivelink: https://drive.google.com/file/d/1m0m9fNw92Fe4KKRlAaybKclbd1EYs7pk/view?usp=drivesdk

✔Clone the code from GitHub repository:

Follow below steps:

Git repository: <https://github.com/SSC369/cryptoverse>

Git clone command: git clone

https://github.com/SSC369/cryptoverse Use this command to

clone code into your project folder.

Install Dependencies:

• Navigate into the cloned repository directory and install libraries:

cd cryptoverse

npm install

✔Start the Development Server:

• To start the development server, execute the following command:

npm run dev(vite) or npm start

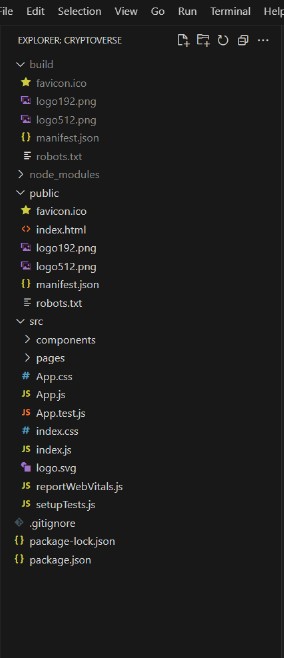
Access the App:

• Open your web browser and navigate to <http://localhost:3000>.

• You should see the Crypto verse app's homepage, indicating that the installation and setup were successful.

You have successfully installed and set up the application on your local machine. You can now proceed with further customization, development, and testing as needed

**Project Structure:**



**Project Flow:**

* Project setup and configuration:

**1. Setup React Application:**

* Create a React app in the client folder.
* Install required libraries
* Create required pages and components and add routes.

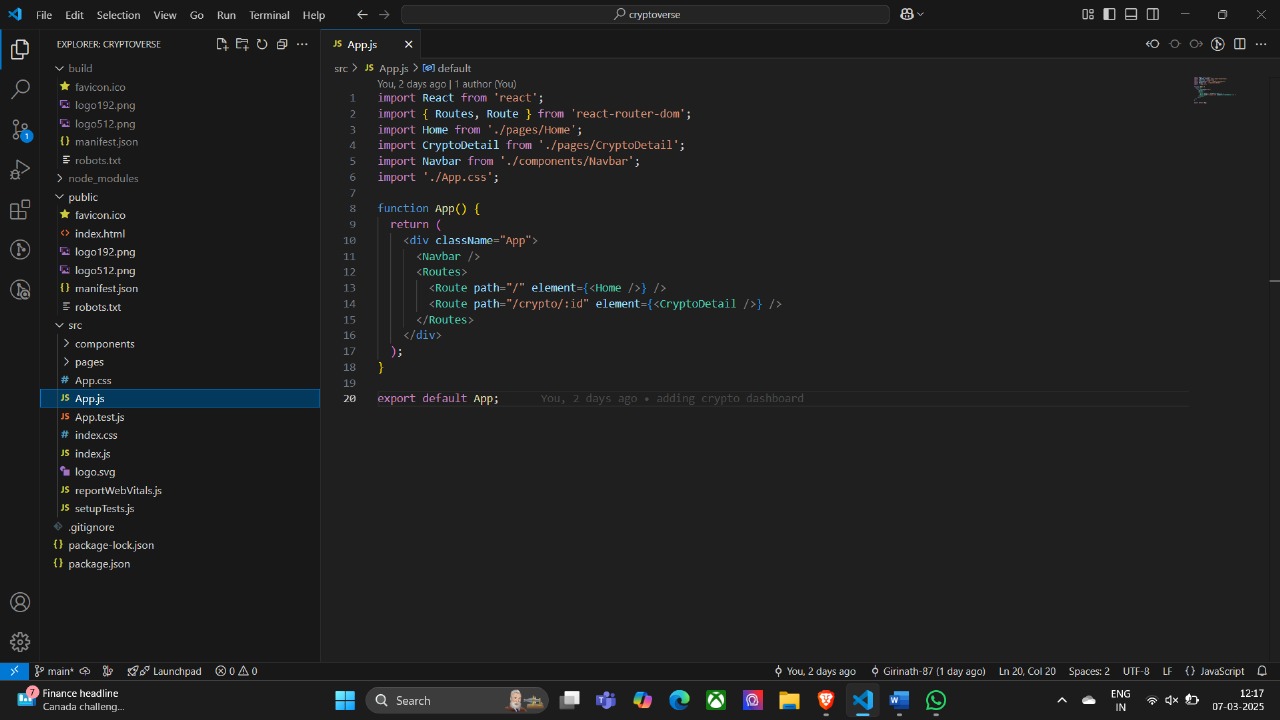
**2.Design UI components:**

* Create Components.
* Implement layout and styling.
* Add navigation**.**

**3.Implement frontend logic:**

* Integration with API endpoints.
* Implement data binding.

**Reference Image:**



**Project Development:**

**Create a redux store:**

1. `import {configure Store} from "@redux js/toolkit";`: This line imports the `configure Store` function from Redux Toolkit. Redux Toolkit is a package that provides utilities to simplify Redux development, making it easier to write Redux logic with less boilerplate code.

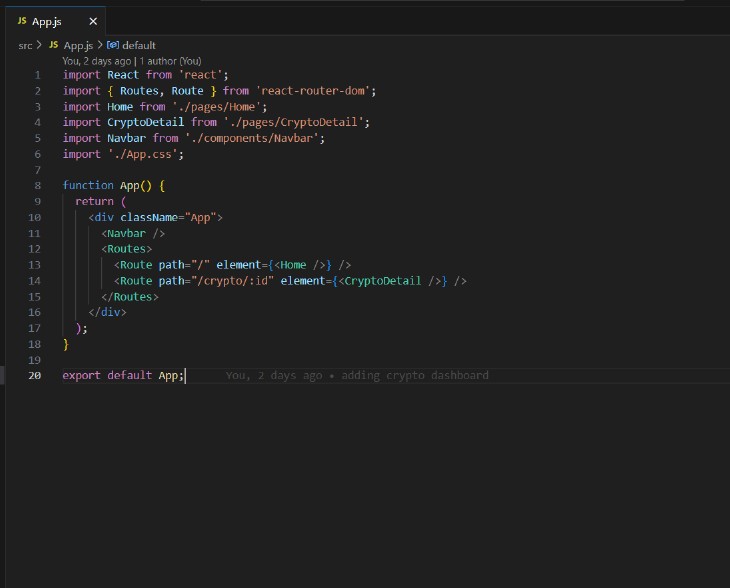
2. `import { CryptoAPI } from "../services/CryptoAPI";`: This line imports the `CryptoAPI` object from the `cryptoApi.js` file located in the `../services` directory. This object likely contains configurations and functions related to making API requests for cryptocurrency data.

3. `export default configure Store({ ... });`: This line exports the Redux store configuration created by the `configure Store` function as the default export of this module.

4. `reducer: { [cryptoApi.reducerPath]: cryptoApi.reducer }`: This part of the configuration specifies the root reducer for the Redux store. In this case, it sets the `cryptoApi.reducer` as the reducer for the slice of state managed by the `cryptoApi` API slice. The `cryptoApi.reducerPath` likely refers to the slice's unique identifier, which is used internally by Redux Toolkit.

5.middleware:(getDefaultMiddleware)=>

Get Default Middleware().concat (cryptoApi.middleware),`: This part of the configuration specifies middleware for the Redux store. Middleware intercepts actions before they reach the reducers and can be used for various purposes, such as logging, asynchronous actions, or handling API requests. Here, it uses the `get Default Middleware` function provided by Redux Toolkit to get the default middleware stack and appends the `CryptoAPI. Middleware`. This middleware likely handles asynchronous API requests and dispatches corresponding actions based on the API response. This setup allows you to manage and interact with this data using Redux within your React application

****

**Adding Providers in the main function:**

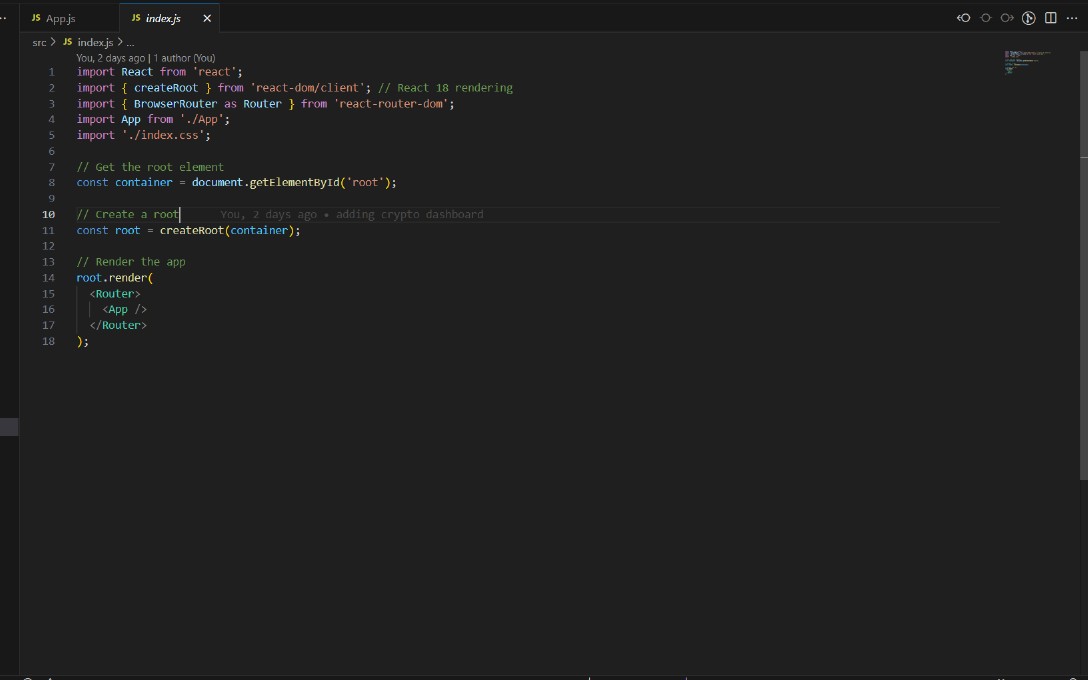
React Router with `BrowserRouter`:

‘<BrowserRouter>`:This component is provided by `react-router-dom` and enables client-side routing using the HTML5 history API. It wraps the application, allowing it to use routing features.

**Redux Provider:**

**-** ` <Provider store={store}>`: This component is provided by `react-redux` and is used to provide the Redux store to the entire application. It wraps the application, allowing all components to access the Redux store.

Overall, this code initializes the React application by rendering the root component (`<APP/>`) into the DOM, while also providing routing capabilities through `BrowserRouter` and state management with Redux through `Provider`. Additionally, it ensures stricter development mode checks with `<React.StrictMode>`.



**Creating cryptocurrencies component:**

1. Component Definition:- `const Cryptocurrencies = ({ simplified }) => { ... }`: Defines a functional component named`Cryptocurrencies`. It accepts a prop named `simplified`, which determines whether to display a simplified version of the list.
2. Initialization:- `const count = simplified ? 10 : 100;`: Initializes the `count` variable based on the value of the `simplified` prop. If `simplified` is true, `count` is set to 10; otherwise, it's set to 100.
3. Fetching Cryptocurrency Data:- `const { data: cryptosList, isFetching } = useGetCryptosQuery(count);`: Uses the `useGetCryptosQuery` hook provided by the `cryptoApi` service to fetch cryptocurrency data. It retrieves the list of cryptocurrencies (`cryptosList`) and a boolean flag (`isFetching`) indicating whether the data is being fetched.
4. Filtering Cryptocurrency Data:- The `useEffect` hook is used to filter the cryptocurrency data based on the `searchTerm` state variable. It updates the `cryptos` state with filtered data whenever `cryptosList` or `searchTerm` changes.
5. Rendering Loader:- `if (isFetching) return ;`: If data is still being fetched (`isFetching` is true), it returns a `Loader` component to indicate that the data is loading.
6. Rendering Search Input:- `!simplified && (...)`: If `simplified` is false, it renders a search input field allowing users to search for specific cryptocurrencies by name.

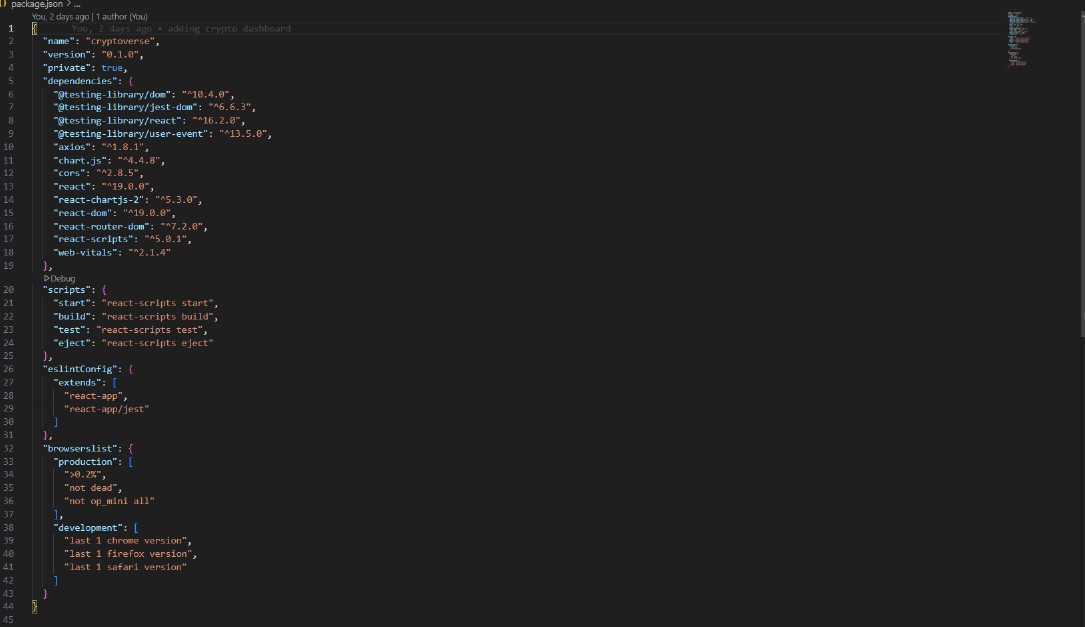
7. Rendering Cryptocurrency Cards: -

The `Row`and`Col`components from Ant Design are used to create a grid layout for displaying cryptocurrency cards.- For each cryptocurrency in the `cryptos` array, it renders a `Card` component containing details such as name, price, market cap, and daily change. Each card is wrapped in a `Link` component, allowing users to navigate to the details page of a specific cryptocurrency.

8. Return Statement:

- `return (...)`: Returns JSX representing the component's structure and content.

Overall, this component fetches cryptocurrency data, filters it based on a search term, and renders the data in a visually appealing format with card-based UI. It also provides a search functionality for users to find specific cryptocurrencies



**Project Execution:**

**Project Source code:**

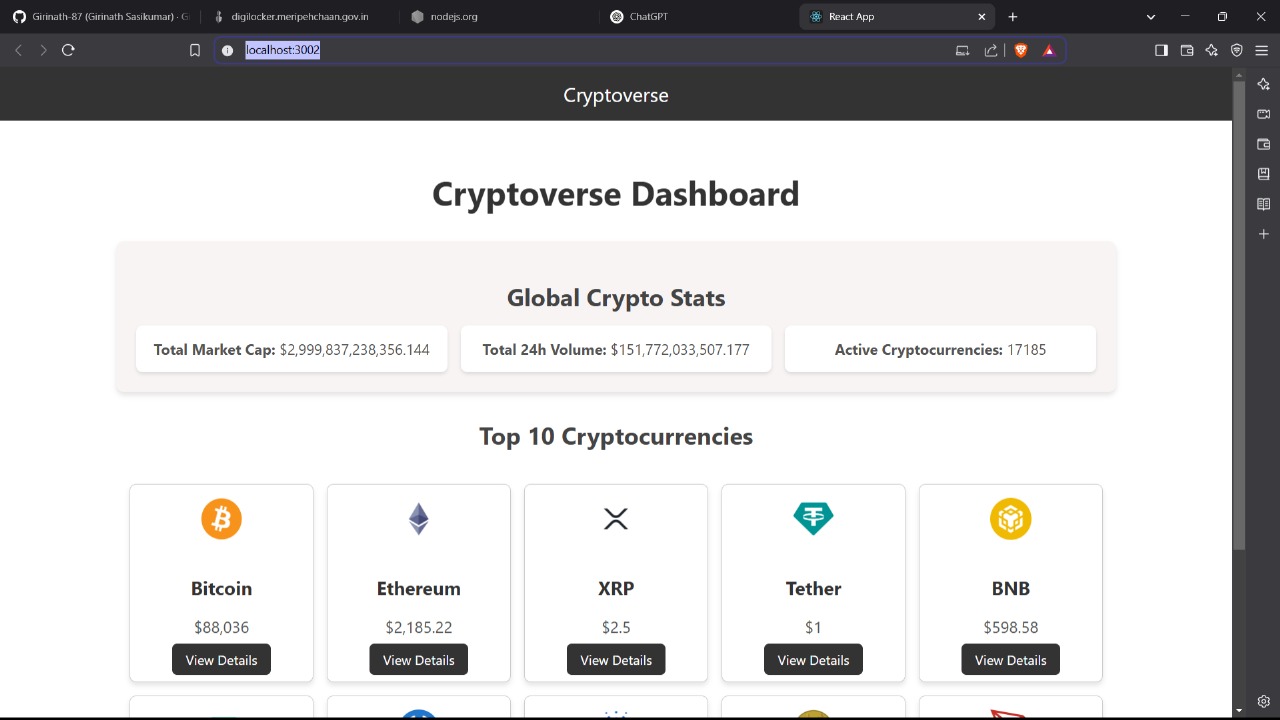
* **GitHub link:**  https://github.com/Girinath-87/Crypto-Dashboard

**Demo video for execution:**

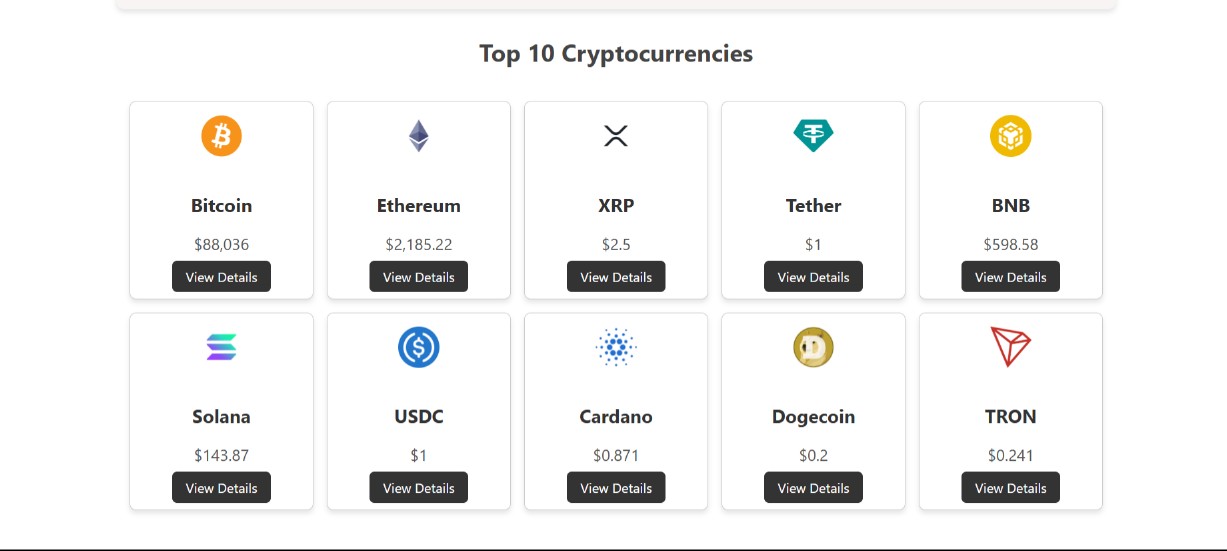
* **Google Drive Link:** https://drive.google.com/file/d/1m0m9fNw92Fe4KKRlAaybKclbd1EYs7pk/view?usp=drivesdk

**User Interface snips:**

**Homepage:** This page consists of stats of global crypto like total cryptocurrencies, total exchanges, market cap etc. Also consist of top 10 cryptocurrencies in the world.



**Crypto currencies page :** This pages contains all cryptocurrencies which are currently in flow in the world. There is also a search feature where users can search and findoutabout their desired cryptocurrency.



Crypto currency details page : This page contains the line chart with data representation of price of cryptocurrencies. Also contains statistics and website links of cryptocurrencies.

A graph showing the price of bitcoin

AI-generated content may be incorrect.

**TEAM DETAILS**

**TEAM LEAD:**

* GIRINATH.S [Mail id- thamayanthisasi83@gmail.com]

**TEAM MEMBERS:**

* GOKUL.V [Mail id- mrthamizhang8@gmail.com]
* GOPI.D [Mail id- dhamodharangopi276@gmail.com]
* GOWTHAM.R [Mail id- gowth7010@gmail.com]
* HARIHARAN.S [Mail id- hariharan.s0044@gmail.com]

**\*\*\*\*\*\*\*\*THANK YOU\*\*\*\*\*\*\*\***