CRYPTOVERSE

**(CryptocurrencyDashboard)**

* **TeamDetails:**

|  |  |
| --- | --- |
| TEAMLEADER | EMAILID |
| KISHORE K M | [kishorepvr2005@gmail.com](mailto:kishorepvr2005@gmail.com) |

|  |  |
| --- | --- |
| TEAMMEMBERS | EMAILID |
| JAYASURYA M | [jaya2005surya@gmail.com](mailto:jaya2005surya@gmail.com) |
| NANDHAKUMAR K | [nandhakumar162005@gmail.com](mailto:nandhakumar162005@gmail.com) |
| THANGARAJ S B | [Thangamindia47@gmail.com](mailto:Thangamindia47@gmail.com) |
| UDAYA J | [Udaya2kboy@gmail.com](mailto:Udaya2kboy@gmail.com) |

**CRYPTOVERSE:**

**Introduction:**

The cryptoverse refers to the expansive digital ecosystem surrounding cryptocurrenciesandblockchaintechnology.Itencompassesavariousapplications, including decentralized finance, virtual worlds, and digital assets, enabling users to engage in innovative financial and creative activities.

Cryptocurrency is a type of digital or virtual currency that uses cryptography for security. It operates on decentralized networks based on blockchain technology, whichis adistributedledgerenforcedbyanetworkofcomputers (oftenreferredto as nodes).

# CurrentTrendsinthe Cryptoverse:

**Mass Adoption:** Cryptocurrencies are moving beyond rich markets, attracting interest from a broader audience, including everyday consumers and institutional investors.

**Environmental Concerns:** The energy consumption associated with mining cryptocurrencies has raised concerns, prompting a shift towards more sustainable practices like Proof of Stake.

**Educational Initiatives:**There is a growing need for educational resources tohelpthegeneralpublicunderstandthecomplexitiesofcryptocurrency.

# Pre-requisites:

### Node.jsandnpm:

Node.jsisapowerfulJavaScriptruntimeenvironmentthat allowsyou to run JavaScript code on the local environment. It providesa scalableand efficient platform for building network application.

* + Download:<https://nodejs.org/en/download/>

**React.js:**It’s 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.

* Startthedevelopmentserver: npm start

Thiscommand launches the developmentserver, and access our React app at [http://localhost:3000](http://localhost:3000/)in our web browser.

✔ HTML, CSS, and JavaScript: Basic knowledge of HTML for creating the structureofyourapp,CSSforstyling,andJavaScriptforclient-sideinteractivity is essential.

✔**Version Control:**Use GIT for version control, that enabling collaboration and tracking changes throughout the development process. We use the Platform GitHub an 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 for our preferences, we use Visual Studio Code.

Visualstudiocode: https://code.visualstudio.com/download.

**InstallDependencies**:

* Navigateintotheclonedrepositoryandinstalllibraries: cd crypto

npminstall

✔StarttheDevelopmentServer:

Tostartthedevelopmentserver,executethefollowing command: npm run dev (vite) or npm start.

### OURGITHUBREPOSITORYLINK:

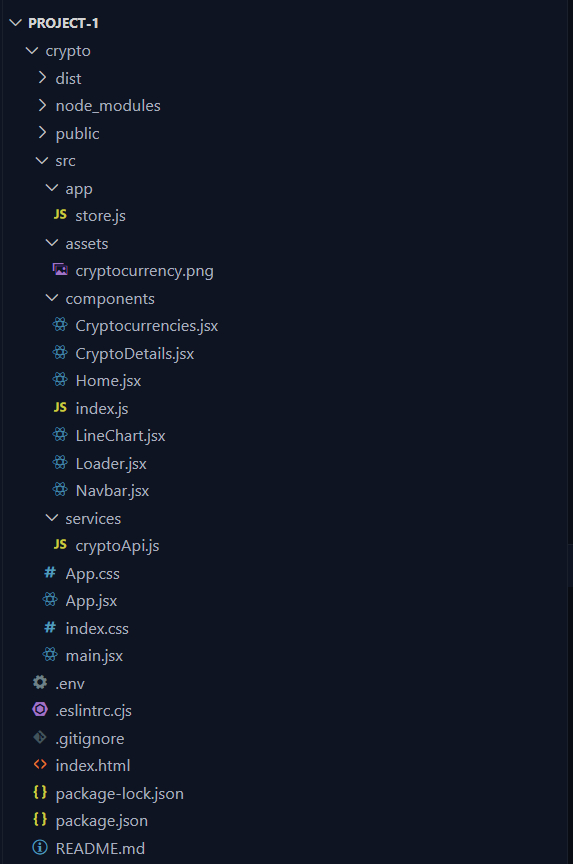
<https://github.com/kishore-2606/cryptocurrency3>

### AccesstheApp:

* + Openyourwebbrowserandnavigateto<http://localhost:5173/>
  + YoushouldseetheourCryptoverseapp'shomepage, indicating that the installation andsetupwere successful.

Nowwehavesuccessfullyinstalledandsetuptheapplicationonour local machine. We can now proceed with further customization, development, and testing as needed.

## ProjectStructure:

****

**ProjectFlow:**

* Projectsetupandconfiguration:

### SetupReactApplication:

* + CreateaReactappintheclientfolder.
  + Installrequiredlibraries
  + Createrequiredpagesandcomponentsandaddroutes.

### DesignUIcomponents:

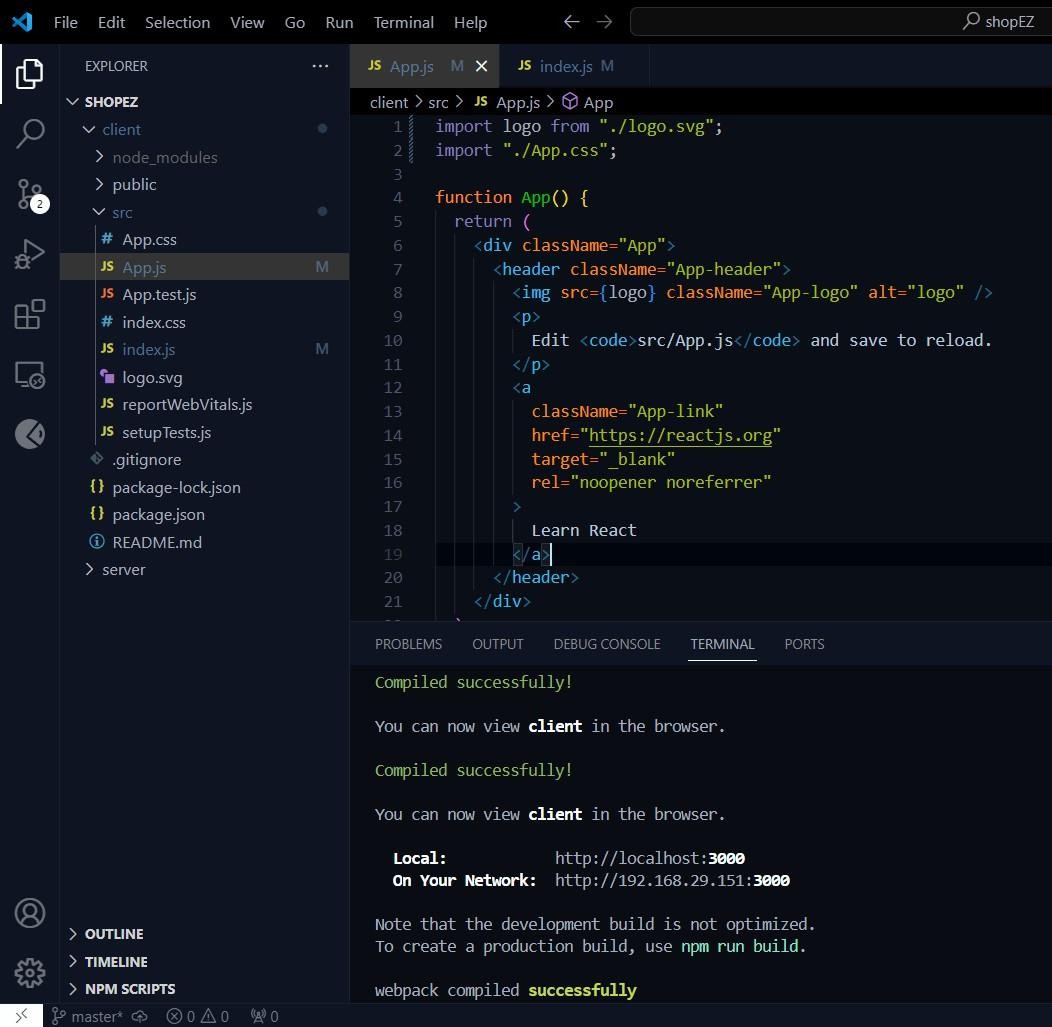
* + CreateComponents.
  + Implementlayout andstyling.
  + Addnavigation.

### Implementfrontendlogic:

* + IntegrationwithAPIendpoints.
  + Implement databinding.

Reference Video Link: <https://drive.google.com/file/d/1-CFhkyPbOypG2yTYT3rDo8LSCVMH8iTq/view?usp=drivesdk>

**ReferenceImage:**



## ProjectDevelopment:

### Createareduxstore:

* 1. import{configureStore}from"@reduxjs/toolkit";`:Thislineimportsthe

`configureStore`functionfromRedux Toolkit.ReduxToolkitis apackage that provides utilities to simplify Redux development, making it easier to write Redux logic with lessboilerplatecode.

* 1. `import{cryptoApi}from"../services/cryptoApi";`:Thislineimportsthe

`cryptoApi` object from the `cryptoApi.js` file located in the `../services` directory.

Thisobjectlikelycontainsconfigurationsandfunctionsrelatedtomaking API requests for cryptocurrency data.

* 1. ‘exportdefaultconfigureStore({...});`:ThislineexportstheRedux store configuration created by the `configureStore’ function as the default export of this module.
  2. `reducer:{[cryptoApi.reducerPath]:cryptoApi.reducer}`:Thispart ofthe configuration specifies the rootreducer for the Redux store. In this case, it sets the

`cryptoApi.reducer`asthereducerforthesliceofstatemanagedby the

`cryptoApi` API slice. The `cryptoApi.reducerPath` likely refers to the slice'suniqueidentifier, which is used internally by Redux Toolkit.

* 1. `middleware: (getDefaultMiddleware) => getDefaultMiddleware().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 variouspurposes,suchaslogging,asynchronousactions,orhandlingAPI requests. Here, it uses the

`getDefaultMiddleware` function provided by Redux Toolkit to get the default middlewarestackandappendsthe`cryptoApi.middleware`.

Thismiddlewarelikely handles asynchronous API requests and dispatches corresponding actions based on the API response.

this configuration sets up a Redux store with a specific reducer and middlewareprovided by the `cryptoApi` object, which presumablymanagesstaterelated to cryptocurrency data fetched from an external API. This setup allows you to manage and interact with this data using Redux within your React application.



,andqueryfunctionsrequiredformakingrequeststothecryptocurrency API.

## CreateaAPIsliceusingReduxtoolkit’s:

* 1. ImportStatements:

-`import { createApi, fetchBaseQuery } from "@reduxjs/toolkit/query/react";`:Thislineimportsthenecessary functions from Redux Toolkit's query-related module.

`createApi` is used to create an API slice, while `fetchBaseQuery` is a utility function provided by Redux Toolkit for making network requests using `fetch`.

* 1. HeaderandBaseURLConfiguration:

`constcryptoApiHeaders={...}`:Thisobjectcontainsheadersrequired for making requests to the cryptocurrency API. The values for `X- RapidAPI-Key"` and

`"X-RapidAPI-Host"`areretrievedfromenvironmentvariablesusing

`import.meta.env`.

-`constbaseUrl=[https://coinranking1.p.rapidapi.com](https://coinranking1.p.rapidapi.com/)`:Thisvariable holds the base URL for the cryptocurrency API, which is also retrieved from environment variables.

* 1. RequestCreationFunction:

‘constcreateRequest=(url)=>({url,headers:cryptoApiHeaders});`:This function.

`createRequest` takes a URL and returns an object with the URL and headers required for making a request. It utilizes `cryptoApiHeaders` to include necessary headers intherequest.

* 1. CreateAPISlice:

`exportconstcryptoApi=createApi({...})`:Thispartusesthe

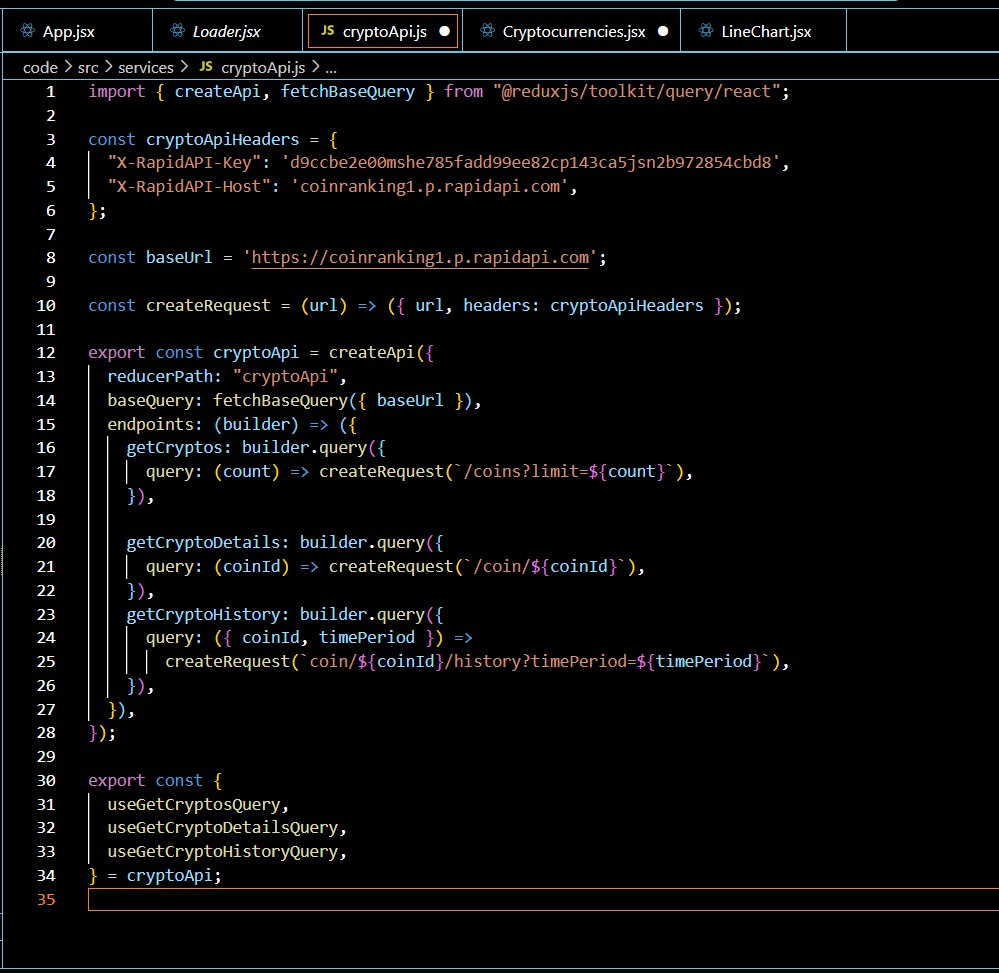
`createApi` function to create an API slice named `cryptoApi`. It takes an object with several properties:

* + - `reducerPath`:Specifiesthepathunderwhichtheslice'sreducerwill be mounted in the Redux store.
    - `baseQuery`:ConfiguresthebasequeryfunctionusedbytheAPI slice. In this case, it uses `fetchBaseQuery` with the base URL specified.
    - `endpoints`:DefinestheAPIendpointsavailableintheslice.It'san object with keys corresponding to endpoint names and values being endpoint definitions.
  1. APIEndpoints:
     + `getCryptos`, `getCryptoDetails`, `getCryptoHistory`: These areendpoints defined using the `builder.query` method. Each endpoint is configured with a `query` function that returns the request configuration object created by `createRequest`.
  2. ExportingHooks:
     + `exportconst{...}`:Thislineexportshooksgeneratedbythe

`createApi`function,allowingcomponentstoeasilyfetchdatafromthe API slice. Each hook corresponds to an endpoint defined in the

`endpoints`object.

Overall, this code sets up an API slice named `cryptoApi` using Redux Toolkit's query functionality. It defines endpoints for fetching cryptocurrencies, cryptocurrency details, and cryptocurrency history.



## AddingProvidersinthemainfunction:

ReactRouterwithBrowserRouter:

* + `<BrowserRouter>`:Thiscomponentisprovidedby`react-router- dom` and enables client-side routing using the HTML5 history API. It wraps the application, allowing it to use routing features.

ReduxProvider:

-

* + `<Providerstore={store}>`:Thiscomponentisprovidedby`react- redux` and is used to provide the Redux store to the entire application. It wraps the application, allowing all components toaccess the Redux store.

Overall, thiscodeinitializestheReactapplicationbyrenderingtheroot component (`<App />`) into the DOM, while also providing routing capabilities through

`BrowserRouter`andstatemanagementwithReduxthrough`Provider`. Additionally, it ensures stricter development mode checks with ‘<React.StrictMode>’.



## CreatingaLinechartcomponent:

ThiscodedefinesaReactcomponentcalled`LineChart`whichrendersa line chart usingthe`react-chartjs-2` library.

1. Imports:

importReactfrom"react":Importsthe`React` module.

import{Line}from"react-chartjs-2";`:Importsthe`Line`component from the`react-chartjs-2` library, which is used to render line charts.

import{Col,Row,Typography}from"antd";`:Importsspecific components from the Ant Design library, including `Col`, `Row`, and

`Typography.const{Title}=Typography;`:Destructuresthe`Title` component from the`Typography` module.

1. ComponentDefinition:

constLineChart=({coinHistory,currentPrice,coinName})=>{...}`: Definesa functional component called `LineChart`. It takes three props: `coinHistory`,currentPrice`, and `coinName`.

1. DataPreparation:
   * Insidethecomponent, itloopsthrough the`coinHistory`datatoextract

`coinPrice`and`coinTimestamp`arrays.Thesearrayswillbeusedas data points for the line chart.

1. ChartData:

`constdata={...}`:Definesthedataobjectforthelinechart.It includes labels (timestamps) and datasets (coin prices).

1. Rendering:
   * Insidethereturnstatement,itrendersthechartheader,includingthe coin name, price change, and current price.
   * `Row`and`Col`fromAntDesignareusedtostructurethelayout.
   * The`Line`componentrenderstheactuallinechartusingthedata object defined earlier.
2. Export:
   * `exportdefaultLineChart;`:Exportsthe`LineChart`componentas the default export.

Overall,thiscomponentreceiveshistoricaldata(`coinHistory`), current price (`currentPrice’), and the name of the cryptocurrency (`coinName’) as props, and renders a line chart displaying the historical price data. It also includes additional information such as the price change and current price displayed above the chart.



# Creatingcryptocurrenciescomponent:

1. ComponentDefinition:

`constCryptocurrencies=({simplified})=>{...}`:Definesafunctional component named `Cryptocurrencies`. It accepts a prop named

`simplified`,whichdetermineswhethertodisplayasimplifiedversionof the list.

1. Initialization:

‘constcount=simplified?10:100:Initializes the`count`variablebased on the value of the `simplified` prop. If `simplified` is true, `count` is set to 10; otherwise, it's set to 100.

1. FetchingCryptocurrencyData:

`const{data:cryptosList,isFetching}=useGetCryptosQuery(count);`: Uses the

`useGetCryptosQuery` hook provided by the `cryptoApi’ service tofetch cryptocurrency data. It retrieves the list of cryptocurrencies (`cryptosList`) and a boolean flag (`isFetching`) indicating whether the data is being fetched.

1. FilteringCryptocurrencyData:
   * The`useEffect` hookisusedto filter thecryptocurrency databasedon the

`searchTerm` state variable. It updates the `cryptos` state with filtereddatawhenever`cryptosList`or`searchTerm`changes.

1. RenderingLoader:
   * `if(isFetching) return <Loader />;`: Ifdatais still being fetched (`isFetching` is true), it returns a `Loader` component to indicate that the data is loading.
2. RenderingSearchInput:
   * `!simplified&&(...)`:If`simplified`isfalse,itrendersasearchinput fieldallowinguserstosearchforspecificcryptocurrenciesbyname.
3. RenderingCryptocurrency Cards:
   * The`Row`and`Col`componentsfromAntDesignareusedtocreate a grid layout for displayingcryptocurrency 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.
4. ReturnStatement:
   * `return(...)`:ReturnsJSXrepresentingthecomponent'sstructureand content.

Overall, this component fetches cryptocurrency data, filters it based on a searchterm, andrenders the data in a visually appealingformat withcard- based UI. It also provides a search functionality for users to find specific cryptocurrencies.

.

## Createacomponenttoshowthedetailsofcryptocurrency:

ThiscodedefinesaReactfunctionalcomponentcalled`CryptoDetails’ responsible fordisplaying detailed information about a specific cryptocurrency. Let's break down the code:

* 1. ComponentDefinition:

constCryptoDetails=()=>{...}`:Definesafunctionalcomponentnamed CryptoDetails`. It doesn't accept any props directly but utilizes React

Router'suseParams`hooktogetthe`coinId`parameterfromtheURL.

* 1. StateInitialization:
     + Initializesstatevariables`timePeriod`and`coinHistory`.

`timePeriod`represents the selected timeperiod for displaying cryptocurrencyhistory,and`coinHistory`storeshistoricaldataofthe selected cryptocurrency.

* 1. FetchingData:

Utilizes`useGetCryptoDetailsQuery’and‘usGetCryptoHistoryQuery’ hooks provided bythe `cryptoApi` service to fetchdetails and historicaldataofthe cryptocurrencyspecifiedby`coinId`.Ituses ‘coinId`obtainedfrom`useParams`to fetch data for the specific cryptocurrency.

* 1. SettingCoinHistory:

Utilizes `useEffect` hook to update the `coinHistory` state when coinHistoryData`changes.Thisensuresthatthecomponentre-renderswith updated historical data.

* 1. RenderingLoader:
     + Displaysaloadingindicator(`<Loader/>`)whiledataisbeingfetched (`isFetching` is true).
  2. TimePeriodSelection:

Renders a `Select` component allowing users to choose the time periodfor displayinghistorical data. It triggers the `setTimePeriod` function when the selection changes.

* 1. RenderingLineChart:
     + Utilizesthe`LineChart`componenttodisplaythehistoricalprice trend ofthecryptocurrencyovertheselectedtimeperiod.
  2. RenderingStatistics:

Displaysvariousstatisticsrelatedtothecryptocurrency,suchasprice,rank, volume,market cap, etc. Thesestatistics are displayedintwo sections:

`stats`and‘genericStats`.

* 1. RenderingDescriptionandLinks:
     + Parsesanddisplaysthedescriptionofthecryptocurrencyusing

`HTMLReactParser`.

* + - Renderslinksrelatedtothecryptocurrency,suchasofficialwebsites, social media, etc.
  1. ReturnStatement:
     + ReturnsJSXrepresentingthestructureandcontentofthecomponent.

Overall, this component fetches and displays detailed informationabout a specific cryptocurrency, including historical price data, key statistics, description, and related links.

# CreateaHomepage:

This component, named `Home`, is a React functional componentresponsiblefor rendering the home page ofthe cryptocurrency dashboard. Let's break down the code:

* 1. ComponentDefinition:

`constHome=()=>{...}`:Definesafunctionalcomponentnamed

`Home`.

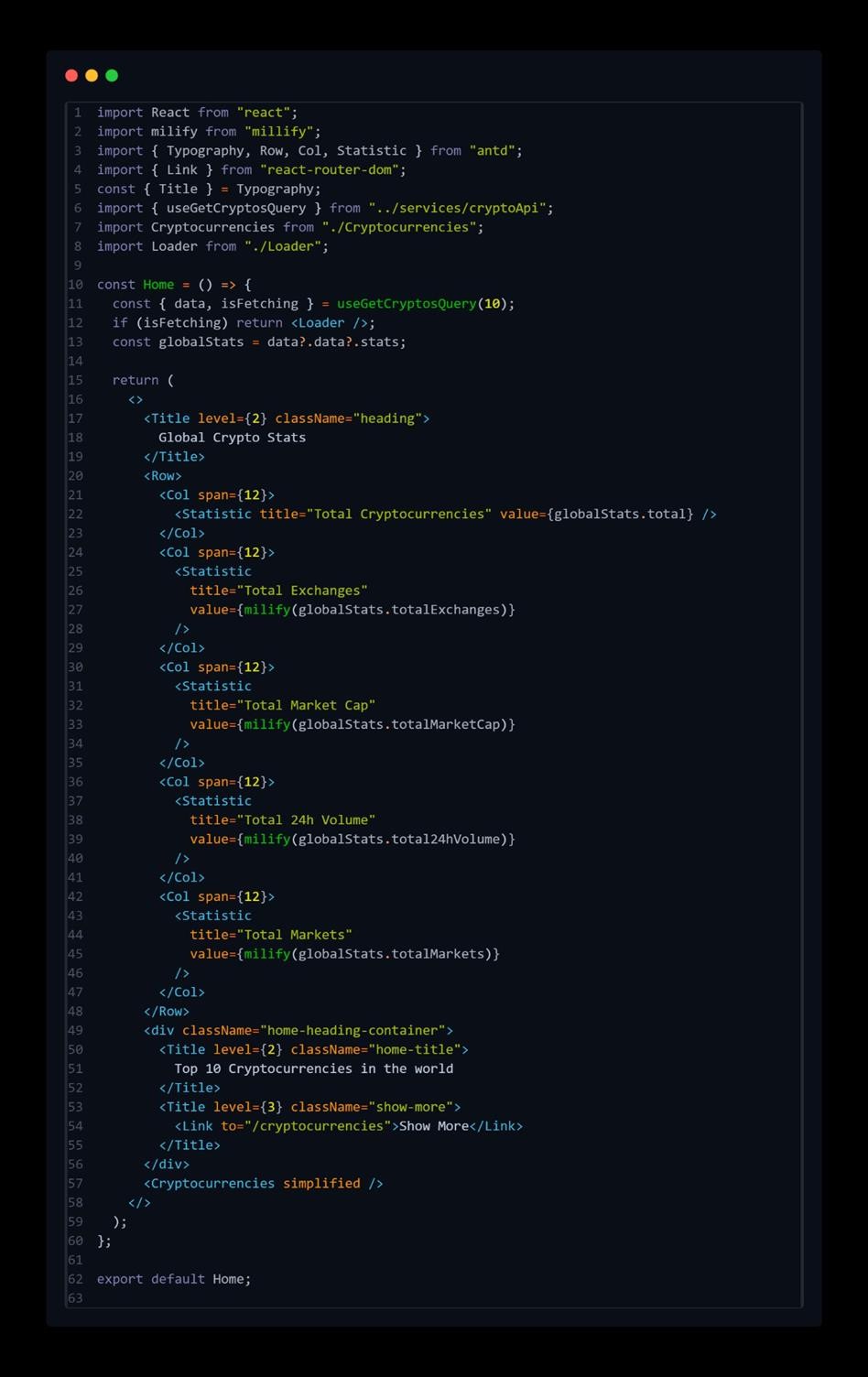
* 1. DataFetching:
     + Uses the `useGetCryptosQuery` hook provided by the `cryptoApi` service to fetch data for the top 10 cryptocurrencies. It retrieves data and a boolean flag indicatingwhetherdataisbeingfetched.
  2. RenderingLoader:
     + Displaysaloadingindicator(`<Loader/>`)whiledataisbeingfetched (`isFetching` is true).
  3. GlobalCrypto Stats:

Renders statistics about the global cryptocurrency market,including total cryptocurrencies, total exchanges, total market cap, total 24-hour volume, and total markets. These statistics are displayedusingthe `Statistic`componentfromAntDesign.

* 1. Top10Cryptocurrencies:
     + Rendersasectiondisplayingthetop10cryptocurrenciesintheworld.
     + Utilizes the `Cryptocurrencies` component with the `simplified` prop settotrueto display a simplified version of the list.
     + Providesalinktoviewmorecryptocurrenciesusingthe`Link` component from React Router.
  2. ReturnStatement:
* ReturnsJSXrepresentingthestructureandcontentofthecomponent.

Overall,thiscomponentfetchesanddisplaysglobal cryptocurrency

statisticsandthe top 10 cryptocurrencies on the homepage of the dashboard. It provides links for users to navigate to the full list of cryptocurrencies.



**ProjectExecution:**

### Hereisthevideolinkofreactapplicationexecution:

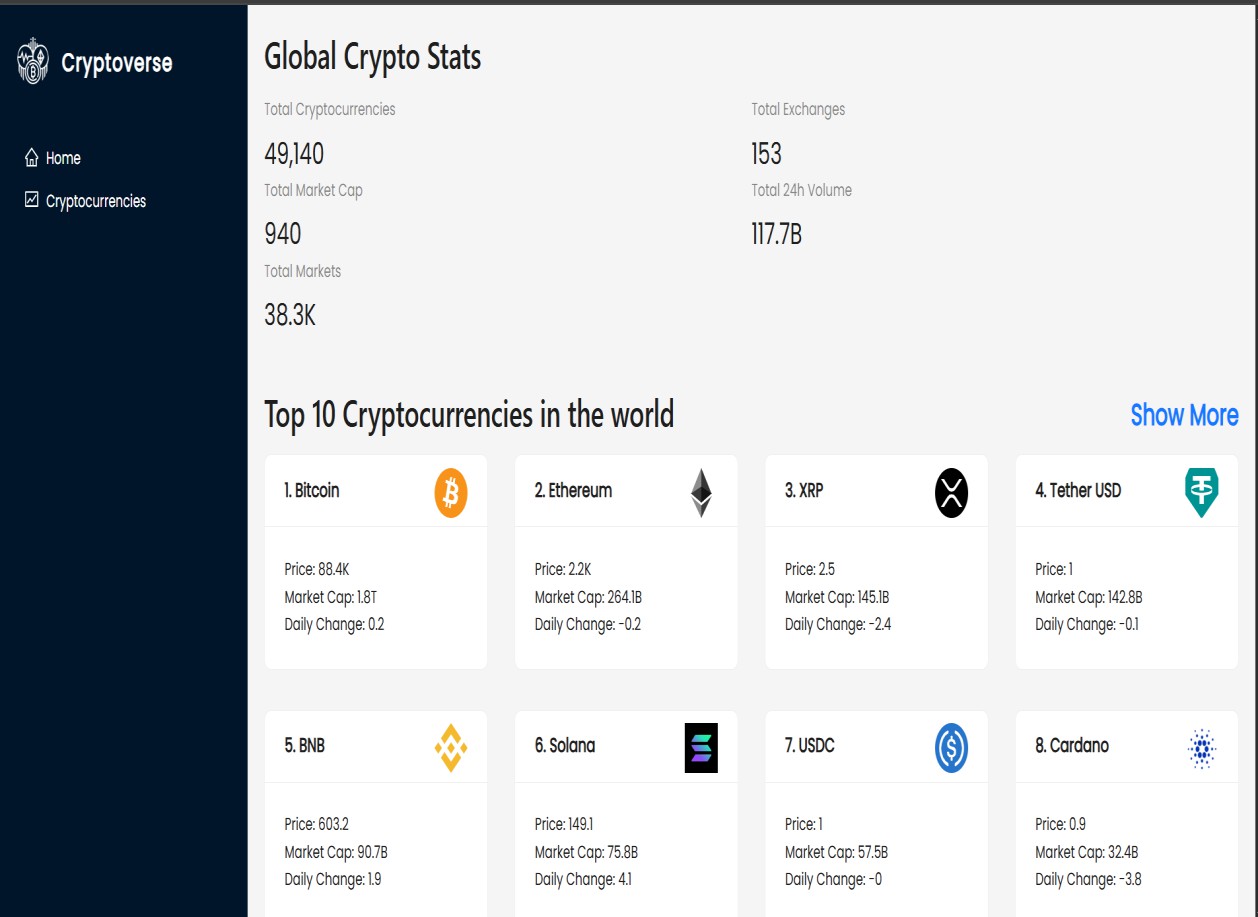
<https://drive.google.com/file/d/1-CFhkyPbOypG2yTYT3rDo8LSCVMH8iTq/view?usp=drivesdk>

**Projectdemo:**

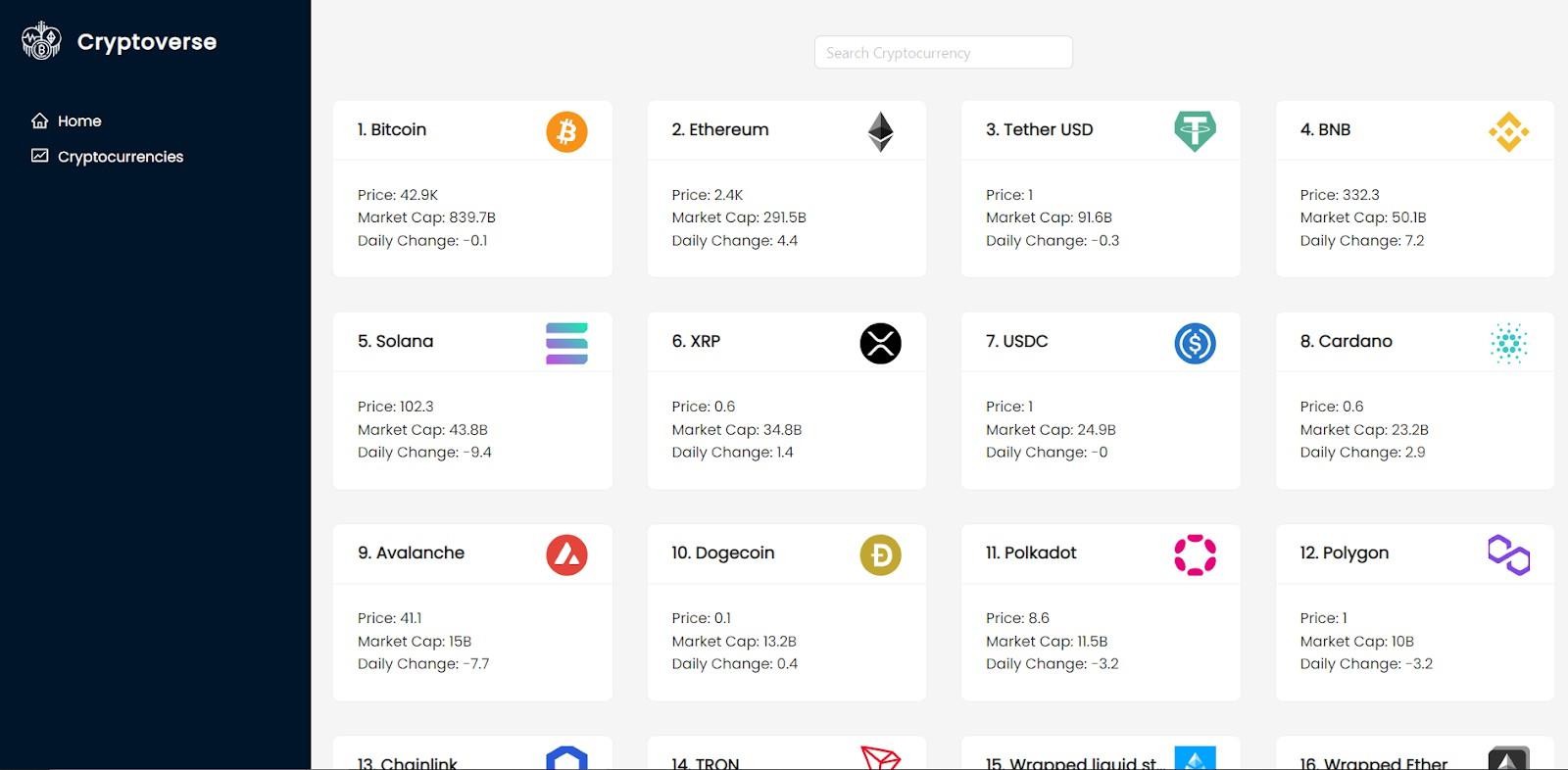
Demo link: <https://drive.google.com/file/d/1-CFhkyPbOypG2yTYT3rDo8LSCVMH8iTq/view?usp=drivesdk>

## UserInterfacesnips:

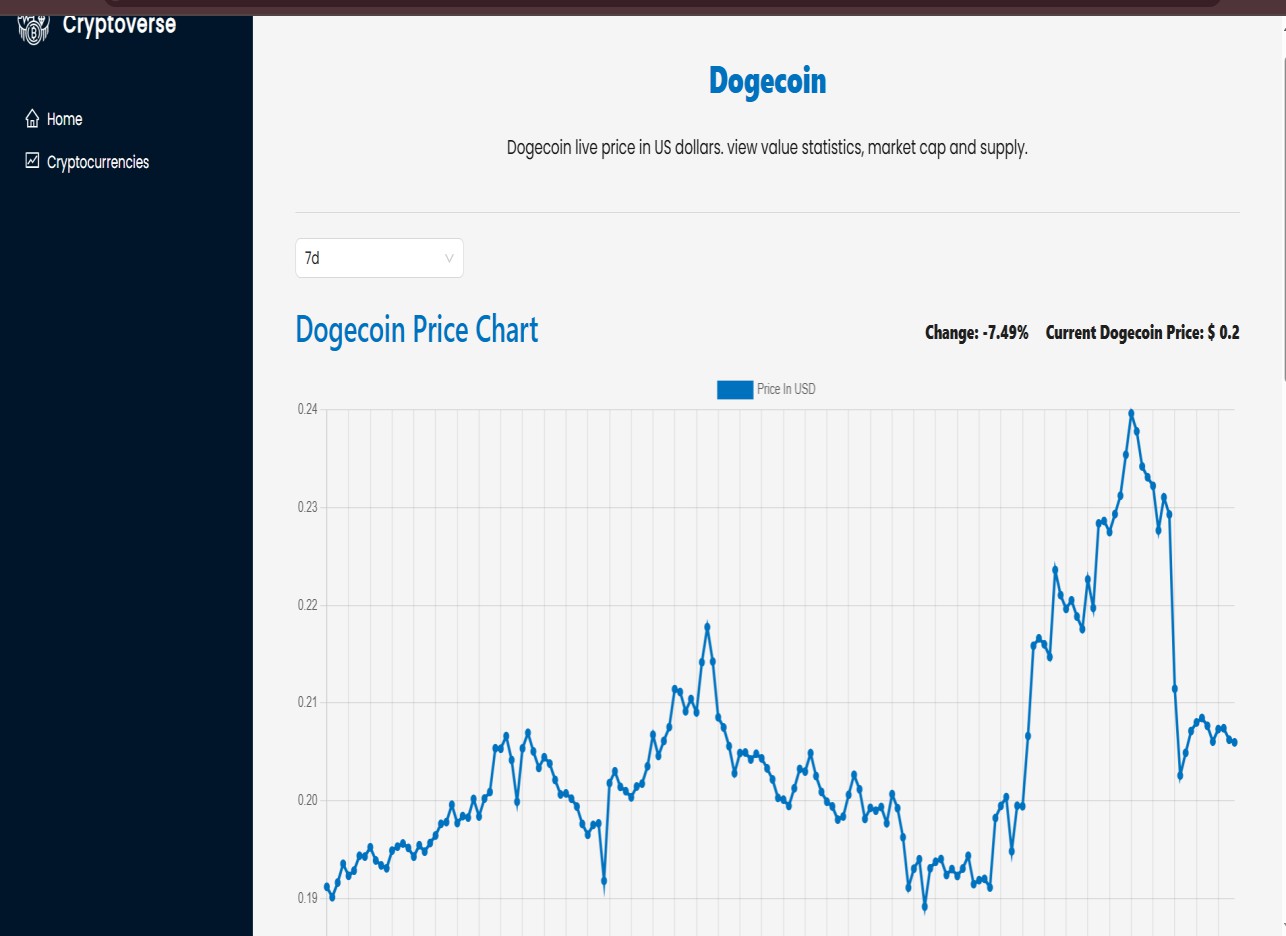
Homepage:Thispagesconsistsofstatsofglobalcryptoliketotalcryptocurrencies, total exchanges, market cap etc. Also consist of top 10 cryptocurrencies in the world.

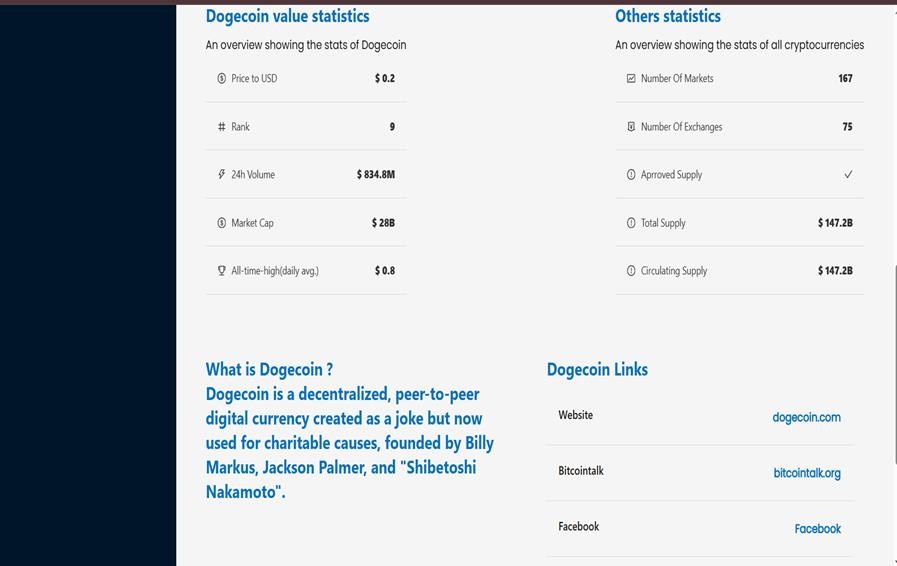


Crypto currencies page: This pages contains all cryptocurrencies which are currently inflow in the world. There is also a search feature where users can searchand findout about their desiredcryptocurrency.



This page contains the line chart with data representation of price of cryptocurrencies.Alsocontainsstatisticsandwebsitelinksof cryptocurrencies.





THANKYOU