React and Redux

Links:

https://redux.js.org/introduction/getting-started
https://react-redux.js.org/introduction/getting-started
https://app.egghead.io/playlists/fundamentals-of-redux-course-from-dan-abramov-bd5cc867
https://www.educative.io/courses/practical-redux
https://www.freecodecamp.org/news/how-to-use-redux-in-reactjs-with-real-life-examples-687ab4441b85/

STEPS:

When using template:

1. npx create-react-app my-app --template redux
2. npm install --save @reduxjs/toolkit redux react-redux react-router-dom
3. Do clean up when working with template and do folder and file creation

When manual setup is followed:

1. npx create-react-app first-redux
2. cd first-redux

3. npm install --save redux
4. npm install --save @reduxjs/toolkit
5. npm install --save redux-thunk
6. npm install --save react-router-dom
7. code .
8. npm start
9. Do clean up
10. Create following folders and files:

    assets

    themes

        index.css
        App.css
        Other css files needed

    routing

        AllRoutes.js
        Routing.jsx

    pages

        Home.jsx
        ...All other pages

    components

        Header.jsx
        Footer.jsx
        NavBar.jsx
        Menu.jsx

redux

- Store.js
- ReducerName.js
- ActionName.js

firebase (if using)

- FirebaseConfig.js
- FirebaseAuth.js
- FirebaseDB.js (realtime/firestore)

Please use Lesson 20 and 21 for basic setup and structure.

Notes:

Redux: a Global state object accessed by our entire project but can only allow state mutations with a action that is in a reducer that gets triggered by a dispatch event.

Rember i our freddy and cake analogy

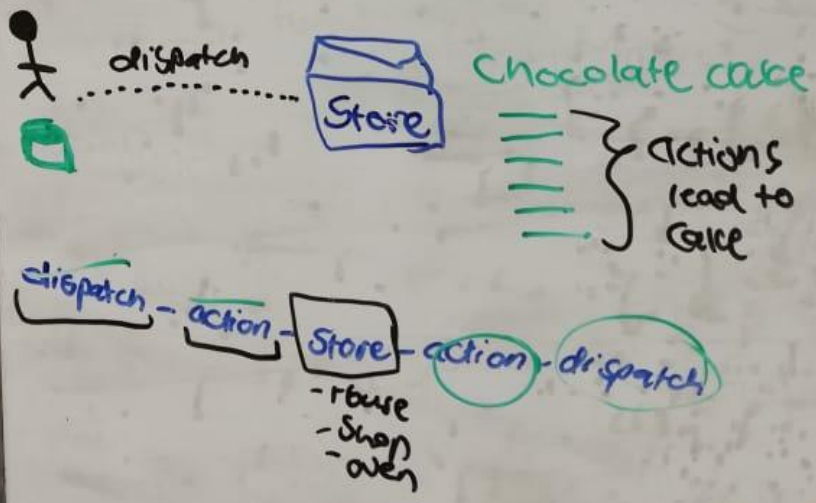
useState()
↳ That Componet

useAuth() Hook
↳ Curr User → Set Locally
But accessed Globaly

Func Sign .....

const {.....} = useAuth()

Redux = Global Acessable State but
locally manipulated within
actions within reducers trigered
by disparches



As seen at the top of the image is how we got to this exact lesson.
We started out with a simple local state and learned how to work with it.
We then moved on to create our own custom hooks that gave us a global state variable that could only be changed from within the hook.

To where we now take a look at redux state management where we have a single global state that receives the intention of an action that needs to be performed from the dispatch to the reducer where it gets updated with in the reducer according to the action applied to it and then returned to our app through the dispatch. This leads to a circular flow within our application and a central point of data reception to which we can have access from every corner of our application.