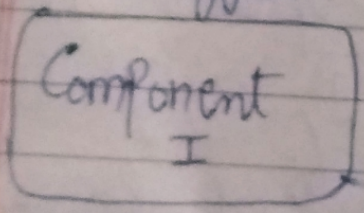
* Redux

→ It is a state-management library.

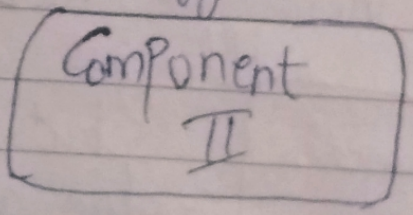
→ React comes in picture when we have way too many states in our project and a lot of components using those states.

Let's suppose, we have 3 components →

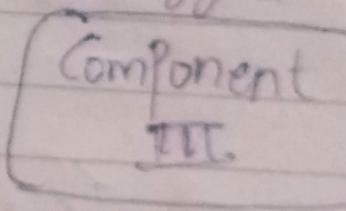
→ isLoggedin



isLogged



isLogged



Suppose, their is a state 'isLoggedIn' and right now it is true and the data shown on all 3 components is same now what if user goes on Component I does some changes and is loggedout. So, now how the other 2 components will get to know that the state of Compon

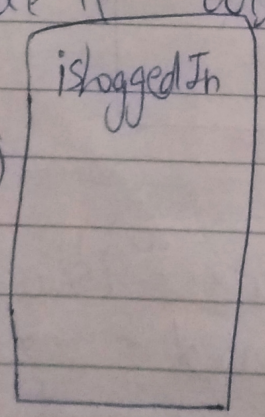
I is been changed.

1st option →

If we consider 'Component I' as parent and other two Components as child, so we can use the value of state using props in the child Components so that the other Components can have the updated value of that state.

* Now what if these three components are completely different components are not related to each other.

2nd option → we can put all the common states used by all the 3 components and keep them all in a separate file, and when the value gets updated for a state it will get updated in this file, but even if the value gets updated how will the other 2 components get to know that the value is been updated.



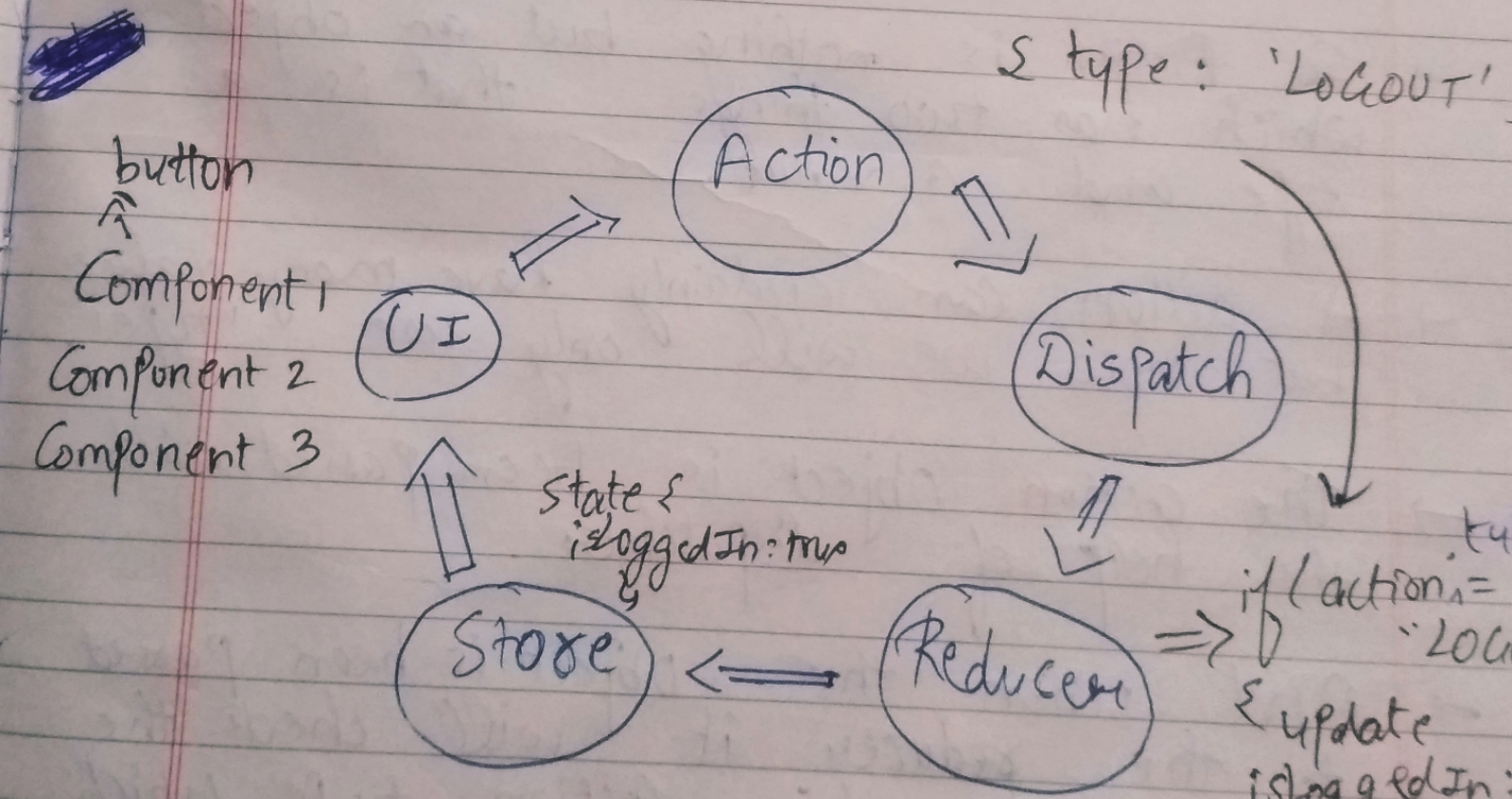
→ So, in order to achieve this "redux" comes in picture.

It does the same thing, it will provide us with a store where we will be having all states.

So, when a state gets updated inside a store & and which soever component is using that state will get to know and would function accordingly.

→ So, basically redux has 4 elements →

(1) Action Creator.
(2) Dispatcher.
(3) Reducers.
(4) Store.



→ So, in the previous diagram we are having 3 Components in our UI.

→ Component 1 is having a button which has a state 'IsloggedIn'.

→ And that state is stored in store with 'isLoggedIn' as true initially.

→ Reducer does the task of updating the state through the action passed to it.

→ Now, we try to connect the cycle and see how it works.

→ So, Action is nothing but an object which has two things, that is a type and a value.

→ A action can certainly have many values but now we will only use 'type'.

→ The action object is been passed with the help of dispatcher.

→ And when the object is been passed to the reducer it will check the Object with the action.type which we have in reducer.

→ And if it matches then we go

ahead and update our value in the store.

→ And once the value is been updated it is been passed as props in all the components.