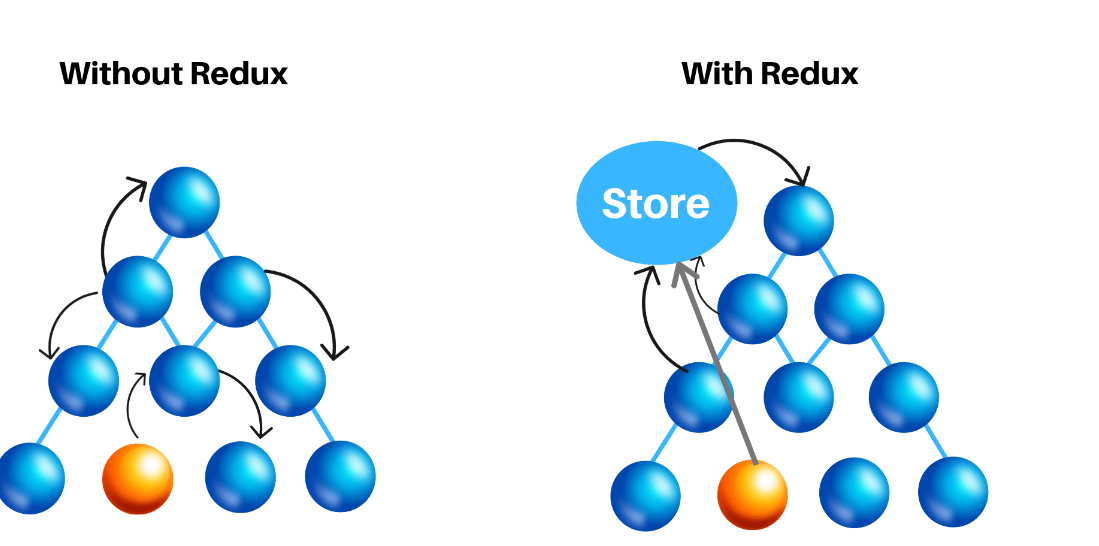
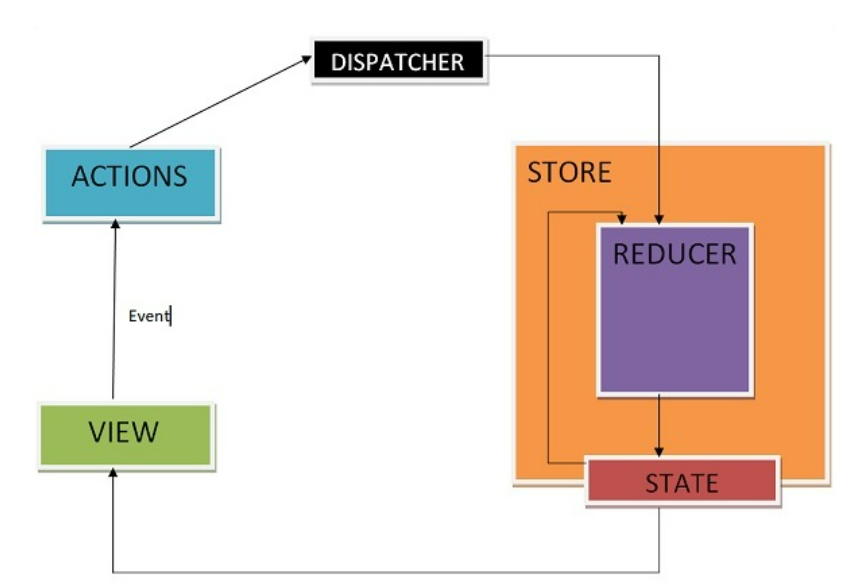
REACT-REDUX USING HOOKS (For Functional components):  
  
=> Redux is among the most popular cross-platform and open-source JavaScript library and is inspired by Facebook’s Flux architecture. It was launched on June 2, 2015.  
  
=>The main feature of Redux is to manage the application state. when you are dealing with a large application, it becomes difficult to manage the state of each component of your application. Redux helps overcome this problem and allows the developer to maintain the state of each component. As a result, you can easily trace when, where, why, and how a particular component has been changed.  
  
  
  
  
  
  
  
Advantages of Redux:  
=> Centralized state management system (global store).  
=> Performance optimization (Avoid re-rendering and Renders only when its data changed).  
=> Avoid prop-drilling.  
=> Pure reducer function.  
=> Storing long term data.  
  
  
Flow of Redux:  
  
  
  
  
Action: These are objects that should have two properties, one describing the **type** of action, and one describing what should be changed in the app state.  
  
  
  
Dispatch: It is used to dispatch actions and trigger state changes to the store.  
Reducer: It is pure function. They change the state of the app, based on the action description and the state change description.  
  
Store: it brings the actions and reducers together, holding and changing the state for the whole app — there is only one store.  
  
  
Example: -  
  
View -   
  
import {useDispatch} from 'react-redux'  
import { useSelector } from 'react-redux'  
  
const getTitleVal = useSelector((state) => state.userReducer.userData)  
  
<button onClick={handleTitleClick}>get Titles</button>  
let handleTitleClick = () => {

dispatch(actionCreator.getAPICallData(myRef.current.value))  
}  
  
Action-creator.js -  
  
export function getAPICallData(id){

return(dispatch)=>{

getapi.getCall(id).then((res)=>{

   dispatch({type:action.GETCALL,payload:res.data})

})

}

}

state.js -  
  
const gstate = {

userData:{

          }

}

Reducer.js -  
  
  
let userReducer= (state=gstate,action)=>{

  if(action.type=actions.GETCALL){

      return{

            ...state,

            userData: action.payload

        }

    }

}

export default userReducer;  
  
index.js -   
  
import { BrowserRouter } from 'react-router-dom';

import { Provider } from 'react-redux';

import thunk from 'redux-thunk';

import{combineReducers,compose,createStore,applyMiddleware}from 'redux';

import userReducer from './state-management/Reducers/userreducer';

let rootreducer = combineReducers({

  userReducer:userReducer

})

const composeEnhancer = window.\_\_REDUX\_DEVTOOLS\_EXTENSION\_COMPOSE\_\_ || compose;

const store = createStore(rootreducer,composeEnhancer(applyMiddleware(thunk)))

const root = ReactDOM.createRoot(document.getElementById('root'));

root.render(

  <React.StrictMode>

    <BrowserRouter>

    <Provider store={store} >

    <App />

    </Provider>

  </BrowserRouter>

   </React.StrictMode>

);