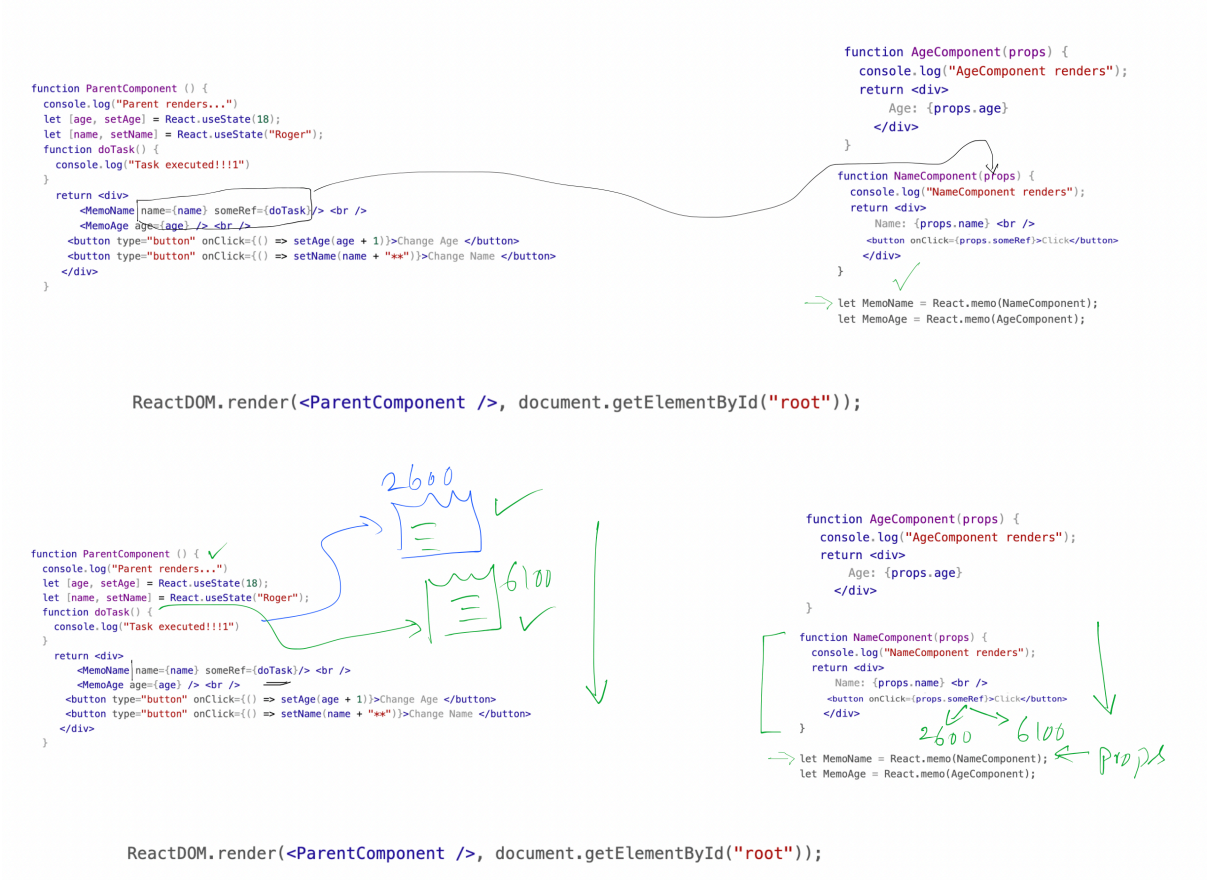
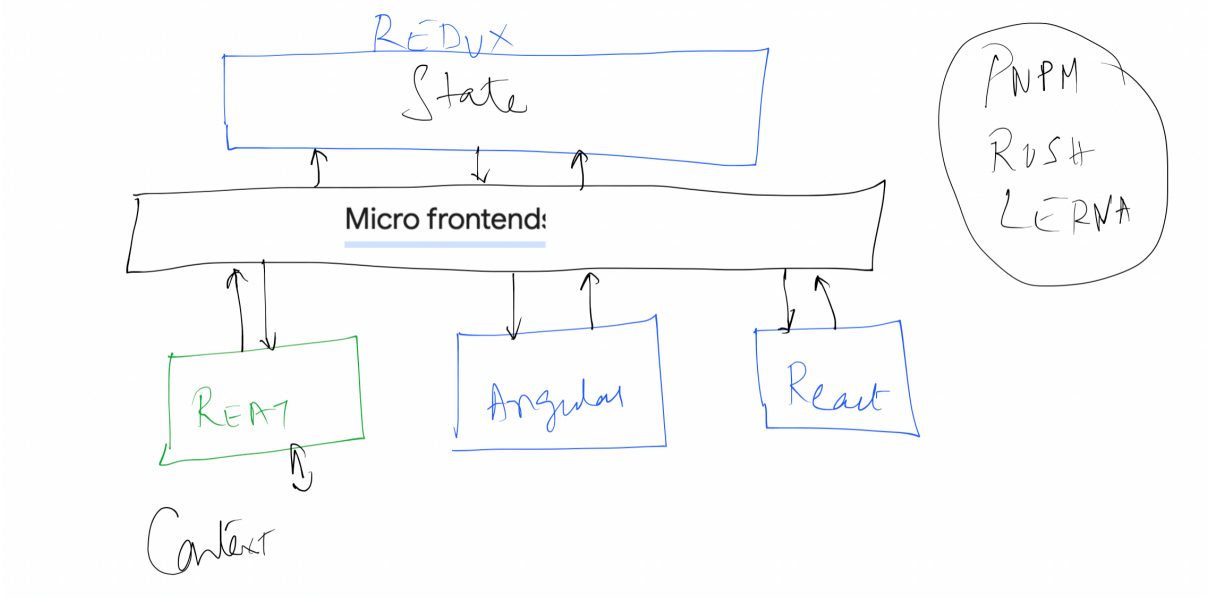
In spite of memorizing the NameComponent still it is re-rendering when any state changes in ParentComponent

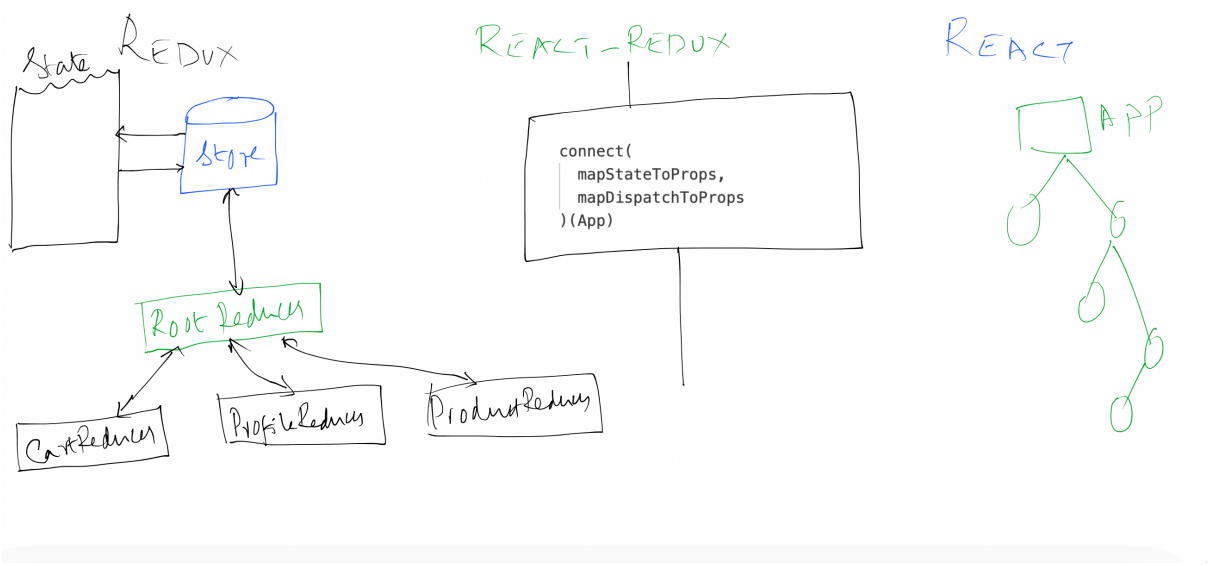


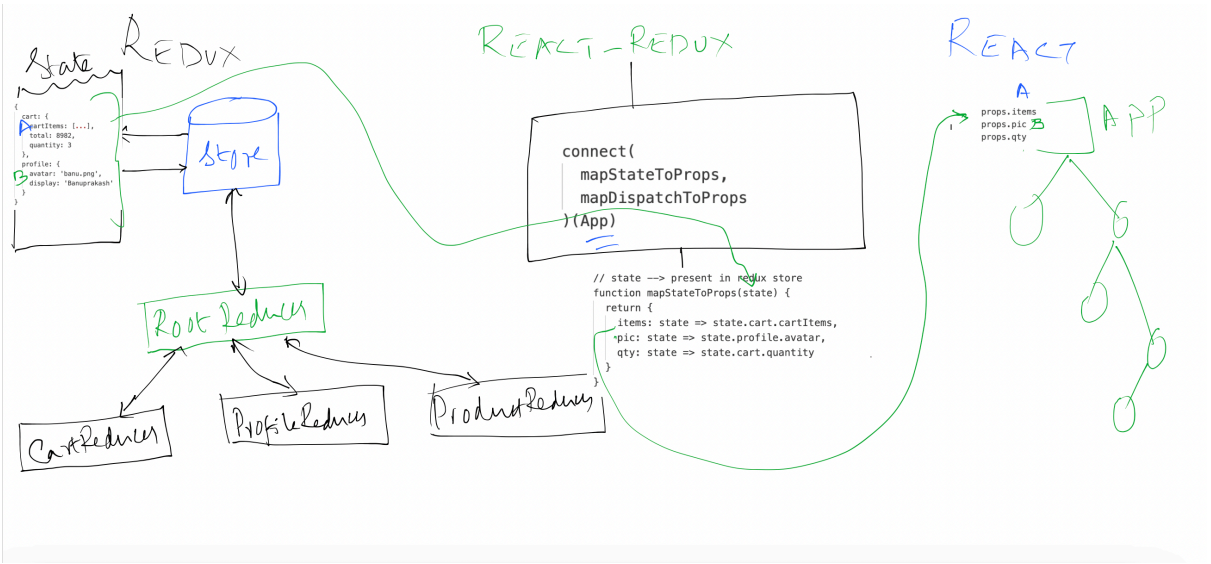
Solution: useCallback() hook to memorize function definition

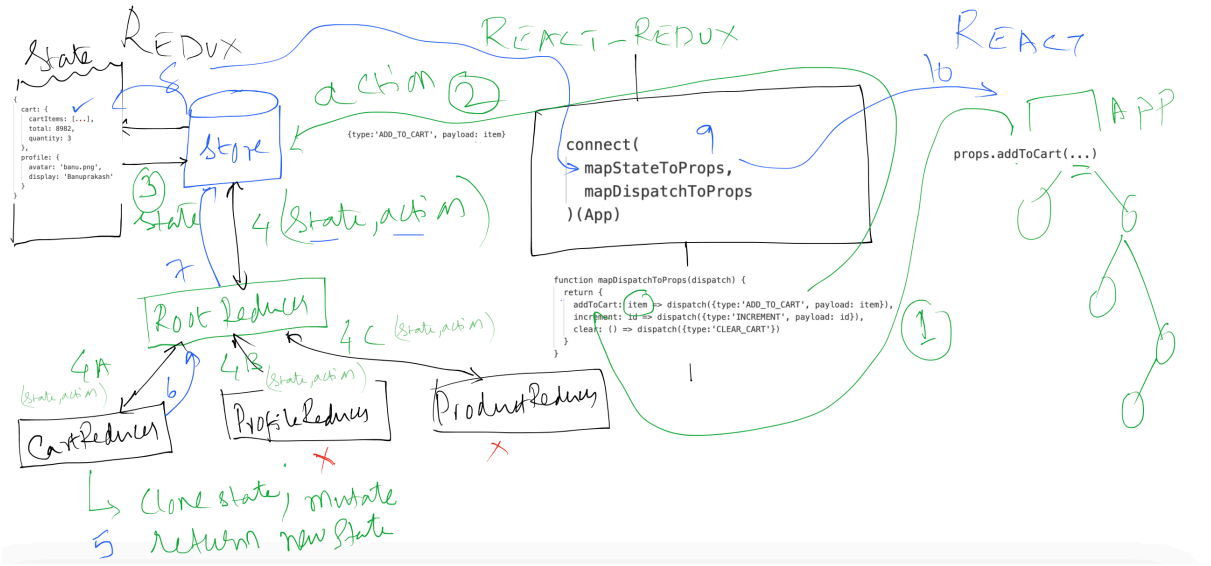
```
function ParentComponent () {
 console.log("Parent renders...")
 let [age, setAge] = React.useState(18);
 let [name, setName] = React.useState("Roger");

 const doTask = React.useCallback(() => function() {
  console.log("Task executed!!!1")
 }, []);

  return <div>
    <MemoName name={name} someRef={doTask}/> <br />
    <MemoAge age={age} /> <br />
   <button type="button" onClick={() => setAge(age + 1)}>Change Age </button>
   <button type="button" onClick={() => setName(name + "**")}>Change Name
</button>
   </div>
 }


function AgeComponent(props) {
```

```
  console.log("AgeComponent renders");
  return <div>
     Age: {props.age}
    </div>
}

function NameComponent(props) {
  console.log("NameComponent renders");
  return <div>
     Name: {props.name} <br />
    <button onClick={props.someRef}>Click</button>
    </div>
}

let MemoName = React.memo(NameComponent);
let MemoAge = React.memo(AgeComponent);
   ReactDOM.render(<ParentComponent />, document.getElementById("root"));
```




State
REDUX
Store
Root Reducer
CartReducer
ProfileReducer
ProductReducer
REACT-REDUX
connect(
mapStateToProps,
mapDispatchToProps
)(App)
REACT
APP


State
REDUX
Store
Root Reducer
CartReducer
ProfileReducer
ProductReducer
REACT-REDUX
connect(
mapStateToProps,
mapDispatchToProps
)(App)
// state --> present in redux store
function mapStateToProps(state) {
return {
items: state => state.cart.cartItems,
pic: state => state.profile.avatar,
qty: state => state.cart.quantity
}
}
REACT
APP
props.items
props.pic
props.qty


State
REDUX
Store
Root Reducer
CartReducer
ProfileReducer
ProductReducer
REACT-REDUX
connect(
mapStateToProps,
mapDispatchToProps
)(App)
function mapDispatchToProps(dispatch) {
return {
addToCart: item => dispatch({type:'ADD_TO_CART', payload: item}),
increment: id => dispatch({type:'INCREMENT', payload: id}),
clear: () => dispatch({type:'CLEAR_CART'})
}
}
{type:'ADD_TO_CART', payload: item}
action
REACT
APP
props.addToCart(...)
Clone state, mutate
return new state