YouTube
NAVER 카페
구멍가게 코딩단

코드로 배우는 리액트

11. 리액트 장바구니 구성

11장. 리액트 장바구니 구성

- API 서버에 구성된 장바구니 관련 기능을 리액트를 이용해서 실제 화면을 구성
- 리덕스 툴킷을 이용해서 로그인 상황에 따라 장바구니에 상품을 추가하고 변경하는 등을 작업을 처리

개발목표

1. 리덕스 툴킷을 이용한 장바구니 상태 관리
2. API 서버와 장바구니 상태 동기화 처리

Index

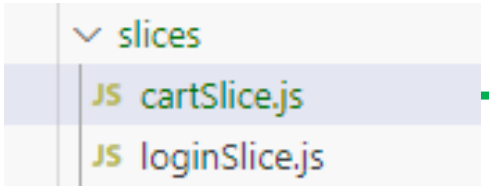
API 서버와 통신

→ api 폴더에 cartApi.js를 추가


```
import jwtAxios from "../util/jwtUtil"
import { API_SERVER_HOST } from "./todoApi"

const host = `${API_SERVER_HOST}/api/cart`

export const getCartItems = async ( ) => {

  const res = await jwtAxios.get(`${host}/items`)
  return res.data

}

export const postChangeCart = async (cartItem) => {

  const res = await jwtAxios.post(`${host}/change`, cartItem)
  return res.data
}
```

API 서버와 통신

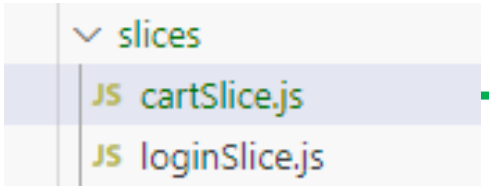
→ cartSlice의 작성 : cartSlice.js를 구성하고 store.js에 이를 추가




```
import { createAsyncThunk, createSlice } from "@reduxjs/toolkit";
import { getCartItems, postChangeCart } from "../api/cartApi";

export const getCartItemsAsync = createAsyncThunk('getCartItemsAsync', () => {
  return getCartItems()
})

export const postChangeCartAsync = createAsyncThunk('postCartItemsAsync', (param)
=> {
  return postChangeCart(param)
})

const initState = []
```

API 서버와 통신

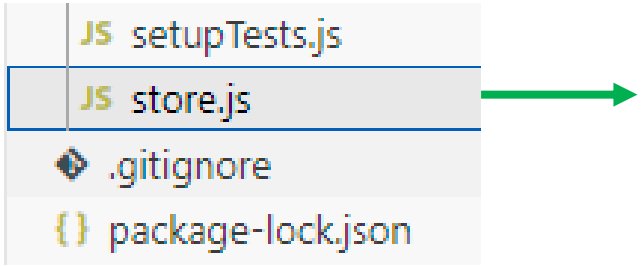
→ cartSlice의 작성 : cartSlice.js를 구성하고 store.js에 이를 추가




```
const cartSlice = createSlice({
  name: 'cartSlice',
  initialState: initState,
  extraReducers: (builder) => {
    builder.addCase(
      getCartItemsAsync.fulfilled, (state, action) => {
        console.log("getCartItemsAsync fulfilled")
        return action.payload
      }
    )
    .addCase(
      postChangeCartAsync.fulfilled, (state, action) => {
        console.log("postCartItemsAsync fulfilled")
        return action.payload
      }
    )
  }
})
export default cartSlice.reducer
```

API 서버와 통신

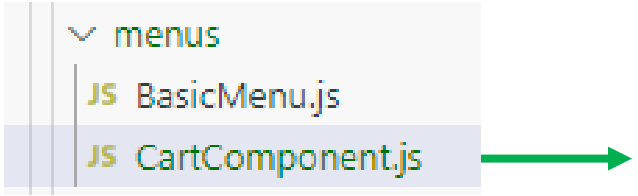
→ store.js에 cartSlice를 추가




```
import { configureStore } from '@reduxjs/toolkit'
import loginSlice from './slices/loginSlice'
import cartSlice from './slices/cartSlice'

export default configureStore({
  reducer: {
    "loginSlice": loginSlice,
    "cartSlice" : cartSlice
  }
})
```

장바구니용 컴포넌트

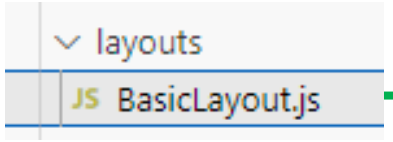
→ components/menus > CartComponent.js파일 추가




```
const CartComponent = () => {
  return (
  <div className="w-full">
    <div>Cart</div>
  </div>
  );
}

export default CartComponent;
```

장바구니용 컴포넌트

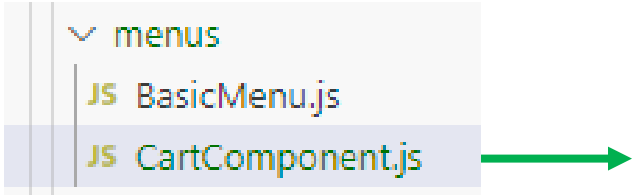
→ BasicLayout내부에서 기본 레이아웃의 일부로 사용되도록 수정




```
import BasicMenu from "../components/menus/BasicMenu";
import CartComponent from "../components/menus/CartComponent";

const BasicLayout = ({children}) => {
  return (
  <>
    {/* 기존 헤더 대신 BasicMenu*/ }
    <BasicMenu/>
    {/* 상단 여백 my-5 제거 */}
    <div className="bg-white my-5 w-full min-w-full flex flex-col space-y-1 md:flex-row md:space-
x-1 md:space-y-0">
      {/* 상단 여백 py-40 변경 flex 제거 */}
      <main className="bg-sky-300 md:w-2/3 lg:w-3/4 px-5 py-5 ">{children}</main>
      {/* 상단 여백 py-40 제거 flex 제거 */}
      <aside className="bg-green-300 md:w-1/3 lg:w-1/4 px-5 flex py-5"><CartComponent/></aside>
    </div>
  </>
  );
}

export default BasicLayout;
```

장바구니용 컴포넌트

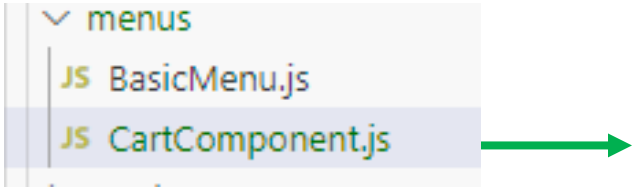
→ 로그인 상태 체크와 장바구니




```jsx
import useCustomLogin from "../../hooks/useCustomLogin";

const CartComponent = () => {

  const {isLogin, loginState} = useCustomLogin();

  return (
  <div className="w-full">
    {isLogin ?
    <div> {loginState.nickname}'s Cart</div>
    :
    <div></div>
    }
  </div>
  );
}

export default CartComponent;
```

장바구니용 컴포넌트

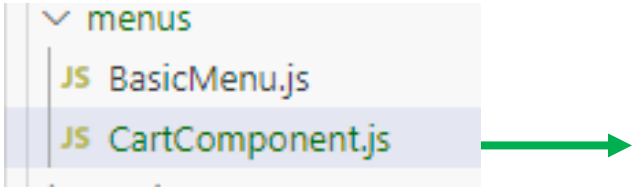
→ 장바구니 아이템 가져오기




```
import { useEffect } from "react";
import useCustomLogin from "../../hooks/useCustomLogin";
import { getCartItemsAsync } from "../../slices/cartSlice";
import { useDispatch } from "react-redux";

const CartComponent = () => {
  const {isLogin, loginState} = useCustomLogin();
  const dispatch = useDispatch()
  useEffect(() => {
    if(isLogin) {
      dispatch(getCartItemsAsync())
    }
  },[isLogin])
  return (
  <div className="w-full">
    {isLogin ? <div> {loginState.nickname}'s Cart</div> : <div></div> }
  </div>
  );
}
export default CartComponent;
```

장바구니용 컴포넌트

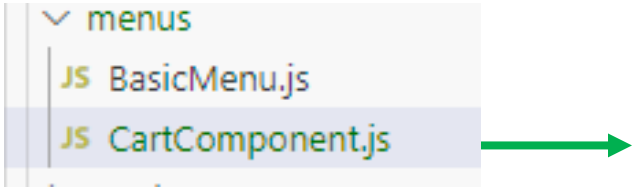
→ 장바구니 아이템 개수 가져오기




```
import { useEffect } from "react";
import useCustomLogin from "../../hooks/useCustomLogin";
import { getCartItemsAsync } from "../../slices/cartSlice";
import { useDispatch, useSelector } from "react-redux";

const CartComponent = () => {

  const {isLogin, loginState} = useCustomLogin();

  const dispatch = useDispatch()

  const cartItems = useSelector(state => state.cartSlice)

  useEffect(() => {
    if(isLogin) {
      dispatch(getCartItemsAsync())
    }
   },[isLogin])
```

장바구니용 컴포넌트

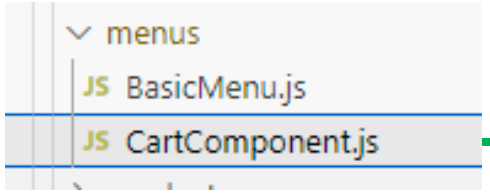
→ 장바구니 아이템 개수 가져오기




```
  return (
  <div className="w-full">
    {isLogin ?
    <div className="flex">
      <div className="m-2 font-extrabold">
        {loginState.nickname}'s Cart
      </div>
      <div className="bg-orange-600 w-9 text-center text-white font-bold
rounded-full m-2">{cartItems.length}</div>
    </div>
    :
    <div></div>
    }
  </div>
  );
}

export default CartComponent;
```

장바구니용 컴포넌트

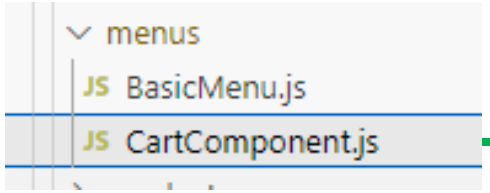
→ CartComponet는 작성된 useCustomCart를 이용하도록 수정




```
import { useEffect } from "react";
import useCustomLogin from "../../hooks/useCustomLogin";
import useCustomCart from "../../hooks/useCustomCart";

const CartComponent = () => {

  const {isLogin, loginState} = useCustomLogin()

  const {refreshCart, cartItems} = useCustomCart()

  useEffect(() => {

    if(isLogin) {

      refreshCart()
    }

  },[isLogin])
```

장바구니용 컴포넌트

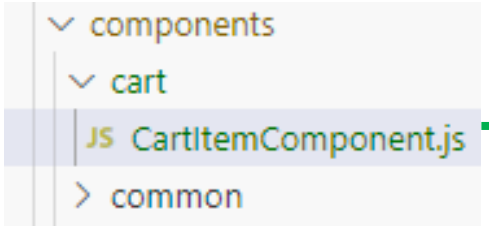
→ CartComponet는 작성된 useCustomCart를 이용하도록 수정




```
  return (
  <div className="w-full">
    {isLogin ?

    <div className="flex">
      <div className="m-2 font-extrabold">
        {loginState.nickname}'s Cart
      </div>
      <div className="bg-orange-600 w-9 text-center text-white font-bold
rounded-full m-2">{cartItems.length}</div>
    </div>
    :
    <div></div>
    }
  </div>
  );
}

export default CartComponent;
```

장바구니 아이템 컴포넌트

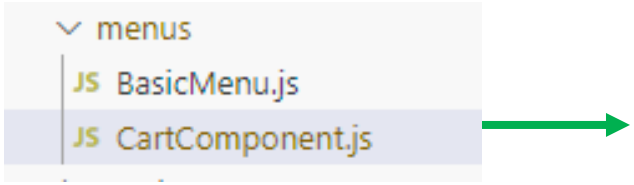
→ components > CartItemComponent 추가




```jsx
const CartItemComponent = ({cino, pname, price, pno, qty, imageFile}) => {

  return (
    <>
      <div>{cino} -- {pname} </div>
    </>
  );
}

export default CartItemComponent;
```

장바구니 아이템 컴포넌트

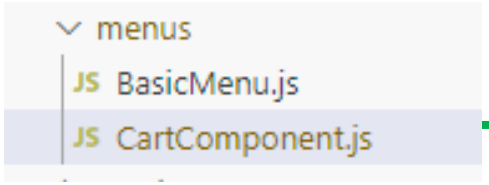
→ menus/CartComponent 에서 각 장바구니 아이템을 CartItemComponent로 출력




```
import { useEffect } from "react";
import useCustomLogin from "../../hooks/useCustomLogin";
import useCustomCart from "../../hooks/useCustomCart";
import CartItemComponent from "../cart/CartItemComponent";

const CartComponent = () => {

  const {isLogin, loginState} = useCustomLogin()

  const {refreshCart, cartItems} = useCustomCart()

  useEffect(() => {
    if(isLogin) {
      refreshCart()
    }
  },[isLogin])
```

장바구니 아이템 컴포넌트

→ menus/CartComponent 에서 각 장바구니 아이템을 CartItemComponent로 출력

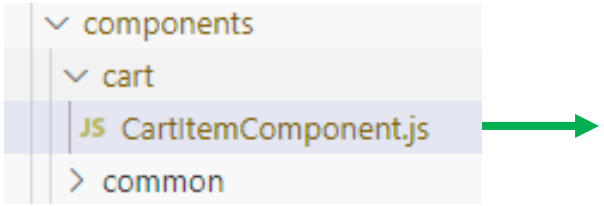



```
  return (
  <div className="w-full">
    {isLogin ?
    <div className="flex flex-col">
      <div className="w-full flex">
        <div className="font-extrabold text-2xl w-4/5"> {loginState.nickname}'s Cart </div>
        <div className="bg-orange-600 text-center text-white font-bold w-1/5 rounded-full m-1">
          {cartItems.length}
        </div>
      </div>
      <div>
        <ul> {cartItems.map( item => <CartItemComponent {...item} key={item.cino}/>)}</ul>
      </div>
    </div>
    :
    <></>
    }
  </div>
  );
}
export default CartComponent;
```

장바구니 아이템 컴포넌트

장바구니 아이템 출력

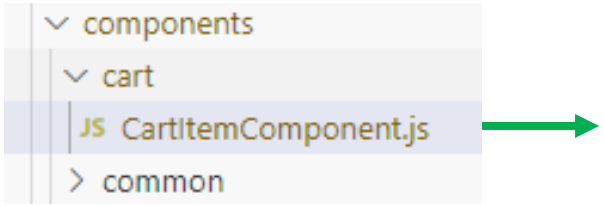
→ cart/CartItemComponent.js




```
import { API_SERVER_HOST } from "../../api/todoApi";

const host = API_SERVER_HOST

const CartItemComponent = ({cino, pname, price, pno, qty, imageFile}) => {

  const handleClickQty = (amount) => {
  }

  return (
  <li key={cino} className="border-2">
      <div className="w-full border-2">
        <div className=" m-1 p-1 ">
          <img src={`${host}/api/products/view/s_${imageFile}`}/>
        </div>
        <div className="justify-center p-2 text-xl ">
          <div className="justify-end w-full">          </div>
          <div>Cart Item No: {cino}</div>
          <div>Pno: {pno}</div>
          <div>Name: {pname}</div>
          <div>Price: {price}</div>
          <div className="flex ">
            <div className="w-2/3"> Qty: {qty} </div>
            <div>
             <button  className="m-1 p-1 text-2xl bg-orange-500 w-8 rounded-lg"
              onClick={() => handleClickQty(1)} > +  </button>
              <button className="m-1 p-1 text-2xl bg-orange-500 w-8 rounded-lg"
              onClick={() => handleClickQty(-1)} > - </button>
            </div>
          </div>
```

장바구니 아이템 출력

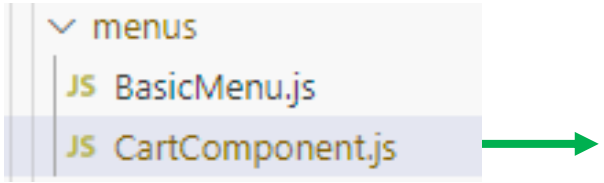
→ cart/CartItemComponent.js




```
          <div className="flex text-white font-bold p-2 justify-center">
            <button
            className="m-1 p-1 text-xl text-white bg-red-500 w-8 rounded-lg"
            onClick={() => handleClickQty(-1 * qty)}
            >
            X
          </button>
          </div>
          <div className='font-extrabold border-t-2 text-right m-2 pr-4'>
            {qty * price} 원
          </div>
          </div>
        </div>
      </div>
    </li>
  );
}

export default CartItemComponent;
```

장바구니 아이템 출력

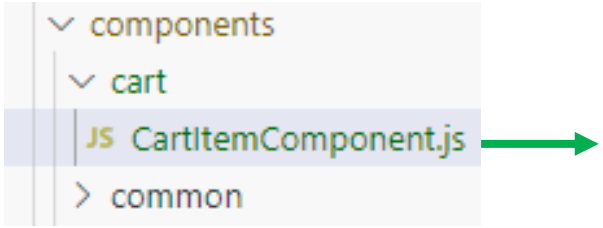
→ 수량 변경 이벤트 처리




```
import { useEffect, useMemo } from "react";
import useCustomCart from "../../hooks/useCustomCart";
import CartItemComponent from "../cart/CartItemComponent";

const CartComponent = () => {

  …생략
  const { refreshCart, cartItems, changeCart } = useCustomCart();

…


      <div>
        <ul>
          {cartItems.map( item =>
            <CartItemComponent {...item} key={item.cino}
changeCart={changeCart} />
          )}
        </ul>
      </div>
```

장바구니 아이템 출력

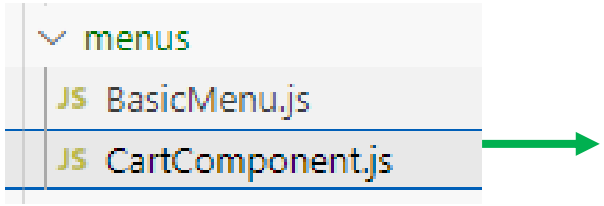
→ 수량 변경 이벤트 처리




```
const CartItemComponent = ({cino, pname, price, pno, qty, imageFile,
changeCart}) => {

  const handleClickQty = (amount) => {

    changeCart({cino: cino, pno: pno, qty: qty + amount})

  }

…생략
```

장바구니 아이템 출력

→ 이메일 전달




```
<div>
  <ul>
    {cartItems.map( item =>
      <CartItemComponent {...item}
      key={item.cino}
      changeCart ={changeCart}
      email={loginState.email}/>)}
  </ul>
</div>
```

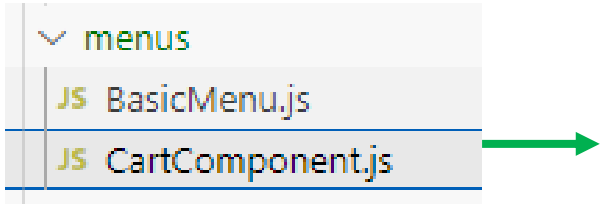
장바구니 아이템 출력

→ 이메일 전달


```
const CartItemComponent = ({cino, pname, price, pno, qty, imageFile,
changeCart, email}) => {

  const handleClickQty = (amount) => {

    changeCart({ email, cino, pno, qty: qty + amount})
  }
```

장바구니 아이템 출력

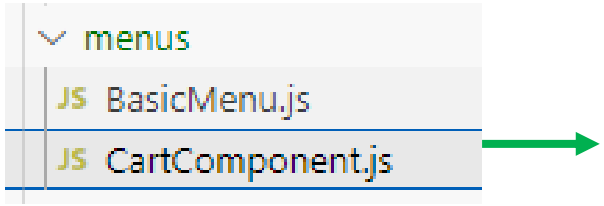
→ 이메일 전달




```
import { useEffect,useMemo } from "react";
import useCustomLogin from "../../hooks/useCustomLogin";
import useCustomCart from "../../hooks/useCustomCart";
import CartItemComponent from "../cart/CartItemComponent";

const CartComponent = () => {
  const {isLogin, loginState} = useCustomLogin()
  const {refreshCart, cartItems, changeCart } = useCustomCart()

  const total = useMemo(() => {
    let total = 0
    for(const item of cartItems) {
      total += item.price * item.qty
    }
    return total
  },[cartItems])

  useEffect(() => {
    if(isLogin) {
      refreshCart()
    }
  },[isLogin])
```

장바구니 아이템 출력

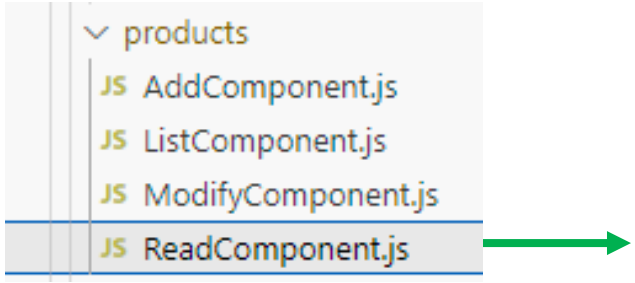
→ 이메일 전달




```
  return (
  <div className="w-full">
    {isLogin ?
    <div className="flex flex-col">
      …생략
      <div>
        <div className="text-2xl text-right font-extrabold">
          TOTAL: {total}
        </div>
      </div>
    </div>
    :
   <></>
    }
  </div>
  );
}

export default CartComponent;
```

상품 조회 기능 수정

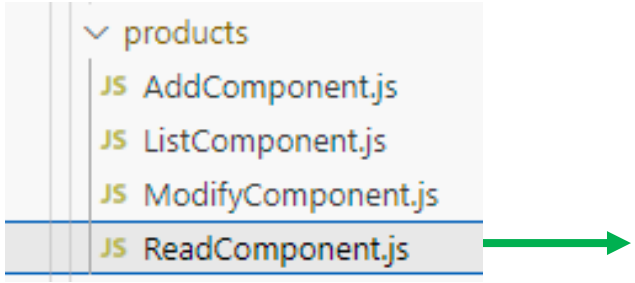
→ components/products/ReadComponet.js 수정




```
import { useEffect, useState } from "react"
import  {getOne} from "../../api/productsApi"
import { API_SERVER_HOST } from "../../api/todoApi"
import useCustomMove from "../../hooks/useCustomMove"
import FetchingModal from "../common/FetchingModal"
import useCustomCart from "../../hooks/useCustomCart"
import useCustomLogin from "../../hooks/useCustomLogin"

const initState = {
  pno:0,
  pname: '',
  pdesc: '',
  price: 0,
  uploadFileNames:[]
}

const host = API_SERVER_HOST
```

상품 조회 기능 수정

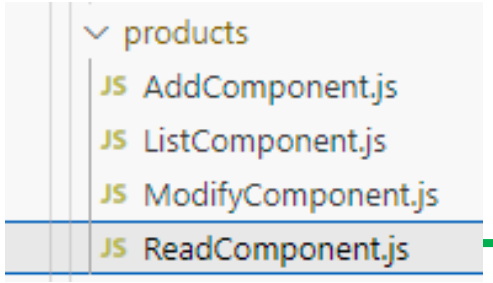
→ components/products/ReadComponet.js 수정




```
const ReadComponent = ({pno }) => {

  const [product, setProduct] = useState(initState)

  //화면 이동용 함수
  const {moveToList, moveToModify} = useCustomMove()

  //fetching
  const [fetching, setFetching] = useState(false)

  //장바구니 기능
  const {changeCart, cartItems} = useCustomCart()

  //로그인 정보
  const {loginState} = useCustomLogin()
```

상품 조회 기능 수정

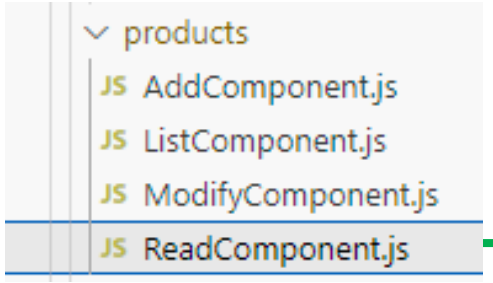
→ components/products/ReadComponet.js 수정




```
const handleClickAddCart = () => {
  let qty = 1
  const addedItem = cartItems.filter(item => item.pno === parseInt(pno))[0]

  if(addedItem) {
    if(window.confirm("이미 추가된 상품입니다. 추가하시겠습니까?") === false){
      return
    }
    qty = addedItem.qty + 1
  }
  changeCart({email: loginState.email, pno:pno, qty:qty})
}

...생략...
```

상품 조회 기능 수정

→ components/products/ReadComponet.js 수정




```
      <div className="flex justify-end p-4">
        <button type="button"
          className="inline-block rounded p-4 m-2 text-xl w-32  text-white bg-
green-500"
          onClick={handleClickAddCart} > Add Cart </button>
        <button type="button" onClick={() => moveToModify(pno)}
         className="inline-block rounded p-4 m-2 text-xl w-32  text-white bg-
red-500" > Modify </button>
        <button type="button" onClick={moveToList}
          className="rounded p-4 m-2 text-xl w-32 text-white bg-blue-500">
          List
        </button>
      </div>
```

감사합니다.