React

json data server

```
npm install -g json-server
# 또는
yarn global add json-server
```

db.json

```json
{
   "users": [
     { "id": 1, "name": "Alice", "age": 30, "email": "alice@example.com", "city": "New
York" },
     { "id": 2, "name": "Bob", "age": 25, "email": "bob@example.com", "city": "Los Angeles"
},
     { "id": 3, "name": "Charlie", "age": 35, "email": "charlie@example.com", "city":
"Chicago" },
     { "id": 4, "name": "David", "age": 28, "email": "david@example.com", "city": "Houston"
},
     { "id": 5, "name": "Eve", "age": 22, "email": "eve@example.com", "city": "Phoenix" },
     { "id": 6, "name": "Jona", "age": 40, "email": "jona@example.com", "city":
"Philadelphia" },
     { "id": 7, "name": "Kim", "age": 31, "email": "kim@example.com", "city": "San Antonio"
}
   ],
}
```

```
"products": [
     { "id": "A1", "name": "Laptop", "price": 1200, "category": "Electronics", "inStock":
true },
     { "id": "B2", "name": "Mouse", "price": 25, "category": "Electronics", "inStock":
true },
     { "id": "C3", "name": "Keyboard", "price": 75, "category": "Electronics", "inStock":
false },
     { "id": "D4", "name": "Monitor", "price": 300, "category": "Electronics", "inStock":
true },
     { "id": "E5", "name": "Tablet", "price": 450, "category": "Electronics", "inStock":
true },
     { "id": "F6", "name": "Smartphone", "price": 900, "category": "Electronics",
"inStock": true },
     { "id": "G7", "name": "Headphones", "price": 150, "category": "Electronics",
"inStock": false },
     { "id": "H8", "name": "T-Shirt", "price": 20, "category": "Apparel", "inStock": true
},
     { "id": "I9", "name": "Jeans", "price": 60, "category": "Apparel", "inStock": true
},
     { "id": "J10", "name": "Book", "price": 15, "category": "Books", "inStock": true }
   ]
 }
```

package.json

```json
{
   "name": "my-json-server-project",
   "version": "1.0.0",
   "description": "",
   "main": "index.js",
   "scripts": {
     "start": "json-server --watch db.json --port 3001"
   },
   "keywords": [],
   "author": "",
   "license": "ISC",
   "dependencies": {
     "json-server": "^0.17.4" // 예시 버전1
   }
 }
```

실행 명령어

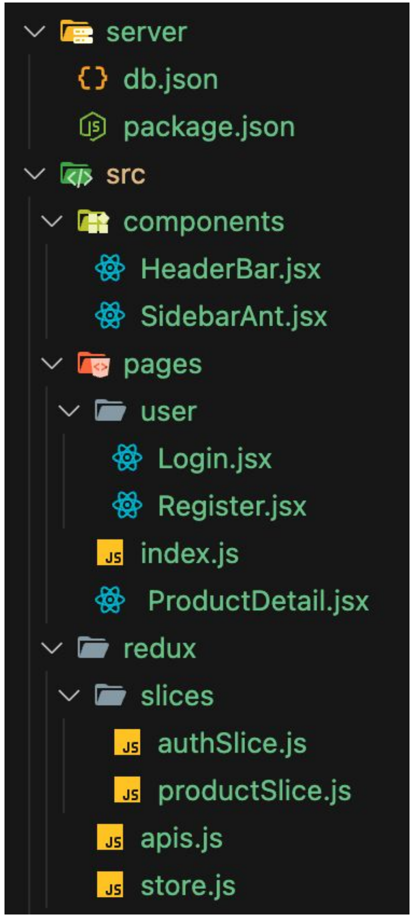
```
json-server --watch db.json --port 3001
```

라이브러리

react-redux
@reduxjs/toolkit
react-router-dom
antd
@ant-design/icons

Login, Logout, Register, 상품등록

전체 폴더 구조



```
server
    db.json
    package.json
src
    components
        HeaderBar.jsx
        SidebarAnt.jsx
    pages
        user
            Login.jsx
            Register.jsx
        index.js
        ProductDetail.jsx
    redux
        slices
            authSlice.js
            productSlice.js
        apis.js
        store.js
```

components/HeaderBar.jsx

```
import { Button, Space } from 'antd';
import { useNavigate } from 'react-router-dom';
import { useSelector, useDispatch } from 'react-redux';
import { logout } from '../redux/slices/authSlice';

const HeaderBar = () => {
 const navigate = useNavigate();
 const dispatch = useDispatch();

 // Redux에서 로그인 상태 가져오기
 const user= useSelector((state) => state.auth.user);

 const handleLogout = () => {
   dispatch(logout());
   navigate('/login');
 };
```

```
return (
  <div
    style={{
      display: 'flex',
      alignItems: 'center',
      justifyContent: 'space-between',
      padding: '0 24px',
      height: '100%',
      backgroundColor: '#2c3e50',
      boxShadow: '0 2px 8px #f0f1f2',
    }}
  >
    {/* 왼쪽: 로고 */}
    <div style={{ fontSize: '20px', fontWeight: 'bold', color: '#1890ff' }}>
      MySystemLogo
    </div>
```

```
     {/* 오른쪽: 로그인 상태에 따라 다르게 표시 */}
     <Space>
       {user ? (
         <>
           <span style={{ color: 'white' }}>{user.name}님</span>
           <Button type="primary" danger onClick={handleLogout}>로그아웃</Button>
         </>
       ) : (
         <>
           <Button onClick={() => navigate('/login')}>로그인</Button>
           <Button onClick={() => navigate('/register')}>회원가입</Button>
         </>
       )}
     </Space>
   </div>);};
export default HeaderBar;
```

components/SidebarAnt.jsx

```
import { Layout, Menu } from 'antd';
import { Link, useLocation } from 'react-router-dom';
import { useDispatch, useSelector } from 'react-redux';
import { productList } from '../redux/slices/productSlice';
import { FolderOpenOutlined, FileTextOutlined } from '@ant-design/icons';
import { useEffect } from 'react';

const { Sider } = Layout;

const SidebarAnt = () => {
 const location = useLocation();
 const dispatch = useDispatch();
 const products = useSelector((state) => state.product.productData);
```

```
// ✅ 컴포넌트 마운트 시 상품 목록 불러오기
 useEffect(() => {
   dispatch(productList());
 }, [dispatch]);

 // ✅ 카테고리별로 그룹화
 const groupedProducts = products.reduce((acc, product) => {
   const { category } = product;
   if (!acc[category]) {
     acc[category] = [];
   }
   acc[category].push(product);
   return acc;
 }, {});
```

```
return (
  <Sider
    width={240}
    style={{
      backgroundColor: '#d4f1f9',
      paddingTop: '70px',
    }}
  >
    <Menu
      mode="inline"
      selectedKeys={[location.pathname]}
      style={{
        backgroundColor: 'transparent',
        borderRight: 0,
      }}
    >
```

```
     {Object.entries(groupedProducts).map(([category, items]) => (
         <Menu.SubMenu key={category} icon={<FolderOpenOutlined />}
           title={category}
         >
           {items.map((item) => (
             <Menu.Item key={`/product/${item.id}`} icon={<FileTextOutlined />}>
               <Link to={`/product/${item.id}`}>{item.name}</Link>
             </Menu.Item>
           ))}
         </Menu.SubMenu>
       ))}
     </Menu>
   </Sider>
 );
};
export default SidebarAnt;
```

pages/user/Login.jsx

```
import React, { useState } from 'react';
import { useDispatch } from 'react-redux';
import { loginUser } from '../../redux/slices/authSlice';
import { useNavigate } from 'react-router-dom';
import { Input, Button, Card, Space } from 'antd';

const Login = () => {
 const [username, setUsername] = useState('');
 const [password, setPassword] = useState('');
 const dispatch = useDispatch();
 const navigate = useNavigate();
const handleLogin = async () => {
   const result = await dispatch(loginUser({ username, password }));
    if (loginUser.fulfilled.match(result)) {
       navigate('/');
   } else {
     alert(result.payload);
   }
 };

 const goToRegister = () => {
   navigate('/register');
 };
```

```
  return (
    <div style={{ height: '100vh', display: 'flex', alignItems: 'center',
justifyContent: 'center', background: '#f0f2f5' }}>
      <Card title="로그인" style={{ width: 400 }}>
        <Input
          placeholder="아이디"
          value={username}
          onChange={(e) => setUsername(e.target.value)}
          style={{ marginBottom: 16 }}
        />
        <Input.Password
          placeholder="비밀번호"
          value={password}
          onChange={(e) => setPassword(e.target.value)}
          style={{ marginBottom: 16 }}
        />
        <Space direction="vertical" style={{ width: '100%' }}>
            <Button type="primary" block onClick={handleLogin}>
              로그인
            </Button>
            <Button type="link" block onClick={goToRegister}>
              아직 회원이 아니신가요? 회원가입
            </Button>
          </Space>
        </Card>
      </div>
  );
};

export default Login;
```

pages/user/Register.jsx

```
import React, { useState } from 'react';
import { Input, Button, Card, Space, message } from 'antd';
import { useNavigate } from 'react-router-dom';
import { useDispatch } from 'react-redux';
import { registerUser } from '../../redux/slices/authSlice';

const Register = () => {
 const [username, setUsername] = useState('');
 const [password, setPassword] = useState('');
 const [confirm, setConfirm] = useState('');
 const [age, setAge] = useState('');
 const [email, setEmail] = useState('');
 const [city, setCity] = useState('');

 const dispatch = useDispatch();
 const navigate = useNavigate();
```

```javascript
const handleRegister = async () => {
  if (password !== confirm) {
    message.error('❌ 비밀번호가 일치하지 않습니다.');
    return;
  }

  const result = await dispatch(
    registerUser({ name: username, password, age, email, city })
  );

  if (registerUser.fulfilled.match(result)) {
    message.success('✅ 회원가입 완료! 로그인 해주세요.');
    navigate('/login');
  } else {
    message.error(`❌ ${result.payload || '회원가입 실패'}`);
  }
};
```

```
 return (
   <div style={{ height: '100vh', background: '#f0f2f5', display: 'flex',
justifyContent: 'center', alignItems: 'center' }}>
     <Card title="회원가입" style={{ width: 400 }}>
       <Space direction="vertical" style={{ width: '100%' }}>
         <Input placeholder="아이디" value={username} onChange={(e) =>
setUsername(e.target.value)} />
         <Input.Password placeholder="비밀번호" value={password} onChange={(e) =>
setPassword(e.target.value)} />
         <Input.Password placeholder="비밀번호 확인" value={confirm} onChange={(e)
=> setConfirm(e.target.value)} />
         <Input placeholder="나이" value={age} onChange={(e) =>
setAge(e.target.value)} type="number" />
```

```
          <Input placeholder="이메일" value={email} onChange={(e) =>
setEmail(e.target.value)} />
          <Input placeholder="도시" value={city} onChange={(e) =>
setCity(e.target.value)} />
          <Button type="primary" onClick={handleRegister} block>회원가입</Button>
          <Button type="link" onClick={() => navigate('/login')} block>이미
회원이신가요? 로그인</Button>
        </Space>
      </Card>
    </div>
  );
};
export default Register;
```

pages/ProductDetail.jsx

```jsx
// pages/ProductDetail.jsx
import { useParams } from 'react-router-dom';
import { useSelector } from 'react-redux';
const ProductDetail = () => {
 const { id } = useParams();
 const products = useSelector((state) => state.product.productData);
 const product = products.find((p) => p.id === id);
 if (!product) return <div>상품 정보를 찾을 수 없습니다.</div>;
 return (
   <div>
     <h2>{product.name}</h2>
     <p>가격: ₩{product.price}</p>
     <p>카테고리: {product.category}</p>
     <p>재고: {product.inStock ? '있음' : '없음'}</p>
   </div>);};
export default ProductDetail;
```

pages/index.js

```
export { default as user_Login } from "./user/Login.jsx";
export { default as user_Register } from "./user/Register.jsx";
export { default as ProductDetail } from "./ProductDetail.jsx";
```

redux/slices/authSlice.js

```
// redux/slices/authSlice.js
import { createSlice, createAsyncThunk } from '@reduxjs/toolkit';
import {user_api} from '../apis'; // api 불러오기

export const registerUser = createAsyncThunk(
 'auth/registerUser',
 async ({ name, password, age, email, city }, thunkAPI) => {
   try {
     // 중복 확인
     const existing = await user_api.get(`/?name=${name}`);
     if (existing.data.length > 0) {
       return thunkAPI.rejectWithValue('이미 존재하는 사용자입니다.');
     }
     // 사용자 등록
     const res = await user_api.post('/', {
       name,
       password,
       age: Number(age),
       email,
       city,
     });

     return res.data;
   } catch (err) {
     return thunkAPI.rejectWithValue(err.message || '회원가입 실패');
   }
 }
);
```

```
// 로그인 Thunk - username + password 검사
export const loginUser = createAsyncThunk(
 'users/loginUser',
 async ({ username, password }, thunkAPI) => {
   try {
     const res = await user_api.get(`/?name=${username}`);
     const data = res.data;

     if (data.length === 0) {
       return thunkAPI.rejectWithValue('사용자를 찾을 수 없습니다.');
     }

     const user = data[0];
     // console.log("user", typeof user.password)
     // console.log("password", typeof password)
     if (user.password !== password) {
       return thunkAPI.rejectWithValue('비밀번호가 틀렸습니다.');
     }
     return user; // 로그인 성공
   } catch (err) {
     return thunkAPI.rejectWithValue('서버 오류: ' + err.message);
   }
 }
);
```

```
const initialState = {
 isAuthenticated: false,
 user: null,
 status: 'idle',
 error: null,
};
const authSlice = createSlice({
 name: 'auth',
 initialState,
 reducers: {
   logout(state) {
     state.isAuthenticated = false;
     state.user = null;
     state.status = 'idle';
     state.error = null;},},
extraReducers: (builder) => {
   builder
     .addCase(loginUser.pending, (state) => {
       state.status = 'loading';
     })
     .addCase(loginUser.fulfilled, (state, action) => {
       state.status = 'succeeded';
       state.isAuthenticated = true;
       state.user = action.payload;
     })
     .addCase(loginUser.rejected, (state, action) => {
       state.status = 'failed';
       state.error = action.payload;
     });},});
export const { logout } = authSlice.actions;
export default authSlice.reducer;
```

redux/slices/productSlice.js

```
// redux/slices/productSlice.js
import { createSlice, createAsyncThunk } from '@reduxjs/toolkit';
import {product_api} from '../apis'; // axios 인스턴스

// 🔹 비동기 요청: 상품 리스트 가져오기
export const productList = createAsyncThunk(
 'product/productList',
 async (_, thunkAPI) => {
   try {
     const res = await product_api.get('/');
     const data = res.data;

     if (!data || data.length === 0) {
       return thunkAPI.rejectWithValue('상품이 없습니다.');
     }
```

```
      return data; // 전체 상품 배열 반환
    } catch (err) {
      return thunkAPI.rejectWithValue(err.message);
    }
  }
);

// 🔹 초기 상태
const initialState = {
  productData: [],
  status: 'idle', // 'loading' | 'succeeded' | 'failed'
  error: null,
};
```

```javascript
// 🔹 Slice 생성
const productSlice = createSlice({
 name: 'product',
 initialState,
 reducers: {
   // 필요한 경우 제품 데이터 초기화용
   clearProducts(state) {
     state.productData = [];
     state.status = 'idle';
     state.error = null;
   },
 },
extraReducers: (builder) => {
   builder
     .addCase(productList.pending, (state) => {
       state.status = 'loading';
     })
     .addCase(productList.fulfilled, (state, action) => {
       state.status = 'succeeded';
       state.productData = action.payload;
     })
     .addCase(productList.rejected, (state, action) => {
       state.status = 'failed';
       state.error = action.payload;
     });},});
// 🔹 액션/리듀서 export
export const { clearProducts } = productSlice.actions;
export default productSlice.reducer;
```

redux/apis.js

```
// src/api/api.js
import axios from 'axios';

export const user_api = axios.create({
 baseURL: 'http://localhost:3001/users', // json-server 주소
 headers: {
   'Content-Type': 'application/json',
 },
});

export const product_api = axios.create({
   baseURL: 'http://localhost:3001/products', // json-server 주소
   headers: {
     'Content-Type': 'application/json',
   },
 });
```

redux/store.js

```
/* redux/store.js */
import { configureStore } from '@reduxjs/toolkit';
import authReducer from './slices/authSlice';
import productReducer from './slices/productSlice';

const store = configureStore({
 reducer: {
   auth: authReducer,
   product: productReducer,
 },
});

export default store;
```

App.js

```
import { BrowserRouter as Router, Routes, Route } from 'react-router-dom';
import { Layout } from 'antd';
import SidebarAnt from './components/SidebarAnt';
import HeaderBar from './components/HeaderBar';
import * as Pages from './pages'; // 폴더 내 페이지 export
import ProductDetail from './pages/ProductDetail'; // 상품 상세 페이지

const { Header, Sider, Content } = Layout;

function App() {
 return (
   <Router>
     <Routes>
       {/* 로그인 및 회원가입 페이지 (레이아웃 없음) */}
       <Route path="/login" element={<Pages.user_Login />} />
       <Route path="/register" element={<Pages.user_Register />} />
```

```jsx
{/* 공통 레이아웃이 적용된 페이지들 */}
<Route
  path="/*"
  element={
    <Layout style={{ minHeight: '100vh' }}>
      <Header style={{ padding: 0, background: '#fff' }}>
        <HeaderBar />
      </Header>
      <Layout>
        <SidebarAnt />
        <Content
          style={{
            margin: '24px 16px',
            padding: 24,
            background: '#fff',
          }}
        >
```

```
                    <Routes>
                      {/* 정적 라우트 (예: Home 등)
                      <Route path="/" element={<Pages.Home />} /> */}
                      {/* 상품 상세 페이지 */}
                      <Route path="/product/:id" element={<ProductDetail />} />
                      {/* 더 추가할 정적 페이지가 있다면 여기에 작성 */}
                    </Routes>
                  </Content>
                </Layout>
              </Layout>
            }
          />
        </Routes>
      </Router>
  );}
export default App;
```

index.js

```
import React from 'react';
import ReactDOM from 'react-dom/client';
import './index.css';
import App from './App';
import { Provider } from 'react-redux';
import store from './redux/store';
// import reportWebVitals from './reportWebVitals';

const root = ReactDOM.createRoot(document.getElementById('root'));
root.render(
 <Provider store={store}>
   <App />
 </Provider>
);
```

실습3(todos)

```
npm install react-icons
# 또는
yarn add react-icons

npm install classnames
# 또는
yarn add classnames
```

TodoTemplate.jsx

```
const todoTemplateStyle = {
  width: '512px',
  marginLeft: 'auto',
  marginRight: 'auto',
  marginTop: '6rem',
  borderRadius: '4px',
  overflow: 'hidden',
};
const appTitleStyle = {
  background: '#22b8cf',
  color: 'white',
  height: '4rem',
  fontSize: '1.5rem',
  display: 'flex',
  alignItems: 'center',
  justifyContent: 'center',
};
```

```
const contentStyle = {
  background: 'white',
 };
export default function TodoTemplate({ children }) {
  return (
    <div className="TodoTemplate" style={todoTemplateStyle}>
      <div className="app-title" style={appTitleStyle}>
        일정관리
      </div>
      <div className="content" style={contentStyle}>
        {children}
      </div>
    </div>
  );
 }
```

TodoInsert.jsx

```
import React, { useState, useCallback } from 'react';
import { MdAdd } from 'react-icons/md'; // react-icons 라이브러리 필요
const todoInsertStyle = {
   display: 'flex',
   background: '#495057',
 };
const inputStyle = {
   background: 'none',
   outline: 'none',
   border: 'none',
   padding: 0,
   fontSize: '1.125rem',
   lineHeight: 1.5,
   color: 'white',
   flex: 1, // flex-grow: 1, flex-shrink: 1, flex-basis: 0
   marginRight: '1rem', // 버튼과의 간격
 };
```

```
const placeholderStyle = {
  color: '#dee2e6',
};
const buttonStyle = {
  background: '#868e96',
  outline: 'none',
  border: 'none',
  color: 'white',
  paddingLeft: '1rem',
  paddingRight: '1rem',
  fontSize: '1.5rem',
  display: 'flex',
  alignItems: 'center',
  cursor: 'pointer',
  transition: 'background 0.1s ease-in',
};
```

```
const buttonHoverStyle = {
   background: '#adb5bd',
 };
export default function TodoInsert({ onInsert }) {
 const [value, setValue] = useState('');
 const onChange = useCallback((e) => {
   setValue(e.target.value);
 }, []); // 의존성 배열 비움
 const onSubmit = useCallback(
   (e) => {
     e.preventDefault();
     onInsert(value);
     setValue('');
   },
   [onInsert, value]
 );
```

```
  return (
    <form className="TodoInsert" style={todoInsertStyle} onSubmit={onSubmit}>
      <input
        name="todo"
        style={inputStyle}
        value={value}
        onChange={onChange}
        placeholder="할 일을 입력하세요"
        placeholderStyle={placeholderStyle} // React Native 스타일과 혼동될 수 있으므로 주의
      />
      <button type="submit" style={buttonStyle}>
        <MdAdd />
      </button>
    </form>
  );
}
```

TodoList.jsx

```
import React from 'react';
import TodoListItem from './TodoListItem';
const todoListStyle = {
   minHeight: '320px',
   maxHeight: '513px',
   overflowY: 'auto',
 };
const TodoList = ({ todos, onRemove, onToggle }) => {
```

```
return (
   <div className="TodoList" style={todoListStyle}>
     {todos.map((todo) => (
       <TodoListItem
         key={todo.id}
         todo={todo}
         onRemove={onRemove}
         onToggle={onToggle}
       />
     ))}
   </div>
 );
};
export default TodoList;
```

TodoListItem.jsx

```
import React, { memo } from 'react';
import {
 MdCheckBoxOutlineBlank,
 MdCheckBox,
 MdRemoveCircleOutline,
} from 'react-icons/md';
import classNames from 'classnames';
const todoListItemStyle = {
   padding: '1rem',
   display: 'flex',
   alignItems: 'center',
   '&:nth-child(even)': { // React style에서는 직접 적용 어려움
     backgroundColor: '#f8f9fa',
   },
   '& + &': { // React style에서는 직접 적용 어려움
     borderTop: '1px solid #dee2e6',
   },
 };
```

```
const checkboxStyle = {
  cursor: 'pointer',
  flex: 1,
  display: 'flex',
  alignItems: 'center',
  svg: {
    fontSize: '1.5rem',
  },
  '&.checked': { // React style에서는 직접 적용 어려움
    svg: {
      color: '#22b8cf',
    },
    '.text': {
      color: '#adb5bd',
      textDecoration: 'line-through',
    },
  },
};
```

```
const textStyle = {
  marginLeft: '0.5rem',
  flex: 1,
};
const removeStyle = {
  display: 'flex',
  alignItems: 'center',
  fontSize: '1.5rem',
  color: '#ff6b6b',
  cursor: 'pointer',
  '&:hover': { // React style에서는 직접 적용 어려움
    color: '#ff8787',
  },
};
```

```
const checkedStyle = {
   svg: {
     color: '#22b8cf',
   },
   text: {
     color: '#adb5bd',
     textDecoration: 'line-through',
   },
 };
const TodoListItem = memo(({ todo, onRemove, onToggle }) => {
 const { id, text, checked } = todo;
```

```
return (
  <div className="TodoListItem" style={todoListItemStyle}>
    <div
      className={classNames('checkbox', { checked })}
      style={checkboxStyle}
      onClick={() => onToggle(id)}
    >
      {checked ? <MdCheckBox style={checkboxStyle.svg} /> : <MdCheckBoxOutlineBlank
style={checkboxStyle.svg} />}
      <div className="text" style={textStyle}>
        {text}
      </div>
    </div>
```

```
     <div
       className="remove"
       style={removeStyle}
       onClick={() => onRemove(id)}
     >
       <MdRemoveCircleOutline />
     </div>
   </div>
 );
});

export default TodoListItem;
```

todoReducer.js

```
const todoReducer = (todos, action)=>{
   switch (action.type){
     case "insert":
       return [...todos, action.todo];
     case "remove":
       return todos.filter(todo=>todo.id!==action.id);
     case "toggle":
       return todos.map(todo=>
       todo.id===action.id ? {...todo, checked:
!todo.checked}:todo);
     default: return todos
   }
  }
  export default todoReducer;
```

App.js

```
import './App.css';
import TodoTemplate from './components/TodoTemplate';
import TodoInsert from './components/TodoInsert';
import TodoList from './components/TodoList';
import todoJson from './components/data/todos.json'
import { useState, useRef, useCallback, useReducer } from 'react';
import todoReducer from './modules/todoReducer';
const todoList = () =>{
 const list = [];
 for(let i=1;i<=2500;i++){
   list.push({
     id:i,
     text: `todo ${i}`,
     checked: false
   })
 }
 return list;
}
```

App.js

```
function App() {
 const [todos, dispatch] = useReducer(todoReducer, todoJson, todoList)
 const nextId = useRef(todoList.length+1)
 const onInsert = useCallback((text)=>{
   const todo = {
     id: nextId.current,
     text,
     checked: false
   }
   dispatch({type:"insert", todo})
   nextId.current += 1;
 }, [])
 const onRemove =useCallback((id)=>{
   dispatch({type:"remove", id})
 },[])
 const onToggle = useCallback((id)=>{
  dispatch({type:'toggle', id})}, []);
```

App.js

```
 return (
     <TodoTemplate className="main">
       <TodoInsert onInsert={onInsert}/>
       <TodoList
         todos={todos}
         onRemove={onRemove}
         onToggle={onToggle}
       />
     </TodoTemplate>
 );
}
export default App;
```