**ch 04. CRUD**

**Create**

**Read**

**Update**

**Delete**

1. Create form UI
   * App.js

import './App.css';

import React, {Component} from 'react';

import Subject from './components/Subject';

import TOC from './components/TOC';

import Control from './components/Control';

import ReadContent from './components/ReadContent';

import CreateContent from './components/CreateContent';

class App extends Component {

  constructor(props){

    super(props);

    this.state = {

      mode : 'read',

      selected\_content\_id : 1,

      subject : {title:'WEB', sub:'World Wide Web!'},

      welcome : {title:'welcome', desc:'Hello, React!!!'},

      contents : [

        {id:1, title:'HTML', desc:'HTML is for information'},

        {id:2, title:'CSS', desc:'CSS is for design'},

        {id:3, title:'JavaScript', desc:'JavaScript is for interactive'},

      ],

    };

  }

  render(){

    var \_title, \_desc, \_article = null;

    if(this.state.mode ==='welcome'){

      \_title = this.state.welcome.title;

      \_desc = this.state.welcome.desc;

      \_article = <ReadContent title={\_title} desc={\_desc}></ReadContent>;

    }else if(this.state.mode === 'read'){

//var content = this.state.contents[this.state.selected\_content\_id-1];

      //\_title = content.title;

      //\_desc = content.desc;

      for(var i=0 ; i<this.state.contents.length ; i++){

        var data = this.state.contents[i];

        if(data.id === this.state.selected\_content\_id){

          \_title = data.title;

          \_desc = data.desc;

          break;

        }

      }

      \_article = <ReadContent title={\_title} desc={\_desc}></ReadContent>;

    }else if(this.state.mode === 'create'){

      \_article = <CreateContent></CreateContent>;

    }

    return (

      <div className="App">

        <Subject title={this.state.subject.title}

          sub={this.state.subject.sub}

          onChangePage={function(e){

            this.setState({

              mode : 'welcome'

            })

          }.bind(this)}></Subject>

        <TOC data={this.state.contents}

          onChangePage={function(id){

            this.setState({

              mode:'read',

              selected\_content\_id : Number(id)

            });

          }.bind(this)}></TOC>

        <Control

          onChangeMode={function(\_mode){

            //alert('mode를 수정 :',\_mode);

            this.setState({

              mode : \_mode

            });

          }.bind(this)}></Control>

        {\_article}

      </div>

    );

  }

}

export default App;

* Control.js

import React, {Component} from "react";

class Control extends Component{

  render(){

    return (

      <ul>

        <li><a href="/create" onClick={function(e){

          e.preventDefault();

          this.props.onChangeMode('create');

        }.bind(this)}>create</a></li>

        <li><a href="/update" onClick={function(e){

          e.preventDefault();

          this.props.onChangeMode('update');

        }.bind(this)}>update</a></li>

        <li><input  onClick={function(e){

          e.preventDefault();

          this.props.onChangeMode('delete');

        }.bind(this)} type="button" value="delete"/></li>

      </ul>

    );

  };

}

export default Control;

* CreateContent.js

import React, {Component} from 'react';

class CreateContent extends Component{

    render(){

        console.log('Subject');

        return (

            <article>

                <h2>Create</h2>

                <form>

                    <p><input type="text" name="title" placeholder="title"/></p>

                    <p><textarea name="desc" placeholder="desc"></textarea></p>

                    <p><input type="submit"/></p>

                </form>

            </article>

        );

    }

}

export default CreateContent;

* ReadContent.js

import React, {Component} from 'react';

class ReadContent extends Component{

    render(){

        console.log('Subject');

        return (

            <article>

                <h2>{this.props.title}</h2>

                {this.props.desc}

            </article>

        );

    }

}

export default ReadContent;

1. Create form onSubmit 이벤트 생성

* 배열 추가 (concat을 이용해 원본 데이터를 변경하지 않고 새로운 데이터를 생성)

var arr = [1,2];

arr.push(3);

console.log(arr);

var arr2 = [1,2];

var result = arr2.concat(3);

console.log(arr2);

console.log(result);

* App.js

import './App.css';

import React, {Component} from 'react';

import Subject from './components/Subject';

import TOC from './components/TOC';

import Control from './components/Control';

import ReadContent from './components/ReadContent';

import CreateContent from './components/CreateContent';

class App extends Component {

  constructor(props){

    super(props);

    this.max\_content\_id = 3; // UI

    this.state = {

      mode : 'read',

      selected\_content\_id : 1,

      subject : {title:'WEB', sub:'World Wide Web!'},

      welcome : {title:'welcome', desc:'Hello, React!!!'},

      contents : [

        {id:1, title:'HTML', desc:'HTML is for information'},

        {id:2, title:'CSS', desc:'CSS is for design'},

        {id:3, title:'JavaScript', desc:'JavaScript is for interactive'},

      ],

    };

  }

  render(){

    console.log('App render')

    var \_title, \_desc, \_article = null;

    if(this.state.mode ==='welcome'){

      \_title = this.state.welcome.title;

      \_desc = this.state.welcome.desc;

      \_article = <ReadContent title={\_title} desc={\_desc}></ReadContent>;

    }else if(this.state.mode === 'read'){

      //var content = this.state.contents[this.state.selected\_content\_id-1];

      //\_title = content.title;

      //\_desc = content.desc;

      for(var i=0 ; i<this.state.contents.length ; i++){

        var data = this.state.contents[i];

        if(data.id === this.state.selected\_content\_id){

          \_title = data.title;

          \_desc = data.desc;

          break;

        }

      }

      \_article = <ReadContent title={\_title} desc={\_desc}></ReadContent>;

    }else if(this.state.mode === 'create'){

      \_article = <CreateContent

                  onSubmit={function(\_title, \_desc){

                    this.max\_content\_id ++;

                    var \_contents = this.state.contents.concat(

                      {id:this.max\_content\_id, title:\_title, desc:\_desc}

                    );

                    this.setState({

                      contents : \_contents

                    });

                  }.bind(this)}></CreateContent>;

    }else if(this.state.mode === 'update'){

      \_article = <h2>고치려고</h2>

    }else if(this.state.mode === 'delete'){

      \_article = <h2>지우려구</h2>

    }

    return (

      <div className="App">

        <Subject title={this.state.subject.title}

          sub={this.state.subject.sub}

          onChangePage={function(e){

            this.setState({

              mode : 'welcome'

            })

          }.bind(this)}></Subject>

        <TOC data={this.state.contents}

          onChangePage={function(id){

            this.setState({

              mode:'read',

              selected\_content\_id : Number(id)

            });

          }.bind(this)}></TOC>

        <Control

          onChangeMode={function(\_mode){

            //alert('mode를 수정 :',\_mode);

            this.setState({

              mode : \_mode

            });

          }.bind(this)}></Control>

        {\_article}

      </div>

    );

  }

}

export default App;

* CreateContent.js

import React, {Component} from 'react';

class CreateContent extends Component{

    render(){

        return (

            <article>

                <h2>Create</h2>

                <form action="/create\_process" method="post"

                    onSubmit={function(e){

                        e.preventDefault();

                        //alert('submit');

                        //console.log(e);

                        //debugger;

                        //console.log(e.target.desc.value);

                        this.props.onSubmit(

                            e.target.title.value,

                            e.target.desc.value

                        );

                        e.target.title.value='';

                        e.target.desc.value='';

                    }.bind(this)}

                >

                    <p><input type="text" name="title" placeholder="title"/></p>

                    <p><textarea name="desc" placeholder="desription"></textarea></p>

                    <p><input type="submit"/></p>

                </form>

            </article>

        );

    }

}

export default CreateContent;

1. TOC에 뿌려질 this.state.contents가 바뀌지 않으면 TOC는 다시 render하지 않도록 업그레이드(shouldComponentUpdate 함수 이용) : 조그만 app의 경우 push나 concat의 차이를 느끼지 못할 수 있음.

그러나 app을 확장할 경우도 있으므로 push(원본수정)보다 concat(복제본 수정)을 사용 추천

* TOC.js

import React, {Component} from 'react';

class TOC extends Component{

    shouldComponentUpdate(newProps/\* : TOC의 props가 바뀌었을때 값 , newState : state값이 바뀐 값\*/ ){

        console.log('TOC render shouldComponentUpdate()의 새로 바뀐 props값 :'

            , newProps.data, '바뀌기 전 이전값 :'

            , this.props.data

        );

        // return true; //shouldComponentUpdate실행후 render실행

        //return false; // shouldComponentUpdate실행후 render실행되지 않도록 약속되어 있다

        if(newProps.data === this.props.data){

            return false;

        }

        return true;

    }

    render(){

        console.log('TOC render');

        var lists = [];

        var data = this.props.data;

        for(var i=0 ; i<data.length ; i++){

            lists.push(

            <li key={data[i].id}>

                <a href={"/content/"+data[i].id}

                    data-id={data[i].id}

                    onClick={function(e){

                        e.preventDefault();

                        this.props.onChangePage(e.target.dataset.id);

                    }.bind(this)}>

                    {data[i].title}

                </a>

            </li>);

        }

        return (

            <nav>

                <ul>

                    {lists}

                </ul>

            </nav>

        );

    }

}

export default TOC

1. Create 구현 – immutable : shouldComponentUpdate를 사용하고 원본을 바꾸지 않고 setState에다 셋팅

* 배열은 from함수 이용하여 배열 복제

var a = [1,2];

var b = Array.from(a);

a === b;는 결과가 false

* 객체는 Object.assign 함수 이용하여 객체 복제

var a = {name:’홍’, age:20}

var b = Object.assign( {}, a); // a 객체 복제

b.name = ‘김’;

a, b

var c = Object.assign({address:’서울’}, a);

* Immutable 이용

<https://github.com/immutable-js/immutable-js> 참조

* App.js

import './App.css';

import React, {Component} from 'react';

import Subject from './components/Subject';

import TOC from './components/TOC';

import Control from './components/Control';

import ReadContent from './components/ReadContent';

import CreateContent from './components/CreateContent';

class App extends Component {

  constructor(props){

    super(props);

    this.max\_content\_id = 3; // UI

    this.state = {

      mode : 'read',

      selected\_content\_id : 1,

      subject : {title:'WEB', sub:'World Wide Web!'},

      welcome : {title:'welcome', desc:'Hello, React!!!'},

      contents : [

        {id:1, title:'HTML', desc:'HTML is for information'},

        {id:2, title:'CSS', desc:'CSS is for design'},

        {id:3, title:'JavaScript', desc:'JavaScript is for interactive'},

      ],

    };

  }

  render(){

    console.log('App render')

    var \_title, \_desc, \_article = null;

    if(this.state.mode ==='welcome'){

      \_title = this.state.welcome.title;

      \_desc = this.state.welcome.desc;

      \_article = <ReadContent title={\_title} desc={\_desc}></ReadContent>;

    }else if(this.state.mode === 'read'){

      //var content = this.state.contents[this.state.selected\_content\_id-1];

      //\_title = content.title;

      //\_desc = content.desc;

      for(var i=0 ; i<this.state.contents.length ; i++){

        var data = this.state.contents[i];

        if(data.id === this.state.selected\_content\_id){

          \_title = data.title;

          \_desc = data.desc;

          break;

        }

      }

      \_article = <ReadContent title={\_title} desc={\_desc}></ReadContent>;

    }else if(this.state.mode === 'create'){

      \_article = <CreateContent

                  onSubmit={function(\_title, \_desc){

                    // add content to this.state.contents

                    // alert(\_title + '/' + \_desc);

                    this.max\_content\_id ++;

                    //this.state.contents.push(

                    //  {id:this.max\_content\_id, title:\_title, desc:\_desc}

                    //);

                    //var \_contents = this.state.contents.concat(

                    //  {id:this.max\_content\_id, title:\_title, desc:\_desc}

                    //);

                    // 원본 복제

                    var newContents = Array.from(this.state.contents);

                    newContents.push(

                      {id:this.max\_content\_id, title:\_title, desc:\_desc}

                    );

                    this.setState({

                      //contents:this.state.contents

                      //contents : \_contents

                      contents : newContents // 원본을 교체

                    });

                  }.bind(this)}></CreateContent>;

    }else if(this.state.mode === 'update'){

      \_article = <h2>고치려고</h2>

    }else if(this.state.mode === 'delete'){

      \_article = <h2>지우려구</h2>

    }

    return (

      <div className="App">

        <Subject title={this.state.subject.title}

          sub={this.state.subject.sub}

          onChangePage={function(e){

            this.setState({

              mode : 'welcome'

            })

          }.bind(this)}></Subject>

        <TOC data={this.state.contents}

          onChangePage={function(id){

            this.setState({

              mode:'read',

              selected\_content\_id : Number(id)

            });

          }.bind(this)}></TOC>

        <Control

          onChangeMode={function(\_mode){

            //alert('mode를 수정 :',\_mode);

            this.setState({

              mode : \_mode

            });

          }.bind(this)}></Control>

        {\_article}

      </div>

    );

  }

}

export default App;

1. Update 구현 : Read 기능과 Create 기능 결합

<https://reactjs.org/docs/forms.html> 참조 : Docs -> Main Concepts -> 9.Forms의 내용에 근거

* CreateContent.js를 복사하여 수정 : UpdateContent.js

import React, {Component} from 'react';

class UpdateContent extends Component{

    constructor(props){

        super(props);

        this.state = {

            title : this.props.data.title,

            desc : this.props.data.desc

        }

    }

    render(){

        return (

            <article>

                <h2>Update</h2>

                <form action="/create\_process" method="post"

                    onSubmit={function(e){

                        e.preventDefault();

                        this.props.onSubmit(

                            e.target.title.value,

                            e.target.desc.value

                        );

                    }.bind(this)}

                >

                    <p>

                        <input

                            type="text"

                            name="title"

                            value = {this.state.title}

                            onChange={function(e){

                                //console.log(e.target.value);

                                this.setState({

                                    title:e.target.value

                                });

                            }.bind(this)}

                            //defaultValue={this.props.data.title}

                        />

                    </p>

                    <p><textarea name="desc"

                        value={this.state.desc}

                        onChange={function(e){

                            this.setState({

                                desc : e.target.desc

                            });

                        }.bind(this)}

                        // defaultValue={this.state.desc}

                        ></textarea>

                    </p>

                    <p><input type="submit"/></p>

                </form>

            </article>

        );

    }

}

export default UpdateContent;

* UpdateContent.js 리펙토링

import React, {Component} from 'react';

class UpdateContent extends Component{

    constructor(props){

        super(props);

        this.state = {

            id : this.props.data.id,

            title : this.props.data.title,

            desc : this.props.data.desc

        }

        this.inputFormHandler = this.inputFormHandler.bind(this);

    }

    inputFormHandler(e){

        this.setState({

            //[e.target.name] 최신 자바스크립트 문법

            [e.target.name]:e.target.value

        });

    }

    render(){

        return (

            <article>

                <h2>Update</h2>

                <form action="/update\_process" method="post"

                    onSubmit={function(e){

                        e.preventDefault();

                        this.props.onSubmit(

                            //e.target.id.value,

                            //e.target.title.value,

                            //e.target.desc.value

                            this.state.id,

                            this.state.title,

                            this.state.desc

                        );

                    }.bind(this)}

                >

                    <input type="hidden" name="id" value={this.state.id} readOnly="readOnly"></input>

                    <p>

                        <input

                            type="text"

                            name="title"

                            value = {this.state.title}

                            onChange={this.inputFormHandler}

                            //defaultValue={this.props.data.title}

                        />

                    </p>

                    <p><textarea name="desc"

                        value={this.state.desc}

                        onChange={this.inputFormHandler}

                        // defaultValue={this.state.desc}

                        ></textarea>

                    </p>

                    <p><input type="submit"/></p>

                </form>

            </article>

        );

    }

}

export default UpdateContent;

* App.js

import './App.css';

import React, {Component} from 'react';

import Subject from './components/Subject';

import TOC from './components/TOC';

import Control from './components/Control';

import ReadContent from './components/ReadContent';

import CreateContent from './components/CreateContent';

import UpdateContent from './components/UpdateContent';

class App extends Component {

  constructor(props){

    super(props);

    this.max\_content\_id = 3; // UI

    this.state = {

      mode : 'welcome',

      selected\_content\_id : 1,

      subject : {title:'WEB', sub:'World Wide Web!'},

      welcome : {title:'welcome', desc:'Hello, React!!!'},

      contents : [

        {id:1, title:'HTML', desc:'HTML is for information'},

        {id:2, title:'CSS', desc:'CSS is for design'},

        {id:3, title:'JavaScript', desc:'JavaScript is for interactive'},

      ],

    };

  }

  getReadContent(){ // 삭제할 수도 있으므로 이 로직 필요

    for(var i=0 ; i<this.state.contents.length ; i++){

      var data = this.state.contents[i];

      if(data.id === this.state.selected\_content\_id){

        return data;

      }

    }

  }

  getContent(){

    var \_title, \_desc, \_article = null;

    if(this.state.mode ==='welcome'){

      \_title = this.state.welcome.title;

      \_desc = this.state.welcome.desc;

      \_article = <ReadContent title={\_title} desc={\_desc}></ReadContent>;

    }else if(this.state.mode === 'read'){

      let \_content = this.getReadContent();

      \_article = <ReadContent title={\_content.title} desc={\_content.desc}></ReadContent>;

    }else if(this.state.mode === 'create'){

      \_article = <CreateContent

                  onSubmit={function(\_title, \_desc){

                    // add content to this.state.contents

                    // alert(\_title + '/' + \_desc);

                    this.max\_content\_id ++;

                    //this.state.contents.push(

                    //  {id:this.max\_content\_id, title:\_title, desc:\_desc}

                    //);

                    //var \_contents = this.state.contents.concat(

                    //  {id:this.max\_content\_id, title:\_title, desc:\_desc}

                    //);

                    // 원본 복제

                    var newContents = Array.from(this.state.contents);

                    newContents.push(

                      {id:this.max\_content\_id, title:\_title, desc:\_desc}

                    );

                    this.setState({

                      //contents:this.state.contents

                      //contents : \_contents

                      contents : newContents, // 원본을 교체

                      mode : 'read',

                      selected\_content\_id : this.max\_content\_id

                    });

                  }.bind(this)}></CreateContent>;

    }else if(this.state.mode === 'update'){

      let \_content = this.getReadContent();

      //console.log(\_content);

      \_article = <UpdateContent

                  data={\_content}

                  onSubmit={function(\_id, \_title, \_desc){

                    var \_contents = Array.from(this.state.contents); // 복제. 원본 불변 : 성능 튜닝할지 모르니 이렇게 하자

                    var i;

                    for(i=0; i<\_contents.length ; i++){

                      if(\_contents[i].id === Number(\_id)){

                        \_contents[i] = {

                          id:\_id,

                          title:\_title,

                          desc : \_desc

                        };

                        break;

                      }

                    }

                    console.log(i,'번째 :', \_contents)

                    this.setState({

                      contents : \_contents, // 원본을 교체

                      mode : 'read'

                    });

                  }.bind(this)}

          ></UpdateContent>

    }else if(this.state.mode === 'delete'){

      \_article = <h2>지울거예요</h2>

    }

    return \_article;

  }

  render(){

    return (

      <div className="App">

        <Subject title={this.state.subject.title}

          sub={this.state.subject.sub}

          onChangePage={function(e){

            this.setState({

              mode : 'welcome'

            })

          }.bind(this)}></Subject>

        <TOC data={this.state.contents}

          onChangePage={function(id){

            this.setState({

              mode:'read',

              selected\_content\_id : Number(id)

            });

          }.bind(this)}></TOC>

        <Control

          onChangeMode={function(\_mode){

            //alert('mode를 수정 :',\_mode);

            this.setState({

              mode : \_mode

            });

          }.bind(this)}></Control>

        {this.getContent()}

      </div>

    );

  }

}

export default App;

1. Delete 구현

* App.js

import './App.css';

import React, {Component} from 'react';

import Subject from './components/Subject';

import TOC from './components/TOC';

import Control from './components/Control';

import ReadContent from './components/ReadContent';

import CreateContent from './components/CreateContent';

import UpdateContent from './components/UpdateContent';

class App extends Component {

  constructor(props){

    super(props);

    this.max\_content\_id = 3; // UI

    this.state = {

      mode : 'welcome',

      selected\_content\_id : 0,

      subject : {title:'WEB', sub:'World Wide Web!'},

      welcome : {title:'welcome', desc:'Hello, React!!!'},

      contents : [

        {id:1, title:'HTML', desc:'HTML is for information'},

        {id:2, title:'CSS', desc:'CSS is for design'},

        {id:3, title:'JavaScript', desc:'JavaScript is for interactive'},

      ],

    };

  }

  getReadContent(){ // 삭제할 수도 있으므로 이 로직 필요

    for(var i=0 ; i<this.state.contents.length ; i++){

      var data = this.state.contents[i];

      if(data.id === this.state.selected\_content\_id){

        return data;

      }

    }

  }

  getContent(){

    var \_title, \_desc, \_article = null;

    if(this.state.mode ==='welcome'){

      \_title = this.state.welcome.title;

      \_desc = this.state.welcome.desc;

      \_article = <ReadContent title={\_title} desc={\_desc}></ReadContent>;

    }else if(this.state.mode === 'read'){

      let \_content = this.getReadContent();

      \_article = <ReadContent title={\_content.title} desc={\_content.desc}></ReadContent>;

    }else if(this.state.mode === 'create'){

      \_article = <CreateContent

                  onSubmit={function(\_title, \_desc){

                    // add content to this.state.contents

                    // alert(\_title + '/' + \_desc);

                    this.max\_content\_id ++;

                    //this.state.contents.push(

                    //  {id:this.max\_content\_id, title:\_title, desc:\_desc}

                    //);

                    //var \_contents = this.state.contents.concat(

                    //  {id:this.max\_content\_id, title:\_title, desc:\_desc}

                    //);

                    // 원본 복제

                    var newContents = Array.from(this.state.contents);

                    newContents.push(

                      {id:this.max\_content\_id, title:\_title, desc:\_desc}

                    );

                    this.setState({

                      //contents:this.state.contents

                      //contents : \_contents

                      contents : newContents, // 원본을 교체

                      mode : 'read',

                      selected\_content\_id : this.max\_content\_id

                    });

                  }.bind(this)}></CreateContent>;

    }else if(this.state.mode === 'update'){

      let \_content = this.getReadContent();

      //console.log(\_content);

      \_article = <UpdateContent

                  data={\_content}

                  onSubmit={function(\_id, \_title, \_desc){

                    var \_contents = Array.from(this.state.contents); // 복제. 원본 불변 : 성능 튜닝할지 모르니 이렇게 하자

                    var i;

                    for(i=0; i<\_contents.length ; i++){

                      if(\_contents[i].id === Number(\_id)){

                        \_contents[i] = {

                          id:\_id,

                          title:\_title,

                          desc : \_desc

                        };

                        break;

                      }

                    }

                    //console.log(i,'번째 교제함 : ')

                    this.setState({

                      contents : \_contents, // 원본을 교체

                      mode : 'read'

                    });

                  }.bind(this)}

          ></UpdateContent>

    }

    return \_article;

  }

  render(){

    return (

      <div className="App">

        <Subject title={this.state.subject.title}

          sub={this.state.subject.sub}

          onChangePage={function(e){

            this.setState({

              mode : 'welcome'

            })

          }.bind(this)}></Subject>

        <TOC data={this.state.contents}

          onChangePage={function(id){

            this.setState({

              mode:'read',

              selected\_content\_id : Number(id)

            });

          }.bind(this)}></TOC>

        <Control

          onChangeMode={function(\_mode){

            //alert('mode를 수정 :',\_mode);

            if(\_mode === 'delete'){

              if(this.state.selected\_content\_id !== 0 &&

                window.confirm('정말로 삭제하시겠습니까?')){

                var \_contents = Array.from(this.state.contents);

                var i;

                for(i=0 ; i<\_contents.length ; i++){

                  if(\_contents[i].id === this.state.selected\_content\_id){

                    \_contents.splice(i, 1); // i번째 1개를 지움

                    alert(this.state.selected\_content\_id + '번째 삭제 완료');

                    break;

                  }

                }

                this.setState({

                  contents : \_contents,

                  mode : 'welcome',

                  selected\_content\_id:0

                });

              }else if(this.state.selected\_content\_id === 0){

                alert('삭제를 원하는 데이터를 선택후 삭제하세요');

              }else{

                alert('안 지웠어요');

              }

            }else {

              this.setState({

                mode : \_mode

              });

            }

          }.bind(this)}></Control>

        {this.getContent()}

      </div>

    );

  }

}

export default App;

1. 추가하면 좋은 item
   * Immutable

<https://immutable-js.com/> 참조

* + Router

하나의 url로 모든 페이지를 access하는 문제 해결.

url에 따른 다른 ui나오도록 도와주는 것을 도와줌

<https://reactrouter.com/en/main> 참조

npm으로 설치해서 사용하는 플러그인

* + Create-react-app

npm run eject : 기본 도구만으로 부족할 경우 감춰진 여러 설정을 수정할 수 있다. 개발환경 수정을 할 수 있으나 돌아갈 수 없음

redux : 컴포넌트가 많아지면 컴포넌트들간 교류가 까다로워짐. 중앙에 데이터 저장소를 하나 만들어 모든 컴포넌트는 그 중앙의 저장소와 직접 연결.

* + React server side rendering

서버쪽에서 웹 페이지를 완성한 후 클라이언트로 완성된 html을 전송하는 걸로 어플리케이션을 구동