**9. 배열 렌더링 하기**

아래의 해당 배열을 렌더링 해보자!

const users = [

{

id: 1,

username: 'velopert',

email: 'public.velopert@gmail.com'

},

{

id: 2,

username: 'tester',

email: 'tester@example.com'

},

{

id: 3,

username: 'liz',

email: 'liz@example.com'

}

];

**UserList.js** (그냥 그대로 코드를 작성할 경우)

import React from 'react';

function UserList() {

    const users = [

        {

          id: 1,

          username: 'velopert',

          email: 'public.velopert@gmail.com'

        },

        {

          id: 2,

          username: 'tester',

          email: 'tester@example.com'

        },

        {

          id: 3,

          username: 'liz',

          email: 'liz@example.com'

        }

      ];

      return (

          <div>

              <div>

                  <b>{users[0].username}</b> <span>({users[0].email})</span>

              </div>

              <div>

                  <b>{users[1].username}</b> <span>({users[1].email})</span>

              </div>

              <div>

                  <b>{users[2].username}</b> <span>({users[2].email})</span>

              </div>

          </div>

      )

}

export default UserList;

**UserList.js** (컴포넌트를 재사용할 수 있도록 만들기)

import React from 'react';

function User({user}){

    return(

        <div>

            <b>{user.username}</b> <span>({user.email})</span>

        </div>

    );

}

function UserList() {

    const users = [

        {

          id: 1,

          username: 'velopert',

          email: 'public.velopert@gmail.com'

        },

        {

          id: 2,

          username: 'tester',

          email: 'tester@example.com'

        },

        {

          id: 3,

          username: 'liz',

          email: 'liz@example.com'

        }

      ];

      return(

          <div>

              <User user = {users[0]} />

              <User user = {users[1]} />

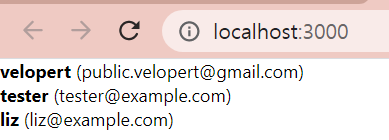
              <User user = {users[2]} />

          </div>

      )

}

export default UserList;

App 에 렌더링 해보면? 🡺 

* **동적인 배열의 렌더링을 위한 map() 함수 사용**
* 배열이 고정적이라면 상관이 없지만, 이렇게 배열의 인덱스를 하나하나 조회해가며 렌더링하는 방법은 동적인 배열을 렌더링하지 못한다.
* 이때 동적인 배열을 렌더링하기 위해 자바스크립트 배열의 내장함수 map()을 사용한다.
* map() : 배열안의 각 원소를 변환하여 새로운 배열을 만들어 줌. 이 함수를 사용하여 일반 데이터 배열을 리액트 엘리먼트로 이루어진 배열로 변환해주기
* map함수는 callback함수를 실행한 결과를 가지고 새로운 배열을 만들 때 사용한다.

callback함수는 currentValue(뭐든 가능), index, array 3개의 파라미터를 갖는다.

**UserList.js** (map 함수 이용)

function User({user}){

    return(

        <div>

            <b>{user.username}</b> <span>({user.email})</span>

        </div>

    );

}

return(

          <div>

              {users.map(user => (

                  <User user = {user} />

              ))}

          </div>

      )

* 이렇게 바꿔주면 된다 ,, 하지만 에러 🡪 key 설정해주기
* **key라는 props를 설정하기**
* 리액트에서 배열을 렌더링 할 때는 이를 설정해 주어야한다.
* key 값을 각 원소들마다 가지고 있는 고유 값으로 설정해야한다. 지금의 경우 id가 고유값

return(

          <div>

              {users.map(user => (

                  <User user = {user} key = {user.id} />

              ))}

          </div>

      )

* 이렇게 key를 설정해준다.
* 만약 배열안의 원소가 가지고 있는 고유한 값이 없다면?
* map() 함수를 사용할 때 설정하는 콜백함수의 두번째 파라미터 index를 key로 사용하기

ex)

return(

          <div>

              {users.map((user,index) => (

                  <User user = {user} key = {index} />

              ))}

          </div>

      )

\* 배열을 렌더링 할 때에는 고유한 key의 값이 있는 것이 중요하며, 만약에 배열 안에 중복되는 key 가 있을 때에는 렌더링 시에 오류메시지가 콘솔에 나타나게 되며, 업데이트가 제대로 이루어지지 않게 된다!