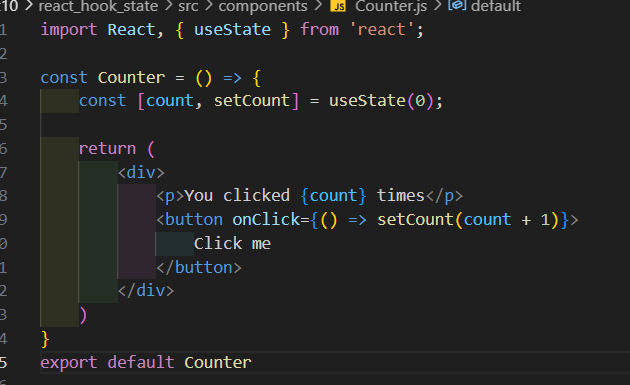
**React Hook**

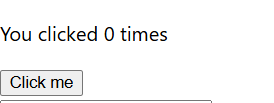
**By traltb@fe.edu.vn**

1. **useState**

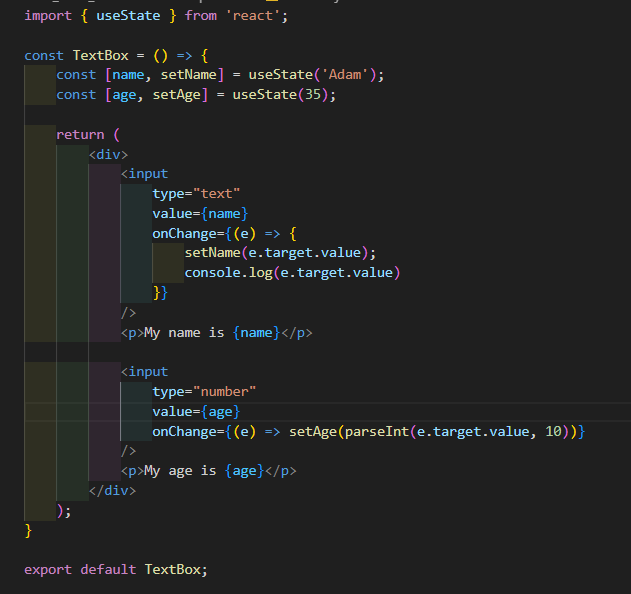
* Dùng để khai báo state trong functional component.
* Trả về một cặp giá trị: giá trị hiện tại của state và một hàm để cập nhật state.

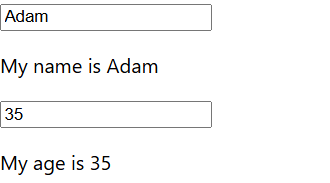
**Ví dụ 1: cập nhật số lần bấm nút**

****

****

**Ví dụ 2: Cập nhật giá trị name, age cho textbox**

****

****

Giải thích:

**<input type="text" value={name} onChange={(e) => { ... }} />**

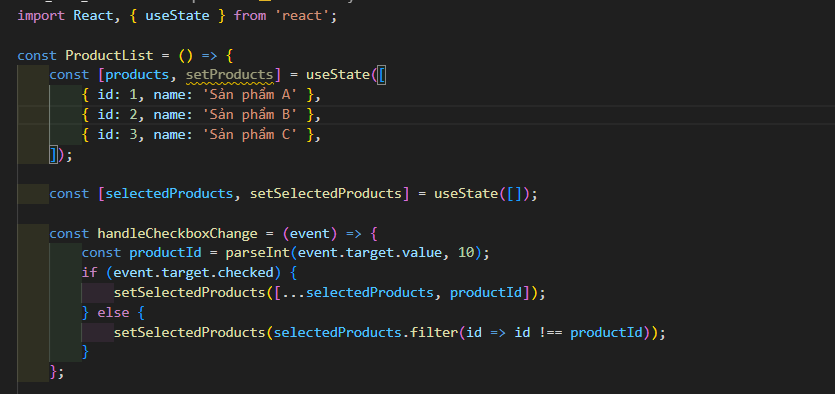
* **type="text"**: Đây là một input text.
* **value={name}**: Giá trị của input được liên kết với state variable name. Điều này có nghĩa là khi giá trị của name thay đổi, giá trị hiển thị trong input cũng sẽ thay đổi theo. Đây gọi là *controlled component*.
* **onChange={(e) => { ... }}**: Đây là một event handler. Nó được gọi mỗi khi người dùng thay đổi giá trị trong input. e là event object, chứa thông tin về sự kiện.
  + **setName(e.target.value)**: Đây là cách chúng ta cập nhật state name. e.target.value lấy giá trị hiện tại của input và setName() cập nhật state name với giá trị đó.
  + **console.log(e.target.value)**: In giá trị hiện tại của input ra console.

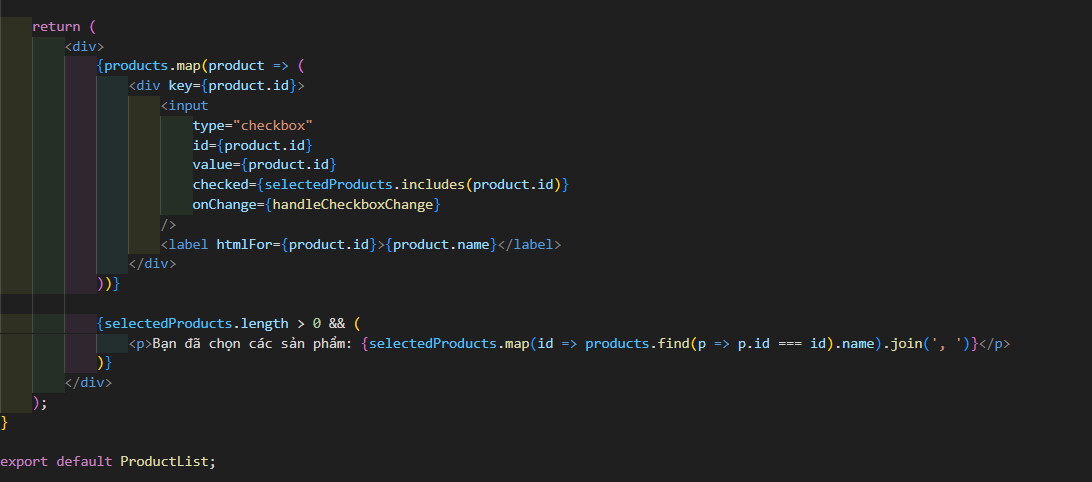
**<input type="number" value={age} onChange={(e) => setAge(parseInt(e.target.value, 10))} />**

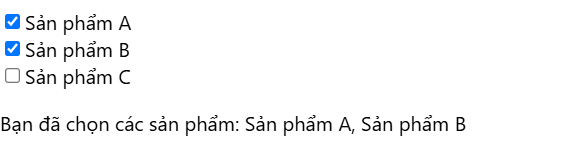
Tương tự như input text, nhưng đây là input number.

* **parseInt(e.target.value, 10)**: Vì e.target.value luôn là một chuỗi (string), chúng ta cần chuyển nó thành số nguyên bằng parseInt. Tham số 10 chỉ định hệ cơ số 10. Việc này quan trọng vì state age được định nghĩa là một số.

**Ví dụ 3: Hiển thị các sản phẩm được chọn khi click vào các checkbox**







**Giải thích:**

**const [products, setProducts] = useState([ ... ]);**

* Khai báo state products bằng hook useState. products là một mảng chứa thông tin về các sản phẩm. Giá trị ban đầu của products là một mảng các object, mỗi object đại diện cho một sản phẩm với id và name.
* setProducts là một function được dùng để cập nhật state products.

**const [selectedProducts, setSelectedProducts] = useState([]);**

* Khai báo state selectedProducts bằng hook useState. selectedProducts là một mảng chứa id của các sản phẩm đã được chọn. Ban đầu, không có sản phẩm nào được chọn, nên giá trị ban đầu là một mảng rỗng [].
* setSelectedProducts là function để cập nhật state selectedProducts.

**const handleCheckboxChange = (event) => { ... };**

* Định nghĩa function handleCheckboxChange để xử lý sự kiện onChange của checkbox. event là object chứa thông tin về sự kiện.
* const productId = parseInt(event.target.value, 10);: Lấy id của sản phẩm từ giá trị (value) của checkbox. event.target.value trả về một chuỗi, vì vậy cần dùng parseInt để chuyển nó thành số nguyên (hệ cơ số 10).
* if (event.target.checked) { ... } else { ... }: Kiểm tra xem checkbox đã được chọn hay chưa.
  + **Nếu được chọn (event.target.checked là true):**
    - setSelectedProducts([...selectedProducts, productId]);: Tạo một *bản sao* của mảng selectedProducts hiện tại bằng spread operator (...) và thêm productId vào cuối mảng. Việc tạo bản sao rất quan trọng vì trong React, bạn cần tạo một *mới* object/array để kích hoạt re-render.
  + **Nếu bỏ chọn (event.target.checked là false):**
    - setSelectedProducts(selectedProducts.filter(id => id !== productId));: Sử dụng filter để tạo một *mảng mới* chỉ chứa các id *khác* với productId. Nói cách khác, nó loại bỏ productId khỏi mảng selectedProducts.

**{products.map(product => ( ... ))}**

* Sử dụng phương thức map để lặp qua mảng products và tạo một phần tử JSX cho mỗi sản phẩm.
* **key={product.id}**: *Bắt buộc* phải có key prop cho mỗi phần tử được tạo ra bởi map. key giúp React xác định các phần tử một cách duy nhất khi danh sách thay đổi. Nên sử dụng một giá trị duy nhất như product.id.

**<div key={product.id}> ... </div>**

* Bọc mỗi sản phẩm trong một div.

**<input type="checkbox" ... />**

* Định nghĩa một checkbox cho mỗi sản phẩm.
* id={product.id}: Gán id của sản phẩm cho id của checkbox.
* value={product.id}: Gán id của sản phẩm cho value của checkbox. Giá trị này được sử dụng trong handleCheckboxChange để xác định sản phẩm nào được chọn/bỏ chọn.
* checked={selectedProducts.includes(product.id)}: Kiểm tra xem sản phẩm hiện tại đã được chọn hay chưa bằng cách kiểm tra xem product.id có tồn tại trong mảng selectedProducts hay không. Nếu có, checkbox sẽ được đánh dấu. Đây là một ví dụ về *controlled component* trong React.
* onChange={handleCheckboxChange}: Gán function handleCheckboxChange làm handler cho sự kiện onChange của checkbox.

**<label htmlFor={product.id}>{product.name}</label>**

* Label cho checkbox. htmlFor phải khớp với id của checkbox để liên kết chúng lại.

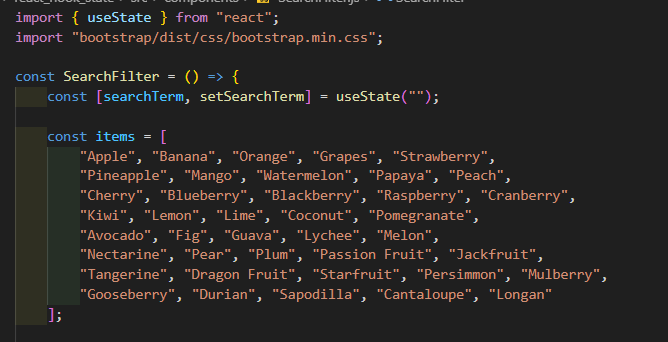
**{selectedProducts.length > 0 && ( ... )}**

* Hiển thị thông báo về các sản phẩm đã được chọn *chỉ khi* có ít nhất một sản phẩm được chọn (selectedProducts.length > 0). Đây là cách viết rút gọn của một biểu thức điều kiện.

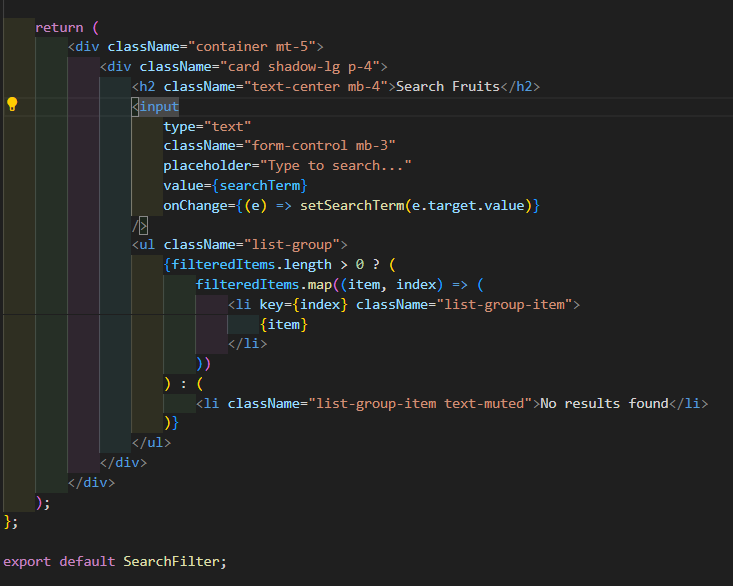
**<p>Bạn đã chọn các sản phẩm: { ... }</p>**

* Hiển thị danh sách các sản phẩm đã được chọn.
* selectedProducts.map(id => products.find(p => p.id === id).name).join(', '): Lặp qua mảng selectedProducts (chứa các id), tìm tên của sản phẩm tương ứng trong mảng products bằng find, và sau đó nối các tên lại với nhau bằng dấu phẩy.
  1. Search Filter

Creating a simple Search Filter component that allows users to filter a list of items based on their search input.



Import useState từ react, tạo component SearchFilter bằng arrow function, tạo searchTerm, tạo danh sách các items



Lọc danh sách dựa trên searchTerm  
items.filter(): lọc các mục trong danh sách  
.toLowerCase(): chuyển hết thành chữ thường  
.includes(searchTerm.toLowerCase(): kiểm tra item có từ khóa searchTerm không  
  
Input để nhập từ khóa  
value={searchTerm}: giá trị của input luôn bằng searchTerm  
onChange={(e) => setSearchTerm(e.target.value)}  
khi người dùng nhập giá trị vào searchTerm sẽ được cập nhập, filteredItems sẽ cập nhật theo từ khóa

Hiển thị kết quả đã nhập bằng ul li  
  
