**1. Создание контекста с фильтром**

Сначала создаём контекст, который будет хранить состояние фильтра и функцию для его изменения:

import React, { createContext, useContext, useState } from 'react';

// Создаём контекст

const FilterContext = createContext();

// Создаём провайдер контекста

export const FilterProvider = ({ children }) => {

  const [filter, setFilter] = useState(''); // Состояние фильтра

  return (

    <FilterContext.Provider value={{ filter, setFilter }}>

      {children}

    </FilterContext.Provider>

  );

};

// Хук для удобного использования контекста

export const useFilter = () => useContext(FilterContext);

**2. Создание компонента для инпута**

Инпут будет записывать значение в setFilter, который мы получили из контекста.

import React from 'react';

import { useFilter } from './FilterContext';

const Input = () => {

  const { setFilter } = useFilter(); // Достаём функцию setFilter из контекста

  const handleInputChange = (event) => {

    setFilter(event.target.value); // Записываем значение инпута в контекст

  };

  return (

    <input

      type="text"

      placeholder="Фильтровать"

      onChange={handleInputChange} // Обработчик для обновления фильтра

    />

  );

};

**3. Создание компонента для списка товаров**

Этот компонент будет использовать значение filter из контекста, чтобы отфильтровать список товаров.

import React from 'react';

import { useFilter } from './FilterContext';

const ProductList = () => {

  const { filter } = useFilter(); // Достаём значение фильтра из контекста

  const products = [

    'Apple',

    'Orange',

    'Banana',

    'Pineapple',

    'Watermelon',

    'Grapes',

    'Strawberry'

  ]; // Пример списка товаров

  // Фильтрация товаров

  const filteredProducts = products.filter((product) =>

    product.toLowerCase().includes(filter.toLowerCase()) // Сравнение с фильтром

  );

  return (

    <ul>

      {filteredProducts.map((product, index) => (

        <li key={index}>{product}</li>

      ))}

    </ul>

  );

};

**4. Композиция приложения**

Теперь объединяем всё вместе, используя провайдер контекста FilterProvider для обёртки компонентов.

import React from 'react';

import { FilterProvider } from './FilterContext';

import Input from './Input';

import ProductList from './ProductList';

const App = () => {

  return (

    <FilterProvider>

      <Input /> {/\* Инпут для фильтрации \*/}

      <ProductList /> {/\* Список товаров \*/}

    </FilterProvider>

  );

};

export default App;

Как всё работает:

Контекст React (FilterContext) используется для глобального хранения состояния фильтра (filter) и функции обновления (setFilter).

Компонент ` обновляет filter` в контексте при вводе текста.

Компонент ` берёт это состояние filter` из контекста и использует его для фильтрации товаров.

Провайдер FilterProvider обеспечивает доступ к filter и setFilter для всех вложенных компонентов.

Результат:

Когда пользователь вводит текст в инпут, список товаров автоматически обновляется с учётом фильтрации.

Контекст позволяет избежать проброса пропсов между компонентами.