Universidad Juárez del Estado de Durango

Facultad de Ingeniería Ciencias y Arquitectura

Ingeniería en Tecnologías Computacionales

Programación Web

Reporte de Practica 24

Diego Rea Morales

Ing. Fabian Gallegos Gutiérrez

7 de Noviembre del 2025

Índice

Contenido

La práctica trata de crear una lista, la cual, esta debe de enumerar la cantidad de elementos que la lista contiene.

A su vez, de botones para limpiar la lista actual y también, eliminar de forma individual los elementos agregados a la lista.

```jsx
import React, { useState } from "react";
import "./App.css";
function App() {
  const [inputValue, setInputValue] = useState("");
  const [listItems, setListItems] = useState([]);
  const handleInputChange = (event) => {
    setInputValue(event.target.value);
  };
  const handleAddItem = () => {
    if (inputValue.trim() !== "") {
      setListItems([...listItems, inputValue]);
      setInputValue("");
    }
  };
  const handleDeleteList = () => {
    setListItems([]);
  };
  const deleteElement = (index) => {
    const updList = listItems.filter((_, i) => i !== index);
    setListItems(updList);
  };
  return (
    <div className="App">
      <div className="input-container">
        <input
          type="text"
          value={inputValue}
          onChange={handleInputChange}
          placeholder="Ingrese un elemento"
        />
        <button onClick={handleAddItem}>Agregar item</button>
        <button onClick={handleDeleteList}>Borrar lista</button>
      </div>
      <div className="list-container">
        <ol>
          {listItems.map((item, index) => (
            <div className="conteiner-datos">
              <div className="datos">
                <li key={index}>{item}</li>
              </div>
              <div className="boton">
                <button
                  className="btn-delete"
                  onClick={() => {
                    deleteElement(index);
                  }}
                >
                  Borrar Dato
                </button>
              </div>
            </div>
          ))}
        </ol>
      </div>
    </div>
  );
}
export default App;
```

```css
.App {
  display: flex;
  flex-direction: row;
  justify-content: space-around;
  align-items: flex-start;
  padding: 20px;
}
.input-container {
  width: 50%;
}
.input-container input {
  width: calc(100% - 130px);
  margin-right: 10px;
  padding: 8px;
  font-size: 16px;
}
.input-container button, .btn-delete {
  padding: 8px 16px;
  font-size: 16px;
  cursor: pointer;
}
.list-container {
  width: 50%;
}
.conteiner-datos{
  display: grid;
  grid-template-columns: repeat(2, 1fr);
  margin: 10px;
}
.list-container ol {
  padding-left: 20px;
}
.list-container li {
  margin-bottom: 10px;
}
.datos{
  grid-column: 1 / 2;
}
.boton{
  grid-column: 2 / 3;
}
.btn-delete{
  padding: 4px;
}
```

Explicación del código JavaScript

```
const [inputValue, setInputValue] = useState("");
const [listItems, setListItems] = useState([]);
```

Primero declaramos dos *hooks*, uno para el valor del elemento que se va agregar y es un arreglo el cuál será la lista, de donde se eliminará y agregaran elementos.

```
const handleInputChange = (event) => {
    setInputValue(event.target.value);
};
```

Este evento se encarga de asignar un valor al elemento que después se agregará a la lista.

```
<input
          type="text"
          value={inputValue}
          onChange={handleInputChange}
          placeholder="Ingrese un elemento"
/>
```

Este evento es ejecutado en el input encargado de recibir el valor a agregar.

```
const handleAddItem = () => {
    if (inputValue.trim() !== "") {
      setListItems([...listItems, inputValue]);
      setInputValue("");
    }
  };
```

Después, cuando el usuario presione el botón de agregar, este ejecutara este evento, el cual, tiene como contenido una condición. Esta evalúa si es que, realmente se haya agregado algo, si no es así, simplemente no se ejecuta y acaba la ejecución del bloque de código, si es verdadera, es decir, que si tenga un contenido dentro de la variable, este lo que va a realizar es, setear la lista, agregando, los valores de la lista (si es que hay), y agrega el valor que el usuario ingreso. Todo esto para que después, el valor de entrada se setee vacío, para que no se quede guardado el ultimo valor ingresado.

```
const handleDeleteList = () => {
    setListItems([]);
  };
```

El evento de eliminar lista lo que realiza es simplemente setear la lista con un arreglo vacío.

```
  const deleteElement = (index) => {
    const updList = listItems.filter((_, i) => i !== index);
    setListItems(updList);
  };
```

Para el evento de eliminar un elemento de la lista se tiene este bloque de código, el cual toma como parámetro el índice del elemento a eliminar, este es provisto por el botón de eliminar.

```
<div className="list-container">
        <ol>
          {listItems.map((item, index) => (
            <div className="conteiner-datos">
              <div className="datos">
                <li key={index}>{item}</li>
              </div>
              <div className="boton">
                <button
                  className="btn-delete"
                  onClick={() => {
                    deleteElement(index);
                  }}
                >
                  Borrar Dato
                </button>
              </div>
            </div>
          ))}
        </ol>
      </div>
```

Este al dar como argumento el índice, el evento de eliminar lo utilizara en un método llamado *filter* el cual lo ejecutaremos con la lista, y toma como arguemento un parámetro (que en nuestro caso no lo necesitamos, por eso el _) y el índice del elemento, para después una comparación que dice que si el índice provisto no es igual al índice del elemento recorrido dentro del arreglo, este lo guardará, mientras que si el índice es igual al del elemento, este no se guardará dentro de una nueva variable la cual se usará de manera temporal, para después setear la lista ingresándole esta nueva variable la cual contendrá todos los elementos que la lista ya tenía a excepción del elemento eliminado.

Conclusiones

En este caso, el objetivo principal de la práctica tuvo como principal intención el poder trabajar con arreglos en React, utilizándolos de manera dinámica, es decir, aumentando su tamaño y decreciéndolo. Esto no solo refuerza el conocimiento en React, sino que también con JavaScript, ya que utilizar de forma adecuada los arreglos nos puede ayudar en temas de optimización.