Step 7: Adding an individual item remover and list clear to the List component

# Introduction:

In the previous exercise we created a component that lists items submitted by a form. Now we’ll continue to add a function to wipe the list and a remove handler that allows us to remove **individual rows** from the list.

# Clearing the array:

Now we have a working list, but nothing to clear it with other than refreshing the page. So, let’s start with adding a button to wipe the list.

This will be a simple task. We will use our existing setContent state and onClick that we’re already familiar with. Let’s go under our form and **add a button**. You can name it something like **Clear list**:

<button>Clear list</button>

Now we’ll create the function that we’ll address to the button. Above our components return tag, create a function called **clearList**. Inside, we simply set the state to an **empty array**:

function clearList(){

    setContent([]);

}

Now all that’s left to do is assign the function to our button using onClick.

<button onClick={clearList}>Clear list</button>

Graphical user interface, text, application

Description automatically generated Graphical user interface, text, application

Description automatically generated

Now the button should be visible and working.

# Individual remove buttons:

Next, we’ll move to the more challenging part. That is the **individual item remover**.

We will use .filter function that is also just a JavaScript function. We will need it to get the **item index** of the item we want to remove.

Start by creating a button named **Remove**. This time we’re going to insert it **into the map list**, inside the list item so every row should have its own button:

content.map((item) => (

    <ul>

        <li>

          <h2>{item}</h2>

          <button>Remove</button>

        </li>

    </ul>

)

Graphical user interface, text, application

Description automatically generated

# Remove handler:

Next, we have to edit our list a bit. We will **assign a key** to our list item. Keys help React identify which items have changed, are added, or are removed. We will set the key as the arrays **item index**. We have to add it to two places, in the map itself as an argument, and **inside** our h2 tag, like shown below:

content.map((item, index) => (

    <ul>

        <li>

            <h2 key={index}>{item}</h2>

            <button>Remove</button>

        </li>

    </ul>

)

Now we will build our **remove handler**. We will place it at the top of our clearList function. Our handler is going to be an **arrow function**. We will pass **removeIndex** as our handlers argument:

const removeHandler = (removeIndex) => {

}

Inside, we will need a **second** arrow function, this one will be using the JavaScript .filter function. We will name it **removeItem**. We will use the filter method on the content array and give it the **arrays item**, and **item index** as an argument:

const removeHandler = (removeIndex) => {

    const removeItem = content.filter((item, index) => {

    });

}

This function will return a **boolean value**. We will compare the indexes of items using the **inequality operator(!==)**.

After the removeItem function has removed the item with the item index, the removeHandler updates the content state as the new array.

const removeHandler = (removeIndex) => {

    const removeItem = content.filter((item, index) => {

        return removeIndex !== index;

    });

    setContent(removeItem);

}

Now our handler should work, all that’s left to do is assign it to our button in the map list. **This will be little different as we’re now passing an arrow function.**

You’ll pass an arrow function to a button like in the example below:

\*\*\*This is an example\*\*\*

<button onClick={() => functionName(argument)}></button>

Now we’ll add it to the list item button using our removeHandler with our item index as an argument:

content.map((item, index) => (

    <ul>

        <li>

             <h2 key={index}>{content}</h2>

             <button onClick={() => removeHandler(index)}>Remove</button>

        </li>

    </ul>

)

Open your browser and test if you can remove individual rows from the list. It should work like shown below:

Graphical user interface, text, application

Description automatically generatedGraphical user interface, text, application

Description automatically generated

We have now completed all our individual components! You should have a general understanding on how React works. Next, we will start building our **Tic-Tac-Toe game** using the features you’ve learned. **See you there!** 😊