WSOA4175A:
Interactive Media 4A | React Exercises Solutions

Note:

This is **one** method of completing the following exercises, not the only method. How you choose to answer will depend on how you interpret the question, but the logic should remain roughly the same.

Exercise 1:

Rendering a List of Names

- Create a React component called `NameList`.
- Use the following array of names inside the component:

```
const names = ["Alice", "Bob", "Charlie", "Diana"];
```

- Render the names as an unordered list (`<ul>`), with each name inside a separate `<li>` element.
- Ensure each `<li>` has a unique `key` using the index of the array.

```jsx
import React from "react";

function NameList() {
  const names = ["Alice", "Bob", "Charlie", "Diana"];

  return (
    <div className="p-4">
      <h2 className="text-lg font-bold mb-2">Name List</h2>
      <ul className="list-disc pl-5">
        {names.map((name, index) => (
          <li key={index} className="text-blue-600">{name}</li>
        ))}
      </ul>
    </div>
  );
```

```
}

export default NameList;
```

Exercise 2:

Simple Input Form

- Create a React component called `SimpleForm`.
- Add an input field and a submit button to the form.
- When the user types in the input field, update the state to reflect the current value of the input.
- When the form is submitted, display an alert with the entered value, and clear the input field.
- Use `useState` to manage the input value.

```
import React, { useState } from "react";

function SimpleForm() {
  const [inputValue, setInputValue] = useState("");

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

  const handleSubmit = (e) => {
    e.preventDefault();
    alert(`You entered: ${inputValue}`);
    setInputValue(""); // Clear the input field
  };

  return (
    <div className="p-4">
      <h2 className="text-lg font-bold mb-2">Simple Input Form</h2>
      <form onSubmit={handleSubmit} className="space-y-2">
        <input
          type="text"
          value={inputValue}
```

```
        onChange={handleInputChange}
        placeholder="Type something..."
        className="border border-gray-300 p-2 rounded"
      />
      <button
        type="submit"
        className="bg-blue-600 text-white px-4 py-2 rounded hover:bg-blue-700"
      >
        Submit
      </button>
    </form>
  </div>
 );
}

export default SimpleForm;
```

Exercise 3:

Task List with Dynamic Entries

- Create a React component called `TaskList`.
- Add an input field and a button.
- Allow users to type in a task and click the button to add the task to a list.
- Render the list of tasks dynamically, and ensure each task has a unique `key` based on its **index**.
- Style the tasks with a border, padding, and margin for better visual separation.

```
import React, { useState } from "react";

function TaskList() {
  const [tasks, setTasks] = useState([]);
  const [task, setTask] = useState("");
```

```
const handleInputChange = (e) => {
  setTask(e.target.value);
};

const addTask = () => {
  if (task.trim() !== "") {
    setTasks([...tasks, task]);
    setTask(""); // Clear the input field after adding the task
  }
};

return (
  <div className="p-4">
    <h2 className="text-lg font-bold mb-2">Task List</h2>
    <div className="space-y-2">
      <input
        type="text"
        value={task}
        onChange={handleInputChange}
        placeholder="Enter a task"
        className="border border-gray-300 p-2 rounded w-full"
      />
      <button
        onClick={addTask}
        className="bg-green-600 text-white px-4 py-2 rounded hover:bg-green-700"
      >
        Add Task
      </button>
    </div>
    <ul className="mt-4 space-y-2">
      {tasks.map((task, index) => (
        <li
          key={index}
          className="border border-gray-300 p-2 rounded bg-gray-50"
        >
          {task}
        </li>
      ))}
    </ul>
  </div>
```

```
  );
}

export default TaskList;
```