UCS 2611 INTERNET PROGRAMMING LAB
EXERCISE 10. TODO WEB APPLICATION USING REACTJS

NAME: Keerthana.S
REG NO: 3122 22 5001 061
CLASS: CSE 'B'

Write index.html to retrieve the name of the user. Write a Ajax to refresh the index.html with the welcome message along with the name when the submit button is clicked. [CO1, K3]

Best Practices to be followed:

1. Design before coding
2. Incremental coding
3. Usage of proper naming convention
4. Usage of Comments to the code
5. Indentation of code

Index.js:

```
import React from 'react'; import ReactDOM from
'react-dom/client'; import './index.css'; import
App from './App'; import reportWebVitals from
'./reportWebVitals';

const root =
ReactDOM.createRoot(document.getElementById('root')); root.render(
  <React.StrictMode>
    <App />
  </React.StrictMode>
);
reportWebVitals();
```

App.js

```
import logo from './logo.svg';
import './App.css'; import
First from './First.js';
function App() {   return (
    <First />
  );
}  export default
App;
```

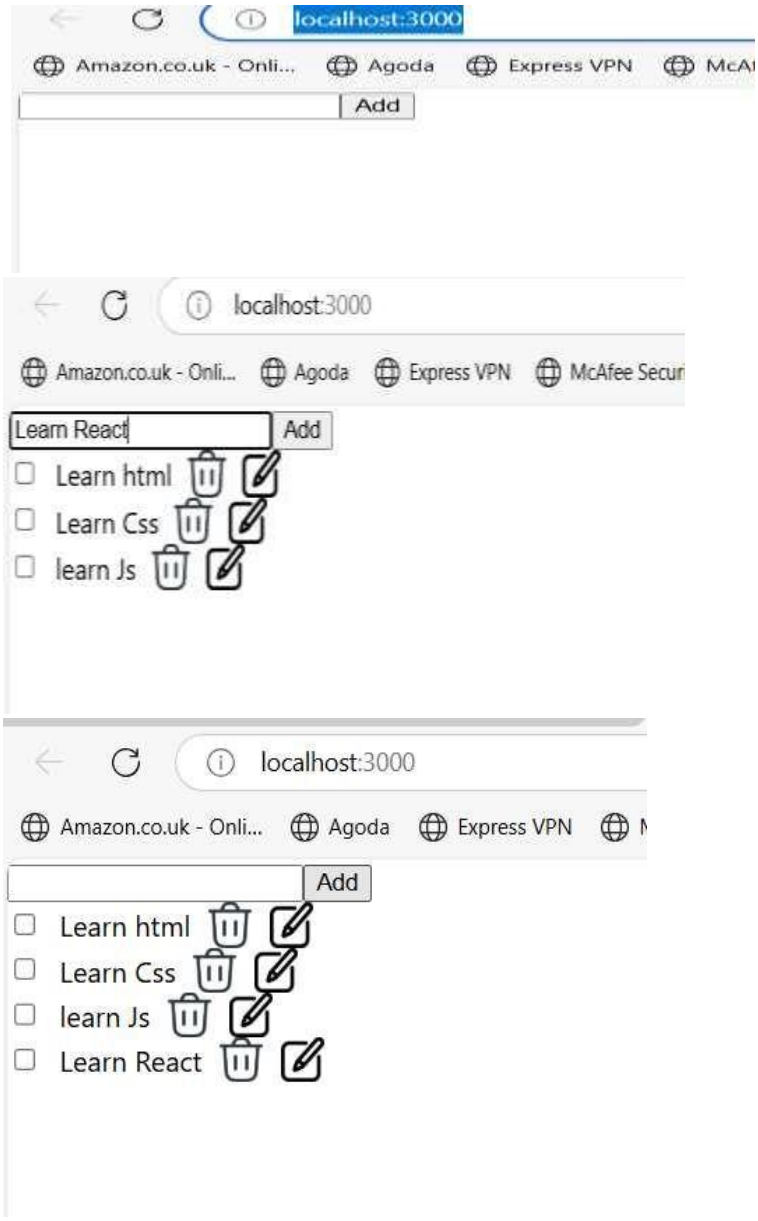
First.js

```
import React, { useState } from "react";
import delIcon from "./del.png";   import
editIcon from "./edit.png";
 function Show() {     const [val, setVal]
= useState('');     const [items, setItems]
= useState([]);     const [editIndex,
setEditIndex] = useState(null);  // Track
editing index     const [editValue,
setEditValue] = useState('');  // Track new
text input

    const change = () => {         if
(val.trim() !== '') {
setItems((prev) => [...prev, { text:
val, done: false }]);
setVal('');
        }
    };
    const updateDoneStatus = (index) => {
setItems((prev) =>
prev.map((item, i) =>                 i ===
index ? { ...item, done:
!item.done } : item
           )
```

```
        );
    };      const deleteItem = (index) => {
setItems((prev) => prev.filter((_, i) => i !==
index));
    };      const startEdit = (index) =>
{         setEditIndex(index);
setEditValue(items[index].text);
    };      const saveEdit = (index) => {
setItems((prev) =>             prev.map((item,
i) =>                 i === index ? { ...item,
text:
editValue } : item
            )         );
setEditIndex(null); // Exit edit mode
    };
return (
        <>
            <input
type="text"
value={val}
onChange={(e) =>
setVal(e.target.value)}
            />
            <button onClick={change}>Add</button>

            {items.map((item, index) => (
<div key={index} style={{ display: 'flex',
alignItems: 'center', gap:
'10px' }}>
                    <input
                        type="checkbox"
checked={item.done}                          onChange={()
=> updateDoneStatus(index)}
                    />

                    {editIndex === index ? (
                        <>
                            <input
```

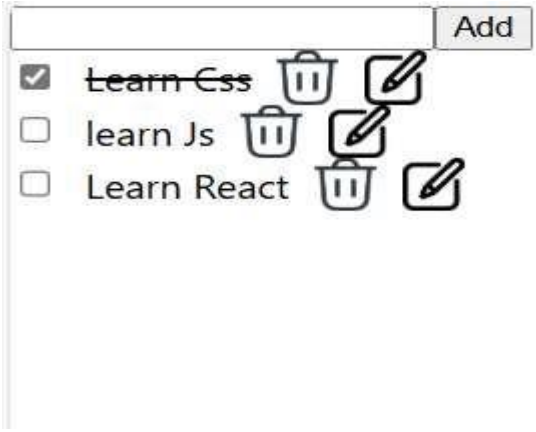
```
                                type="text"
value={editValue}
onChange={(e) => setEditValue(e.target.value)}
                            />
                            <button onClick={()
=> saveEdit(index)}>Save</button>
                        </>
                    ) : (
                        <p style={{ margin: 0 }}>
<span style={{ textDecoration: item.done ? 'line-
through' :
'none' }}>
                                {item.text}
                            </span>
                        </p>
                    )}

                    <input
type="image"
src={delIcon}
alt="Delete"
width="25" height="25"
onClick={() => deleteItem(index)}
                    />
<input                              type="image"
src={editIcon}
alt="Edit"                              width="25"
height="25"
onClick={() => startEdit(index)}
/>
                </div>
            ))}
        </>
    );
}
export default Show;
```

OUTPUT:

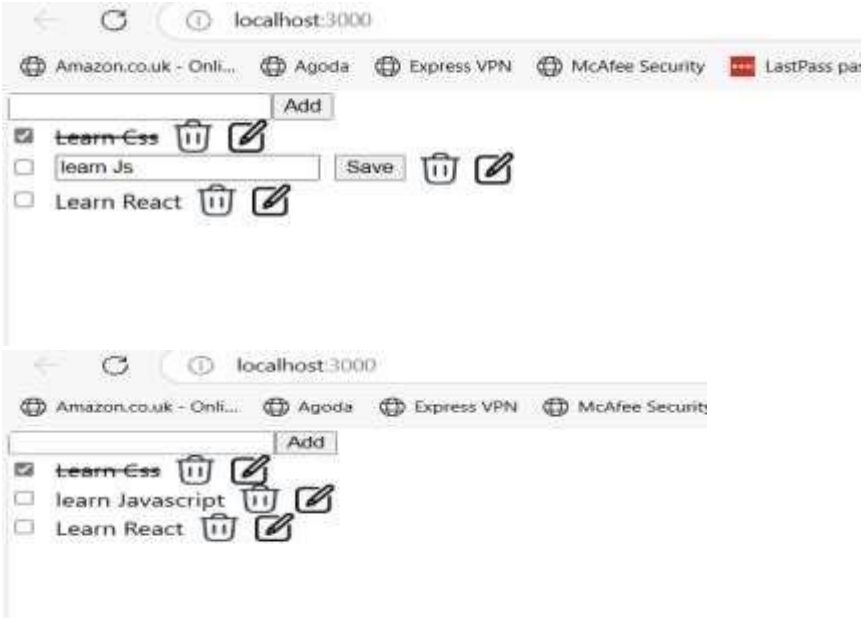


When Completed(strike it) :

When delete(remove it): Removed the Learn html



When editing:(Learn js)



LEARNING OUTCOME:

- I learned how to create an HTML form to capture the user's name.
- I learned how to implement AJAX to asynchronously send and receive data without reloading the page.
- I learned how to update the page dynamically to display a personalized welcome message.
- I learned how to integrate client-side interactions with server-side responses for a seamless user experience.