# Module 4

## Exercise 3(React Js Hooks & ContextAPI)

Q 1.) Create a component that shows different content based on whether the user is logged in or not (e.g., **"Welcome back"** or **"Please log in"**). Use a state hook to toggle the logged-in state and update the UI accordingly.

A 1.)ReactJs

```jsx
import { useContext, useState } from 'react';
import './login.css'
import { UserContext } from '../context/UserContext';


function Login({login}) {

    const { dispatchUserEvent } = useContext(UserContext);

    const [name, setName] = useState('');
    const [email, setEmail] = useState('');

    const handleAddUser = (e) => {
        e.preventDefault();

        if (!name || !email) {
            alert("Please fill all fields");
            return;
        }

        const user = {
            id: Date.now(),
            name,
            email
        };
        console.log(user.id)
        dispatchUserEvent('ADD_USER', { newUser: user });

        setName('');
        setEmail('');
        // login();
    };

    return (
        <div className="login-container">
            <h2>Login Form</h2>

            <form onSubmit={handleAddUser}>
                <label>Name</label>
                <input
```

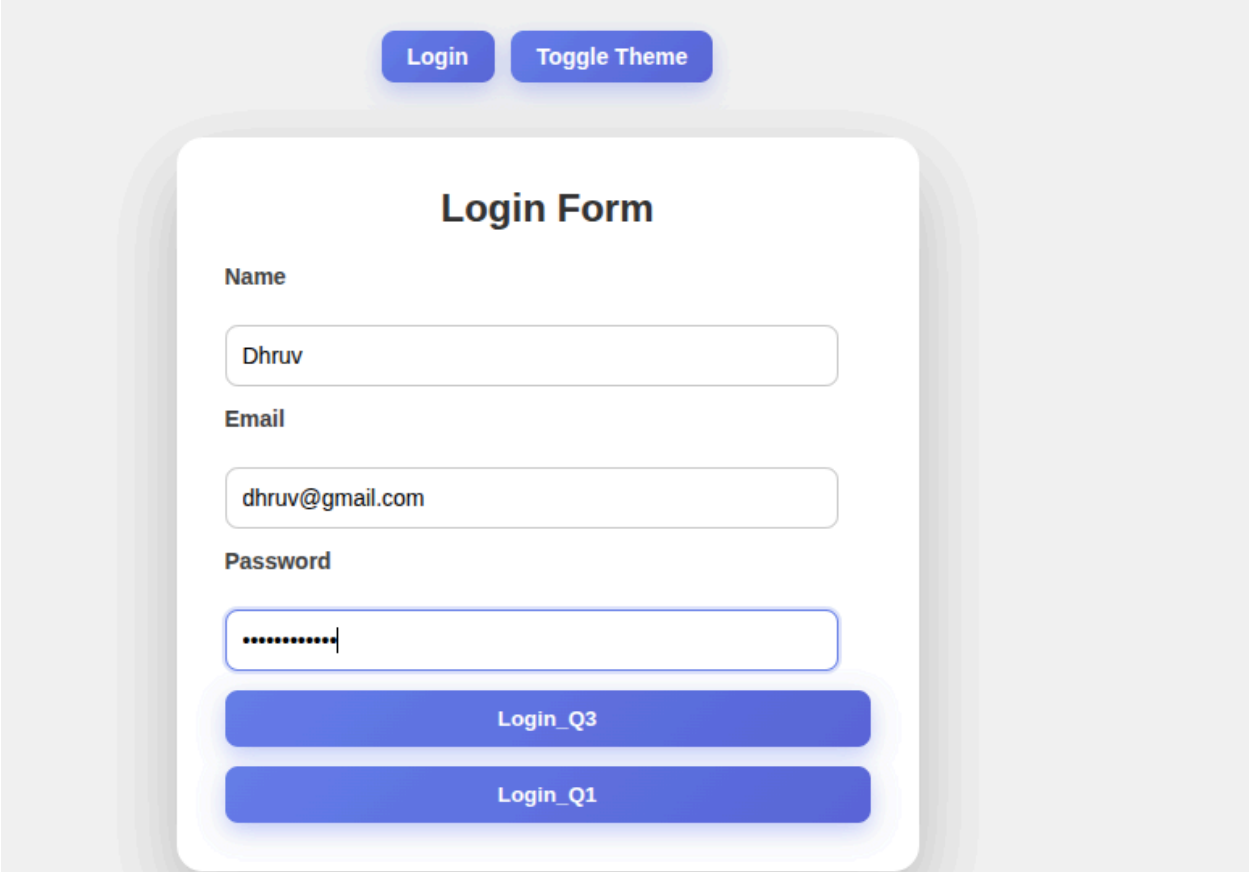
Logout

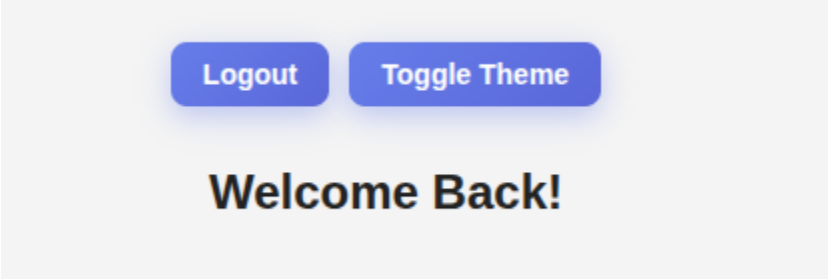
```
function Logout(){
    return(
        <div>
            <h2>Logout</h2>
            <p>You have been logout please login again!</p>
        </div>
    );
}

export default Logout
```

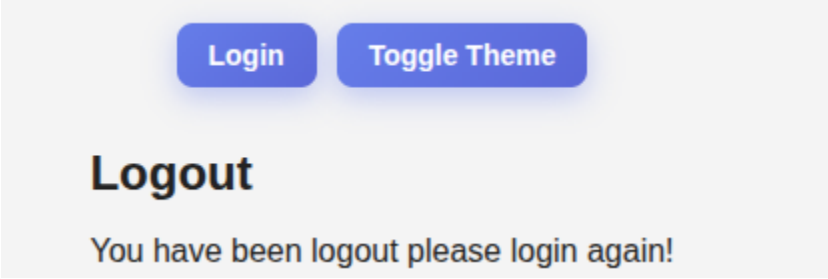
Home

```
function Home({isLogin}){
    return(
        <div>
            {isLogin?(
          <h2>Welcome Back!</h2>
          ):(
            <h2></h2>
          )
          }
        </div>
    );
}

export default Home
```

Output


Login
Toggle Theme
Login Form
Name
Dhruv
Email
dhruv@gmail.com
Password
Login_Q3
Login_Q1


Logout
Toggle Theme
Welcome Back!


Login
Toggle Theme
Logout
You have been logout please login again!

Q 2.) Create a simple **ThemeContext** that toggles between **light** and **dark** themes. Implement a **ThemeProvider** that supplies the current theme, and a **ThemeToggler** component that switches between themes. Make sure that the theme changes dynamically in child components.

A 2.) ReactJs

ThemeContext

```
import { createContext } from "react";

export const ThemeContext=createContext();
```

ThemeProvider

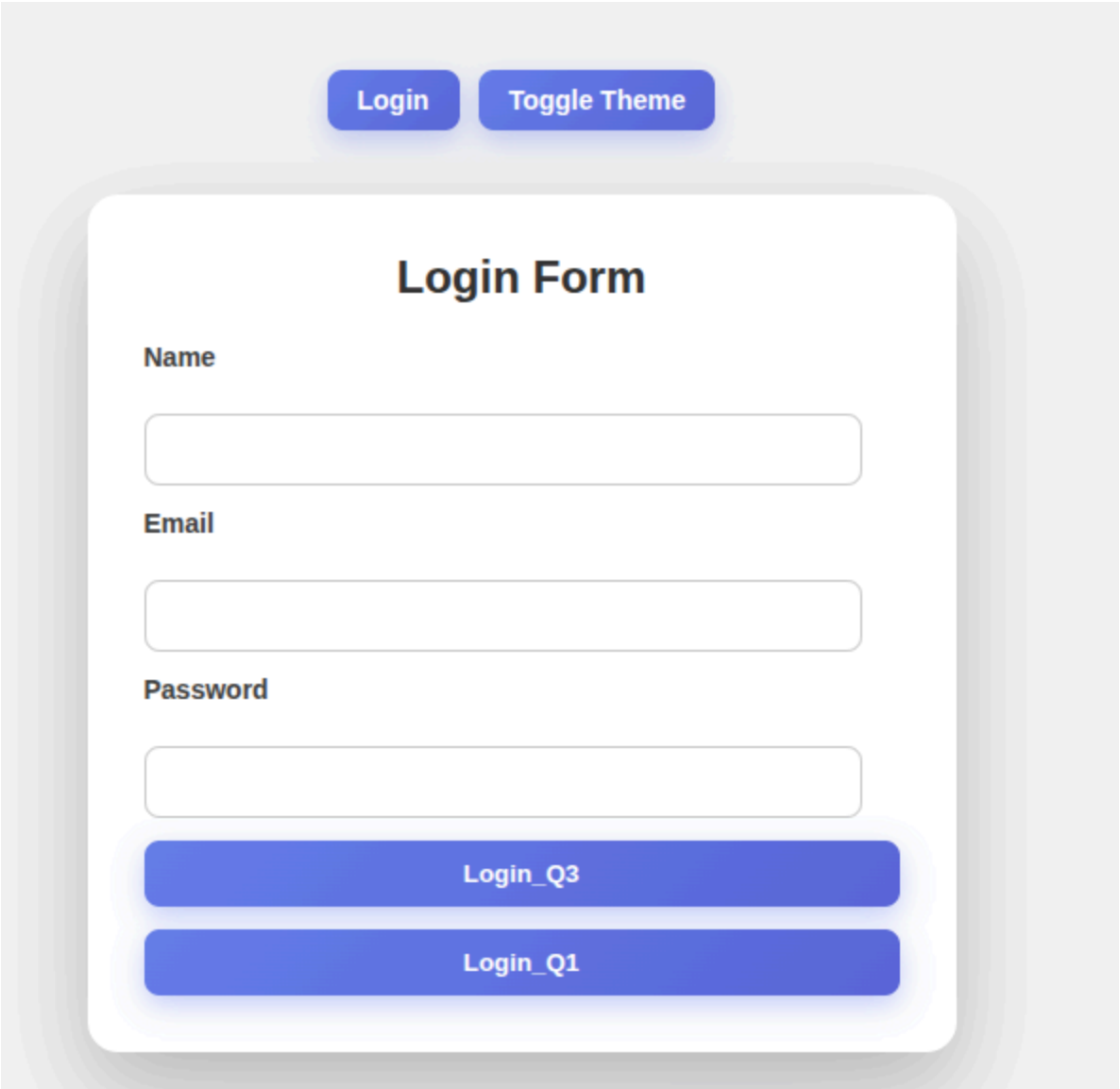
```
import { useState } from "react";
import { ThemeContext } from "../context/ThemeContext";

function ThemeProvider({children}){
    const [theme,setTheme]=useState("light");
    const themeToggler=()=>{
        setTheme(theme==='light'?'dark':'light');
    }
    return(
        <ThemeContext.Provider value={{theme,themeToggler}}>
            {children}
        </ThemeContext.Provider>
    )
}

export default ThemeProvider
```

Index.jsx

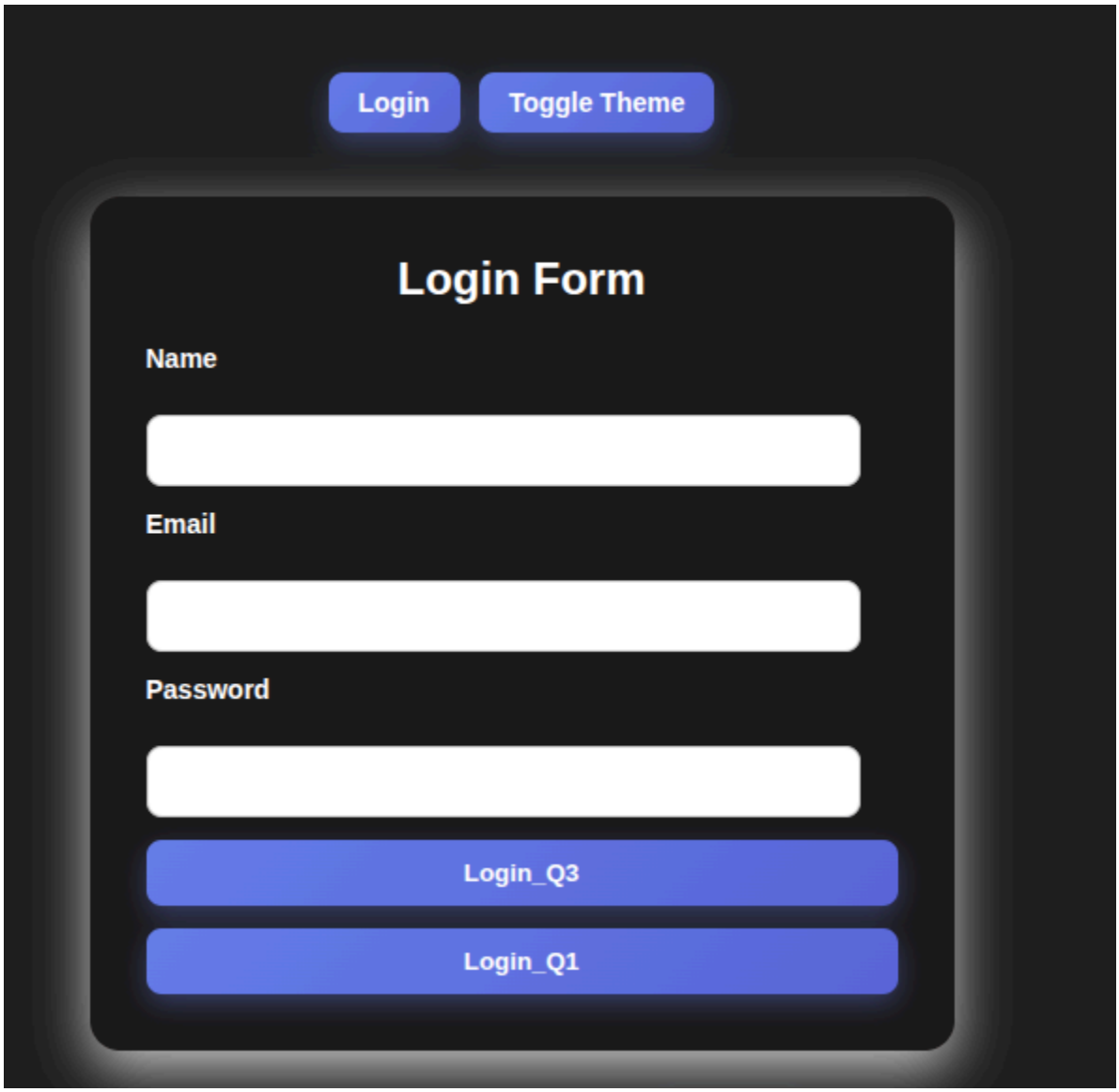
```jsx
import React from 'react';
import ReactDOM from 'react-dom/client';
import './index.css';
import App from './App';
import ThemeProvider from './provider/ThemeProvider';
import { BrowserRouter } from 'react-router-dom';
const root = ReactDOM.createRoot(document.getElementById('root'))
root.render(
  <React.StrictMode>
    <BrowserRouter>
      <ThemeProvider>
        <App />
      </ThemeProvider>
    </BrowserRouter>
  </React.StrictMode>
);
```

Output

Light


Login
Toggle Theme
Login Form
Name
Email
Password
Login_Q3
Login_Q1

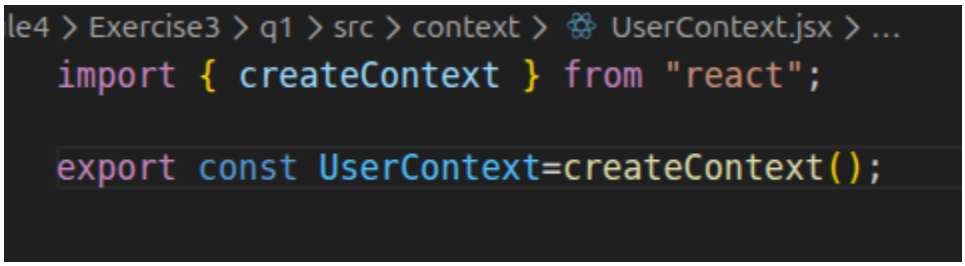
Dark


Login
Toggle Theme
Login Form
Name
Email
Password
Login_Q3
Login_Q1

Q 3.) Create a **UserContext** that holds information about the logged-in user (e.g., name and email). Create a **UserProfile** component that displays the user's information, and a **Login** component that updates the user's data via context when the user logs in.

A 3.) Reactjs

UserContext




UserProvider

4 > Exercise3 > q1 > src > provider > UserProvider.jsx > ...

```jsx
import { UserContext } from "../context/UserContext";
import { useState } from "react";

export function UserProvider({children}) {

  const [users, setUser] = useState([]);

  const dispatchUserEvent = (actionType, payload) => {
    switch (actionType) {

      case "ADD_USER":
        setUser(prevUsers => [...prevUsers, payload.newUser]);
        break;

      default:
        break;
    }
  };

  return (
    <UserContext.Provider value={{ users, dispatchUserEvent }}>
      {children}
    </UserContext.Provider>
  );
}

export default UserProvider;
```

UserProfile

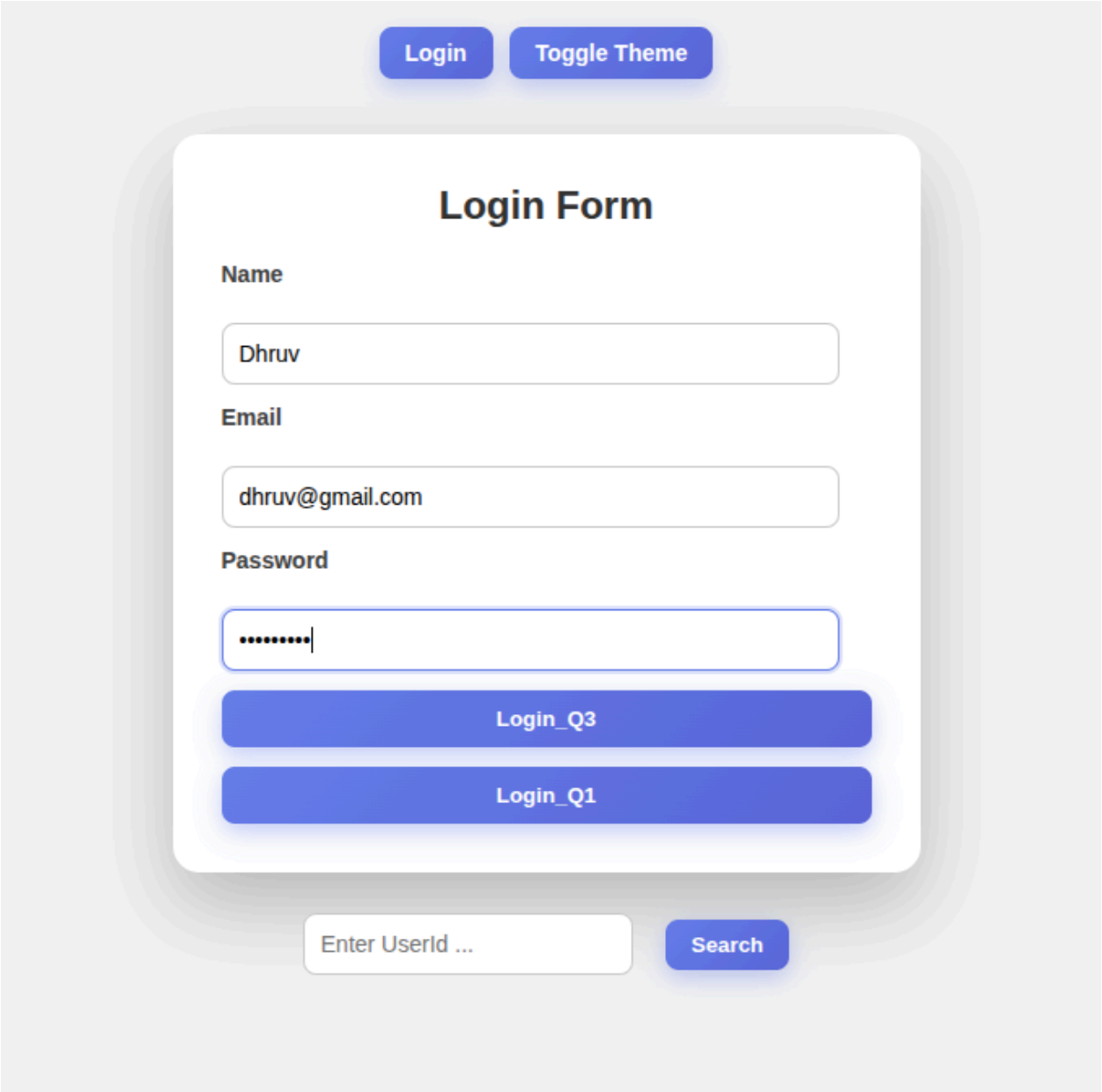
```jsx
import { useContext, useState } from "react";
import { UserContext } from "../context/UserContext";

function UserProfile() {
    const { users } = useContext(UserContext);
    const [user, setUser] = useState({});
    const [id, setId] = useState("");

    const handleSearch = () => {
        const found = users.find(u => u.id === Number(id));
        if (found) setUser(found);
        else setUser({});
    };

    return (
        <div className="container-btn">
            <input
                type="number"
                value={id}
                onChange={e => setId(e.target.value)}
                placeholder="Enter UserId ..."
            />
            <button type="button" onClick={handleSearch}>Search</
            {user.name && <h3>{user.name}</h3>}
            {user.email && <p>{user.email}</p>}
        </div>
    );
}
```

Login.jsx

```jsx
import { useContext, useState } from 'react';
import './login.css'
import { UserContext } from '../context/UserContext';


function Login({login}) {

    const { dispatchUserEvent } = useContext(UserContext);

    const [name, setName] = useState('');
    const [email, setEmail] = useState('');

    const handleAddUser = (e) => {
        e.preventDefault();

        if (!name || !email) {
            alert("Please fill all fields");
            return;
        }

        const user = {
            id: Date.now(),
            name,
            email
        };
        console.log(user.id)
        dispatchUserEvent('ADD_USER', { newUser: user });

        setName('');
        setEmail('');
        // login();
    };

    return (
        <div className="login-container">
            <h2>Login Form</h2>

            <form onSubmit={handleAddUser}>
                <label>Name</label>
                <input
```

Output

Login.jsx



UserProfile.jsx