

Module 4

Exercise 2 (ReactJs States & Props)

Q 1.) Create a counter component with a button.

A 1.) React js

```
import {useState} from 'react';
import './CounterButton.css'
function CounterButton(){

  const [count,counter] =useState(0)

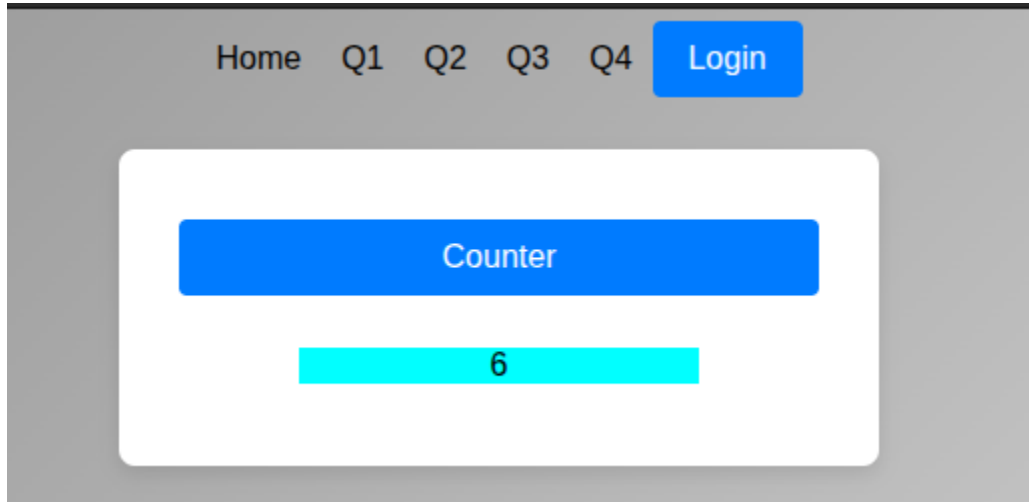
  let incrementCounter={()=>{
    counter(count+1);
  }}

  return(
    <div className="container">
      <button onClick={ incrementCounter}>
        Counter
      </button>

      <p id="display">
        {count}
      </p>
    </div>
  );
}

export default CounterButton;
```

Output

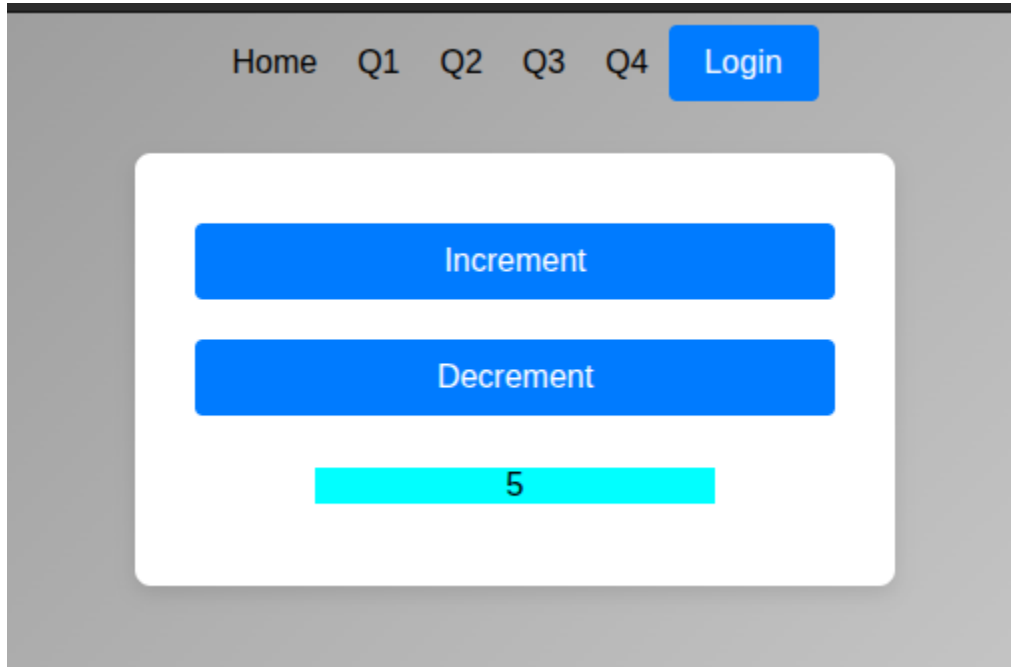


Q 2.) Create a counter with increment and decrement buttons.

A 2.) React js

```
Module4 > Exercise2 > q1 > src > components > IncrementDecrementCounter.jsx > IncrementDecrementCounter
1  import {useState} from 'react';
2  import "./CounterButton.css"
3  function IncrementDecrementCounter(){
4
5      const [count,setCount] =useState(0)
6
7      let incrementCounter=()=>{
8          setCount(count+1);
9      }
10
11     let decrementCounter=()=>{
12         setCount(count-1);
13     }
14
15     return(
16         <div className="container">
17             <button onClick={ incrementCounter}>
18                 Increment
19             </button>
20             <button id="decrement-btn" onClick={ decrementCounter} disabled={count===0}>
21                 Decrement
22             </button>
23
24             <p id="display">
25                 {count}
26             </p>
27         </div>
28     );
29 }
30
31 export default IncrementDecrementCounter;
```

Output



Q 3.) Create a parent() component that has a message in its state.

A 3.) React js

Parent()

```
import { useState } from "react";
import Child from "../Child";

function Parent(){
  const [message, setMessage]=useState("");
  let handleChange=(event)=>{
    setMessage(event.target.value);
  }

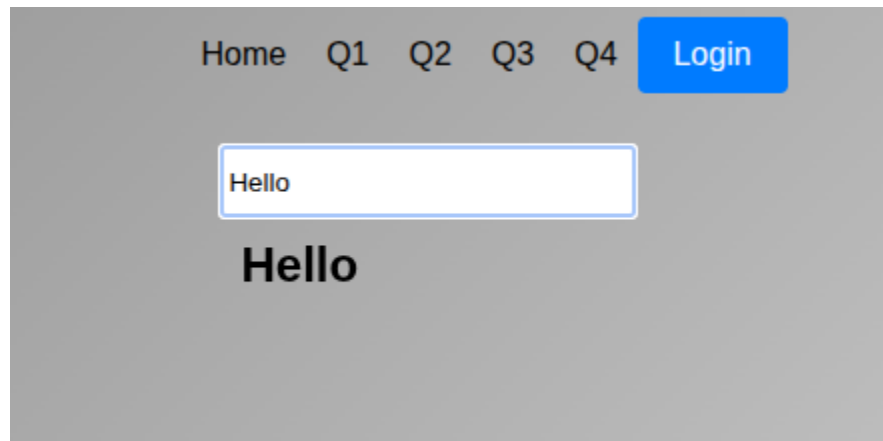
  return(
    <div className="q3-div">
      <input type="text" value={message} onChange={handleChange} placeholder="Type your message" />
      <Child message={message}/>
    </div>
  );
}

export default Parent;
```

Child()

```
function Child(props){  
  return(  
    <h2>{props.message}</h2>  
  );  
}  
  
export default Child
```

Output



Q 4.) Create a **TodoList** component that stores a list of to-dos in its state.

A 4.) React js

TodoList

```
Module4 > Exercise2 > q1 > src > components > TodoList.jsx > TodoList
1  import { useState } from "react";
2  import TodoForm from "../TodoForm";
3
4  function TodoList() {
5      const [showForm, setShowForm] = useState(false);
6      const [todo, setTodo] = useState([]);
7
8      const handleTodo = (obj) => {
9          setTodo((prevTodos) => [...prevTodos, obj]);
10     }
11
12     return (
13         <div>
14             <div className="todo">
15
16                 <div className="todo-header">
17                     <h2>My Todos</h2>
18                     <button className="add-btn" onClick={() => setShowForm(true)}>
19                         Add
20                     </button>
21                 </div>
22
23                 <ul>
24                     {todo.map((item, index) => (
25                         <li key={index}>
26                             <h3>{item.title}</h3>
27                             <p>{item.description}</p>
28                         </li>
29                     ))}
30                 </ul>
31             </div>
32
33             {showForm && (
34                 <div className="modal-overlay">
35                     <TodoForm
36                         onClose={() => setShowForm(false)}
37                         handleTodo={handleTodo}
38                     />
39                 </div>
40             )}
41         </div>
42     );
43 }
```

TodoForm

```
Module4 > Exercise2 > q1 > src > components > TodoForm.jsx > TodoForm
1  import { useState } from 'react';
2  import './TodoForm.css'
3
4  function TodoForm({ onClose, handleTodo }) {
5      const [title, setTitle] = useState("")
6      const [description, setDescription] = useState("")
7
8      const onSave = (e) => {
9          e.preventDefault();
10
11          const todo = {
12              title: title,
13              description: description
14          };
15
16          handleTodo(todo);
17          onClose();
18
19          setTitle("");
20          setDescription("");
21      }
22
23      return (
24          <div className="todo-container">
25              <h2>Add Task</h2>
26              <form onSubmit={onSave}>
27                  <label htmlFor="title">Title</label>
28                  <input
29                      type="text"
30                      name="title"
31                      id="title"
32                      value={title}
33                      onChange={(e) => setTitle(e.target.value)}
34                      required
35                  />
36
37                  <label htmlFor="description">Description</label>
38                  <textarea
39                      name="description"
40                      id="description"
```

Output

TodoForm

Add Task

Title

Explore Context API

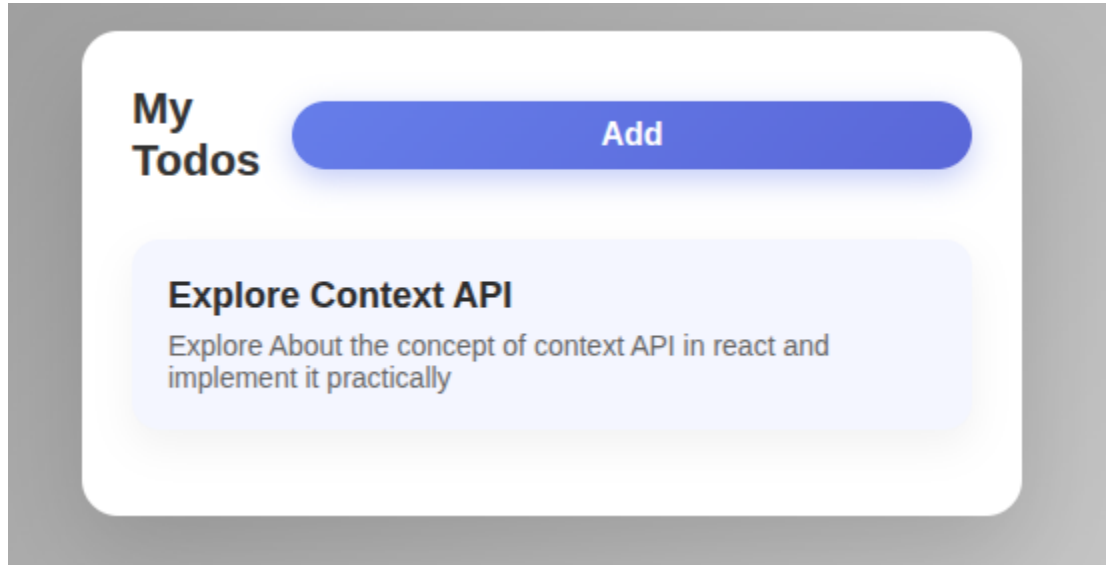
Description

Explore About the concept of context API in react and implement it practically

Save

Cancel

TodoLst



Q 5.) Create a component that toggles between two views: **"Login"** and **"Logout"**.

A 5.) React js

Login

```
import './login.css'
function Login({login}){

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

      <form>
        <label htmlFor="email">Email</label>
        <input type="email" name="email" id="email" required/>
        <label htmlFor="password">Password</label>
        <input type="password" name="password" id="password" required/>

        <button type="button" className="login-btn" onClick={login}>Login</button>
      </form>
    </div>
  );
}

export default Login;
```


Logout

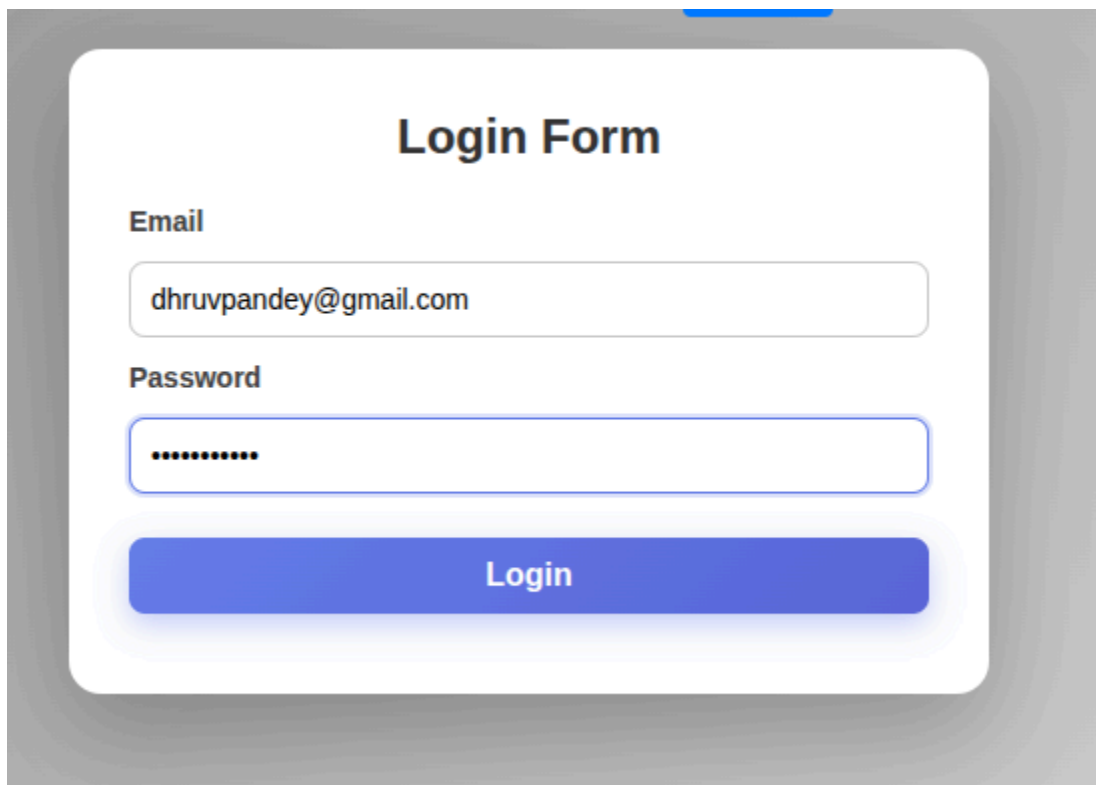
```
function Logout(){  
  return(  
    <div>  
      <h2>Logout</h2>  
      <p>You have been logout please login again!</p>  
    </div>  
  );  
}  
  
export default Logout
```

App.jsx

```
<div>  
  <nav className="home">  
    <Link to="/">Home</Link>  
    <Link to="/q1">Q1</Link>  
    <Link to="/q2">Q2</Link>  
    <Link to="/q3">Q3</Link>  
    <Link to="/q4">Q4</Link>  
    {isLoggedIn ? (  
      <button type="button" onClick={navigateLogout} className="btn">Logout</button>  
    ) : (<button type="button" onClick={navigateLogin} className="btn">Login</button>)}  
  </nav>  
</div>
```

Output

Login



A login form with a white background and rounded corners, centered on a gray background. The form has a title "Login Form" in bold black text. Below the title are two input fields: "Email" with the value "dhruvpandey@gmail.com" and "Password" with masked characters ".....". A blue "Login" button is at the bottom of the form.

Login Form

Email

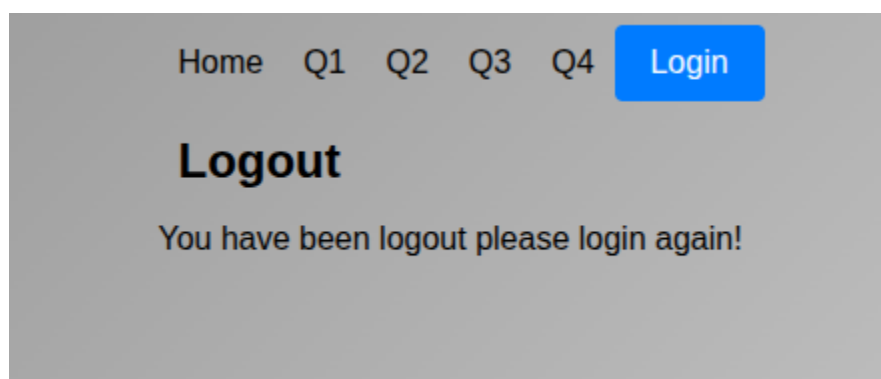
dhruvpandey@gmail.com

Password

.....

Login

Logout



A gray rectangular area containing a navigation bar with links "Home", "Q1", "Q2", "Q3", "Q4", and a blue "Login" button. Below the navigation bar, the word "Logout" is displayed in bold black text, followed by the message "You have been logout please login again!" in a smaller black font.

Home Q1 Q2 Q3 Q4 Login

Logout

You have been logout please login again!