Chapter 4: State and useState Hook

1 Introduction

Is chapter mein humne **state** aur **useState** hook ke baare mein seekha. State component ka data hota hai jo change ho sakta hai, aur jab yeh change hota hai, React UI ko re-render karta hai. `useState` hook functional components mein state manage karta hai.

2 4.1 What is State?

State component ka woh data hota hai jo time ke saath change ho sakta hai, jaise button click pe count badhna ya form input update hona. State change hone pe React component ko re-render karta hai.

2.1 Real-World Example

Ek food delivery app mein "Add to Cart" button click pe order count badhta hai. Yeh count state mein store hota hai.

3 4.2 useState Hook: The Basics

useState ek React hook hai jo state banane aur update karne ka tareeka deta hai. Yeh do cheezein deta hai:

- **State variable**: Data store karta hai.
- **Setter function**: State ko update karta hai.

3.1 Syntax

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

function Component() {
  const [stateVariable, setStateVariable] =
     useState(initialValue);
}
```

3.2 Code Example: Counter App

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>Counter App</title>
  <script
     src="https://cdn.jsdelivr.net/npm/react@18.2.0/umd/react.development.js
  <script
     src="https://cdn.jsdelivr.net/npm/react-dom@18.2.0/umd/react-dom.develo
  <script
     src="https://cdn.jsdelivr.net/npm/@babel/standalone@7.20.6/babel.min.js
```

```
  <script src="https://cdn.tailwindcss.com"></script>
</head>
<body>
  <div id="root"></div>
  <script type="text/babel">
    function CounterApp() {
      const [count, setCount] = React.useState(0);

      const handleIncrement = () => {
        setCount(count + 1);
      };

      const handleDecrement = () => {
        setCount(count - 1);
      };

      return (
        <div className="text-center p-4">
          <h1 className="text-3xl font-bold
             text-blue-600">Counter App</h1>
          <p className="text-xl">Count: {count}</p>
          <button
            className="bg-green-500 text-white px-4 py-2 m-2
               rounded"
            onClick={handleIncrement}
          >
            Add
          </button>
          <button
            className="bg-red-500 text-white px-4 py-2 m-2
               rounded"
            onClick={handleDecrement}
          >
            Subtract
          </button>
        </div>
      );
    }

    const root =
       ReactDOM.createRoot(document.getElementById('root'));
    root.render(<CounterApp />);
  </script>
</body>
</html>
```

3.3 Explanation

- `useState(0)`: `count` ko 0 se initialize kiya.
- `setCount`: `count` ko update karta hai.

- `handleIncrement`: Count ko 1 se badhata hai.
- `handleDecrement`: Count ko 1 se ghatata hai.
- `onClick`: Button click ko handle karta hai.

3.4 Output

- Heading: **Counter App**
- Text: **Count: 0** (initially)
- Buttons: **Add** aur **Subtract**
- Add click pe: 1, 2, 3...; Subtract pe: 0, -1, -2...

3.5 Real-World Example

Restaurant app mein "Add to Cart" button se cart count badhta hai, jo state ka use hai.

4 4.3 Updating State Correctly

State ko direct modify nahi karna chahiye; hamesha `setState` use karo.

4.1 Wrong Way

```
function CounterApp() {
  const [count, setCount] = React.useState(0);

  const handleIncrement = () => {
    count = count + 1; // Wrong: Direct modification
  };

  return <button onClick={handleIncrement}>Add</button>;
}
```

4.2 Correct Way

```
function CounterApp() {
  const [count, setCount] = React.useState(0);

  const handleIncrement = () => {
    setCount(count + 1); // Correct: Use setCount
  };

  return <button onClick={handleIncrement}>Add</button>;
}
```

5 4.4 State with Objects

State ek object bhi ho sakta hai, jaise user ka data.

5.1 Code Example: Object State

```
function UserProfile() {
  const [user, setUser] = React.useState({ name: "Rahul", age:
     25 });

  const updateName = () => {
    setUser({ ...user, name: "Priya" });
  };

  return (
    <div className="text-center p-4">
      <h1 className="text-2xl font-bold">User Profile</h1>
      <p>Name: {user.name}</p>
      <p>Age: {user.age}</p>
      <button
        className="bg-blue-500 text-white px-4 py-2 rounded"
        onClick={updateName}
      >
        Change Name
      </button>
    </div>
  );
}
```

5.2 Explanation

- `useState({ name: "Rahul", age: 25 })`: Object state initialize kiya.
- `{...user}`: Spread operator se purana state copy kiya.
- `setUser`: Naya object pass kiya.

5.3 Output

- Initially: **Name: Rahul, Age: 25**
- Button click pe: **Name: Priya, Age: 25**

6 Common Mistakes

- **Direct State Modification**: `count = count + 1` mat karo.
- **Forgetting Spread Operator**: Object update mein `{...state}` na karne se data lost ho sakta hai.
- **Multiple State Updates**: Ek function mein multiple `setState` carefully handle karo.

7 Interview Tips

- **State kya hai?**

  - Component ka data jo change ho sakta hai, UI re-render trigger karta hai.
- **useState ka use kyun?**
  - Functional components mein state manage karta hai.
- **State vs Props?**
  - State component ke andar manage hota hai, change ho sakta hai.
  - Props parent se pass hote hain, read-only hote hain.

8 Assignment: Practice Time

8.1 Task 1: Simple Counter

- `Counter` component banao jo `useState` se number track kare.
- Buttons: "Increase" (+1) aur "Decrease" (-1).
- Count display karo, Tailwind se style karo.

8.2 Task 2: Toggle Button

- `ToggleButton` component banao jo boolean state track kare.
- Button click pe text "ON" se "OFF" ya vice-versa change kare.

8.3 Task 3: Object State

- `CartItem` component banao jo object state `{ item: "Chai", quantity: 1 }` rakhe.
- Button se quantity badhao.
- Item name aur quantity display karo, Tailwind se style karo.