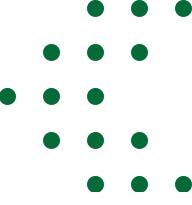




**Eethal Nad**  
IT Solutions

# MERN STACK DEVELOPMENT

**DAY-25**



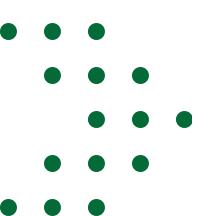
# STATE MANAGEMENT IN REACT

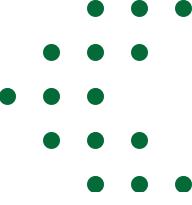
## Title

State Management – useState Hook

## Description

State management is how React components store, update, and use data that changes over time (like input values, counters, toggles).





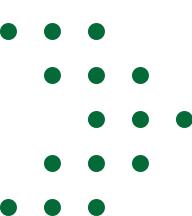
# STATE MANAGEMENT IN REACT

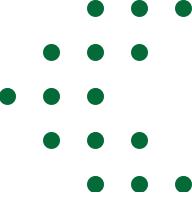
## What is State?

- State is a built-in object in React
- It holds dynamic data
- When state changes → UI re-renders automatically

## Examples of State

- Form input values
- Counter number





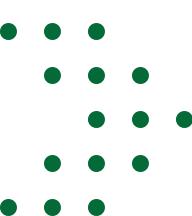
# STATE MANAGEMENT IN REACT

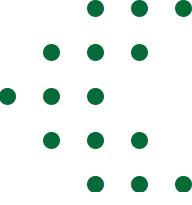
## Why State is Important?

- UI needs to respond to user actions
- Keeps data inside the component
- Avoids manual DOM manipulation
- Makes apps dynamic & interactive

**Without state → static UI**

**With state → dynamic UI**





# STATE MANAGEMENT IN REACT

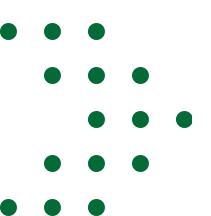
## What is useState?

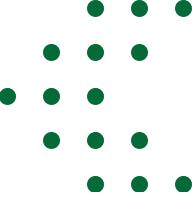
- useState is a React Hook used to create state in functional components

## Syntax

```
const [state, setState] = useState(initialValue);
```

- state → current value
- setState → function to update value
- initialValue → default state





# STATE MANAGEMENT IN REACT

## Updating State Using Previous Value :

React state updates are asynchronous, so the current state value may not update immediately.

### Syntax

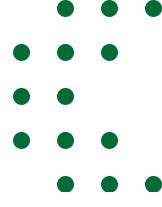
```
setCount(prevCount => prevCount + 1); // inside the function
```

```
<button onClick={() => setCount(prev => prev + 1)}>
```

Increment

```
...</button>
```

```
...
```



# MULTIPLE STATES IN ONE COMPONENT

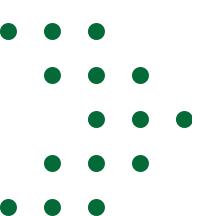
You can use useState multiple times :

```
const [name, setName] = useState("");
```

```
const [age, setAge] = useState(0);
```

```
const [isActive, setIsActive] = useState(true);
```

Each state is independent



# MULTIPLE STATES IN ONE COMPONENT

## Example Code:

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

function UserForm() {

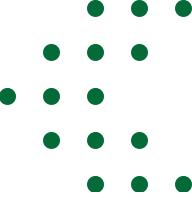
  const [name, setName] = useState("");
  const [email, setEmail] = useState("");
```

# MULTIPLE STATES IN ONE COMPONENT

```
return (  
  <div>  
    <input  
      placeholder="Name"  
      value={name}  
      onChange={(e) => setName(e.target.value)}  
    />
```

# MULTIPLE STATES IN ONE COMPONENT

```
<input  
    placeholder="Email"  value={email}  
    onChange={(e) => setEmail(e.target.value)} />  
  
<p>Name: {name}</p>  
  
<p>Email: {email}</p>  
  
</div>  
  
);  
}
```



# STATE WITH BOOLEAN (TOGGLE)

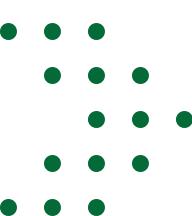
```
const [isLoggedIn, setIsLoggedIn] = useState(false);
```

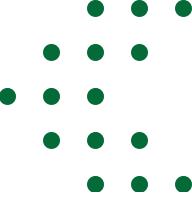
```
<button onClick={() => setIsLoggedIn(!isLoggedIn)}>
```

Toggle

```
</button>
```

```
{isLoggedIn && <h2>Welcome User!</h2>}
```





# STATE WITH BOOLEAN (TOGGLE)

**When to Use State:**

**Use state when:**

- Data changes over time
- UI depends on user actions
- You need dynamic rendering

