🧠 Passing Arguments to Event Handlers in React

In React, sometimes you need to pass **custom arguments** (like `id`, `name`, `index`, etc.) to an event handler function — especially when mapping over lists or working with dynamic data.

✅ The React Way: Use an Arrow Function

👉 Example: Passing one argument

jsx Copy Edit

```
function greetUser(name) { alert(`Hello, ${name}!`); } <button onClick={() => greetUser('Vivek')}>Greet</button>
```

- You wrap the function in an **arrow function** so it doesn't run immediately.
- It only executes **when clicked**.

📦 Full Working Example

jsx Copy Edit

```
function UserList() { const users = ['Vivek', 'Priya', 'Amit']; const handleGreet = (user) => { alert(`Hello, ${user}`); }; return ( <div> {users.map((user, index) => ( <button key={index} onClick={() => handleGreet(user)}> Greet {user} </button> ))} </div> ); }
```

⚠️ Don't do this:

jsx Copy Edit

```
// This calls the function immediately (bad) <button onClick={handleGreet(user)}>Greet</button>
```

🧪 Example: Pass Event + Argument

jsx

Copy Edit

```jsx
function handleClick(name, event) { console.log('Name:', name); console.log('Event:', event); }
<button onClick={(e) => handleClick('Vivek', e)}>Click</button>
```

✅ Alternative (Class Components): `.bind()`

jsx

Copy Edit

```jsx
<button onClick={this.handleClick.bind(this, 'Vivek')}>Click</button>
```

- Not commonly used in function components anymore.