🧩 What is Destructuring?

Destructuring lets you **unpack values** from arrays or **pull properties** from objects into individual variables — in a **clean and short** way.

🔹 1. Destructuring Objects

✅ Basic Syntax:

javascript Copy Edit

```
const person = { name: "Alice", age: 25, city: "New York" }; const { name, age } = person;
console.log(name); // "Alice" console.log(age); // 25
```

✅ Rename variables:

javascript Copy Edit

```
const { name: userName } = person; console.log(userName); // "Alice"
```

✅ Set default values:

javascript Copy Edit

```
const { job = "Unemployed" } = person; console.log(job); // "Unemployed"
```

🔹 2. Destructuring Arrays

✅ Basic Syntax:

javascript Copy Edit

```
const numbers = [10, 20, 30]; const [a, b] = numbers; console.log(a); // 10 console.log(b); // 20
```

✅ Skipping Items:

javascript Copy Edit

```javascript
const [first, , third] = numbers; console.log(third); // 30
```

✅ With default values:

javascript Copy Edit

```javascript
const [x, y, z = 100] = [1, 2]; console.log(z); // 100
```

🔹 3. Destructuring in Functions

✅ Function Parameters (Objects)

javascript Copy Edit

```javascript
function greet({ name, city }) { console.log(`Hello ${name} from ${city}`); } greet(person); // Hello Alice from New York
```

✅ Function Parameters (Arrays)

javascript Copy Edit

```javascript
function logCoords([x, y]) { console.log(`X: ${x}, Y: ${y}`); } logCoords([5, 10]); // X: 5, Y: 10
```

🔹 4. Destructuring in React

✅ Props Destructuring

jsx Copy Edit

```jsx
function Profile({ name, age }) { return <h2>{name} is {age} years old.</h2>; }
```

✅ State Destructuring

jsx

Copy Edit

```
const [count, setCount] = useState(0);
```

This is **array destructuring** — super common in React hooks!

🧠 Quick Recap

Destructuring Type	Example	Use Case
Object	`{ name } = person`	Extract specific keys
Array	`[a, b] = arr`	Get values by position
Function Params	`({ name }) => {}`	Clean function arguments
React State	`[value, setValue] = useState()`	React Hooks