Props

- Props stands for properties.
- They are inputs to a React component.
- Props allow you to pass data from a parent component to a child component.
- They are read-only (immutable inside the child component).

Why Do We Need Props?

- To make components reusable and dynamic.
- Instead of hardcoding values, we pass data via props.

Basic Understanding

```
function Profile(props) {
  return (
    <div>
      <h2>{props.username}</h2>
      <p>Age: {props.age}</p>
    </div>
  );
}

function App() {
  return (
    <Profile username="Manas" age={21} />
  );
}
```

Destructuring Props (Cleaner Way)

```
function Profile({ username, age }) {
  return (
    <div>
      <h2>{username}</h2>
      <p>Age: {age}</p>
    </div>
  );
}
```

Props Are Immutable

- You cannot modify props inside a child component.

```
function Profile(props) {
  props.username = "Changed"; // ❌ Not allowed
  return <h2>{props.username}</h2>;
}
```

All Possibilities of Props in React

1. Basic Props (Strings, Numbers, Booleans)

```jsx
function Greeting({ name, age, isStudent }) {
  return (
    <h2>
      {name} is {age} years old. Student: {isStudent ? "Yes" : "No"}
    </h2>
  );
}

function App() {
  return <Greeting name="Manas" age={21} isStudent={true} />;
}
```

2. Props as Objects

```jsx
function Profile({ user }) {
  return (
    <div>
      <h2>{user.name}</h2>
      <p>Age: {user.age}</p>
    </div>
  );
}

function App() {
  const userObj = { name: "Manas", age: 21 };
  return <Profile user={userObj} />;
}
```

3. Props as Arrays

```
function List({ items }) {
  return (
    <ul>
      {items.map((item, i) => (
        <li key={i}>{item}</li>
      ))}
    </ul>
  );
}

function App() {
  return <List items={["Apple", "Banana", "Mango"]} />;
}
```

4. Props as Functions (Callback Functions)

- Used for event handling or passing logic.

```
function Button({ onClick }) {
  return <button onClick={onClick}>Click Me</button>;
}

function App() {
  const handleClick = () => alert("Button Clicked!");
  return <Button onClick={handleClick} />;
}
```

5. Props as JSX Elements

- Passing React elements.

```
function Card({ content }) {
  return <div className="card">{content}</div>;
}

function App() {
  return <Card content={<h2>Hello World</h2>} />;
}
```

6. Props as Children (Special Prop)

- Anything between component tags becomes props.children.

```
function Layout({ children }) {
  return (
    <div className="layout">
      <header>Header</header>
      <main>{children}</main>
      <footer>Footer</footer>
    </div>
  );
}

function App() {
  return (
    <Layout>
      <h1>Welcome to my site</h1>
      <p>This is content inside children.</p>
    </Layout>
  );
}
```

Default Props

- You can define default values if a prop is not passed.

```
function Button({ label }) {
  return <button>{label}</button>;
}

Button.defaultProps = {
  label: "Click Me"
};
```

9. Spread Operator for Props

- Pass all props at once.

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

function App() {
  const user = { name: "Manas", age: 21 };
  return <Profile {...user} />;
}
```

Thank You