CHAPTER-6
PROPS

3.1 OVEVIEW FOR PROPS:

Props are arguments passed into React components. Props are passed to components via HTML attributes. The main difference between state and props is that **props** are immutable. This is why the container component should define the state that can be updated and changed, while the child components should only pass data from the state using props.

3.2 USING PROPS IN CLASS COMPONENT:

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

class ClassProps extends Component {
  render() {
   return (
     <div>
       <p>Props Class component</p>
       <p>{this.props.value}</p>
     </div>
   );
  }
}

export default ClassProps;
```

3.3 USING PROPS IN FUNCTIONAL COMPONENT:

```
import React from "react";

function FunctionProps(props) {
  return (
    <div>
      <p>Props Function component</p>
      <p>{props.value}</p>
    </div>
  );
}

export default FunctionProps;
```

3.4 PARENT COMPONENT

The parent component means rendering child component and pass the value using props like

```
<ComponentName value={“value”} />
```

```
import React, { useState } from "react";
import ClassProps from "../Class/ClassProps";
import FunctionProps from "../Function/FunctionProps";

function Input() {
const [state, setState] = useState("");
const [render, setRender] = useState("");
return (
<React.Fragment>
<input
type="text"
value={state}
onChange={e => {
setState(e.target.value);
}}
/>
<button style={{margin:'0px 5px'}} onClick={() =>
setRender(state)}>Sent props</button>
<button style={{margin:'0px 5px'}} onClick={() =>
{setRender("");setState("")}}>Reset</button>
{render !== ""? (
<div>
<ClassProps value={render} />
<hr />
<FunctionProps value={render} />
</div>
):null}
</React.Fragment>
);
}

export default Input;
```

VS CODE VIEW AND FILE STRUCTURE: