ASSIGNMENT(29012026)

Q. No.	Question	Expected Tim to complete
1	**Scenario:** An **online shopping website** displays products dynamically. Each product is shown as a separate component, and product data is passed from a parent component. **Questions:** 1. How would you design a **parent–child component structure** in React? 2. What is the difference between **props and state**? 3. How do you pass data from parent to child components using props? 4. How does state help in updating the UI dynamically? 5. What happens when a component's state changes?	

SOLUTIONS:

Scenario

An online shopping website displays products dynamically. Each product is shown as a separate component, and product data is passed from a parent component.

---

1. Designing a parent–child component structure in React

In React, the **parent component** holds the main data (products list) and passes individual product data to **child components**.

Example:

```
function ProductList() {
  const products = [
    { id: 1, name: "Laptop", price: 50000 },
    { id: 2, name: "Phone", price: 20000 }
  ];
```

```
  return (
    <div>
      {products.map(product => (
        <ProductItem key={product.id} product={product} />
      ))}
    </div>
  );
}

function ProductItem({ product }) {
  return (
    <div>
      <h3>{product.name}</h3>
      <p>Price: ₹{product.price}</p>
    </div>
  );
}
```

📌 `ProductList` → Parent
📌 `ProductItem` → Child

---

2. Difference between props and state

Props	State
Passed from parent to child	Managed within a component
Read-only	Can be changed
Used for data sharing	Used for dynamic UI updates

---

3. Passing data from parent to child using props

Data is passed as **attributes** in the child component.

```
<ProductItem product={product} />
```

Child receives data using props:

```
function ProductItem({ product }) {
  return <h3>{product.name}</h3>;
}
```

✔️ Props allow components to be **reusable and modular**

---

4. How state helps in updating the UI dynamically

State stores data that can change over time (e.g., cart count, product availability).

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

When state updates using `setState`, React automatically updates the UI.

📌 Example:

```
<button onClick={() => setCount(count + 1)}>Add to Cart</button>
```

---

5. What happens when a component's state changes?

When state changes:

1. React re-renders the component
2. Virtual DOM compares changes
3. Only the required parts of the UI are updated

✔️ This makes React fast and efficient

Conclusion

- Parent components manage data
- Child components display data using props
- State enables dynamic UI updates
- React re-renders components efficiently when state changes