Assignment Week – 4

Parent – Child Components, Props and State in React

Name: U. Rohith

HT No: 2303A52198

Batch: 35

Introduction

Modern web applications require dynamic and responsive user interfaces. React is a popular JavaScript library that follows a component-based architecture, enabling developers to build reusable and maintainable UI elements. In an online shopping website, products are often displayed dynamically, where each product is represented as a separate component and data is passed from a parent component.

This assignment explains how parent–child component structure is designed in React and how props and state are used to manage and update data dynamically.

---

1. Parent–Child Component Structure in React

In React, applications are built using components. A **parent component** manages data and application logic, while **child components** are responsible for displaying that data.

For an online shopping website:

- The **parent component** (e.g., ProductList) stores product details such as name and price.
- Each **child component** (e.g., ProductCard) displays an individual product.

Example Structure:

```
App
 └── ProductList (Parent)
       ├── ProductCard (Child)
       ├── ProductCard (Child)
       └── ProductCard (Child)
```

This structure promotes reusability and ensures a clear separation of concerns.

2. Difference Between Props and State

Props and state are two important concepts in React used to handle data.

Props	State
Passed from parent to child	Managed within the component
Read-only	Can be updated
Used to transfer data	Used to manage dynamic data
Cannot be modified by child	Modified using state updater functions

Props are used to pass data between components, whereas **state** is used to manage data that changes over time within a component.

3. Passing Data from Parent to Child Using Props

Data is passed from a parent component to a child component using **props**. The parent defines attributes while rendering the child component, and the child receives them as props.

Conceptual Example:

```
<ProductCard name="Laptop" price={55000} />
```

Inside the child component:

```
props.name
props.price
```

This allows the child component to display data provided by the parent without modifying it.

4. Role of State in Dynamic UI Updates

State allows a component to store and manage data that changes during user interaction. In an online shopping website, examples include:

- Cart count
- Quantity of products
- User actions such as button clicks

When state is updated, React automatically updates the user interface to reflect the new data without reloading the page.

Thus, state plays a crucial role in making the UI dynamic and interactive.

5. What Happens When a Component's State Changes

When a component's state changes:

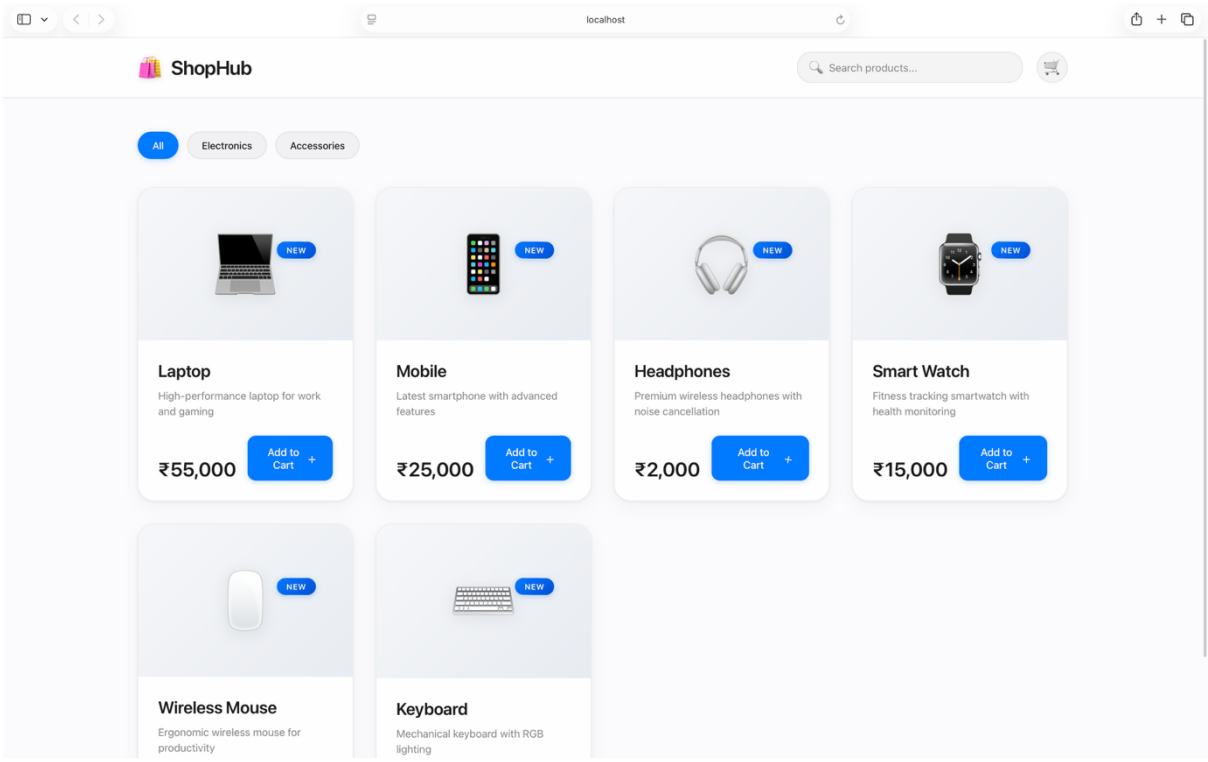
1. React detects the state change.
2. The component is re-rendered.
3. The updated data is reflected in the UI.
4. Only the affected components are updated, improving performance.

This process is known as **re-rendering** and ensures efficient UI updates.

Output

When implemented practically:

- Products are displayed dynamically.
- Each product appears as a separate component.
- UI updates automatically when user interaction occurs (e.g., cart count increases).



Conclusion

React’s component-based architecture simplifies UI development by dividing applications into reusable components. Props enable data flow from parent to child components, while state manages dynamic data within components. Together, they allow React applications to update the user interface efficiently and dynamically, making them suitable for modern web applications such as online shopping platforms.
