Full Stack Development

Scenario Question and Answers

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.

Q1. How would you design a parent–child component structure in React?

Sol : In a React-based online shopping website, the parent–child component structure is designed to ensure modularity and reusability. The parent component (for example, *ProductList*) stores and manages the product data such as names, prices, and descriptions. The child component (for example, *ProductCard*) is responsible for displaying individual product details.

The parent component passes product data to child components using props. Each product is rendered as a separate child component. This structure improves code organization, reusability, and maintainability.

Q2. What is the difference between props and state?

Sol : Props and state are both used to manage data in React, but they serve different purposes.

Props (properties) are used to pass data from a parent component to a child component. Props are read-only and cannot be modified by the child component.

State, on the other hand, is managed within the component itself. It is mutable and can be updated using functions like useState. State is used for dynamic data that can change over time, such as user input or cart quantity.

In simple terms, props are external data passed to a component, while state is internal data managed by the component.

Q3. How do you pass data from parent to child components using props?

Sol : Data is passed from a parent to a child component by adding attributes to the child component when it is called.

For example, the parent component sends product information as props, and the child component receives and displays it. This creates a one-way data flow from parent to child, which makes the application predictable and easier to debug.

Q4. ***How does state help in updating the UI dynamically?***

SOl : State allows a component to store dynamic values that may change over time, such as product quantity or cart count. When the state value changes, React automatically re-renders the component and updates the UI.

For example, when a user clicks "Add to Cart," the state variable storing the quantity increases, and the updated value is shown on the screen without refreshing the page.

Thus, state helps create interactive and dynamic user interfaces.

Q5. ***What happens when a component's state changes?***

Sol : When a component's state changes, React triggers a re-render of that component. React then updates the Virtual DOM and compares it with the previous version. Only the parts of the UI that have changed are updated in the real DOM.

This process ensures efficient updates and better performance, as React does not reload the entire page.

Conclusion : In React applications, parent components manage and pass data using props, while state manages dynamic data within components. When state changes, React efficiently updates the UI, making applications fast and interactive.