**Hands-on: 12. ReactJS-HOL**

**Introduction**

In React, conditional rendering allows components to render different content based on specific conditions. It makes the UI dynamic and interactive by changing what is displayed without needing to reload the page. React also offers techniques to control rendering behavior, such as using element variables and preventing components from rendering entirely. Let’s explore these important concepts in detail.

1. **Conditional Rendering in React**

Conditional rendering in React is the process of displaying components or elements based on certain conditions.

Common Methods:

* Using if statements:

if (isLoggedIn) {

return <Dashboard />;

} else {

return <Login />;

}

* Using ternary operator:

return (

<div>

{isLoggedIn ? <Dashboard /> : <Login />}

</div>

);

* Using logical && operator:

{hasPermission && <SecretContent />}

These methods allow you to dynamically update the UI based on user interaction, data loading, authentication status, etc.

1. **Element Variables**

In React, element variables are used to store JSX elements in a variable and conditionally assign them later in the render method.

* Example:

let message;

if (isMorning) {

message = <h1>Good Morning!</h1>;

} else {

message = <h1>Good Evening!</h1>;

}

return <div>{message}</div>;

* Benefits:
* Makes complex conditions easier to manage.
* Increases readability by separating logic from the return statement.

1. **Preventing Components from Rendering**

Sometimes, you may want to prevent a component from rendering based on a condition. You can do this by returning null.

* Example:

function Notification(props) {

if (!props.show) {

return null; // Prevents rendering

}

return <div>You have new messages!</div>;

}

Returning null tells React not to render anything, though the component’s lifecycle methods (like useEffect) will still run if the component is mounted.

This is useful for:

* Modal windows
* Tooltips
* Conditional messages
* Hiding components temporarily without unmounting their parent

**Conclusion**

Conditional rendering in React is a key feature for building flexible and interactive UIs. Whether using if statements, ternary operators, or logical expressions, React offers intuitive ways to control what the user sees. Element variables improve readability in complex conditions, and returning null provides an efficient way to prevent unnecessary components from rendering. Together, these tools help developers create responsive and dynamic user interfaces with ease.