**React Basics**

**1. What is React.js?**

React.js is a JavaScript library for building user interfaces, primarily used for developing **single-page applications (SPAs)**. It was developed by **Facebook** and allows developers to create **dynamic and interactive** web applications efficiently.

**Why use ReactJS?**

The key objective of ReactJs is to develop User Interfaces that enhance application speed and performance by using virtual DOM.

**Key Features:**

**Component-Based Architecture** – UI is built using reusable components.  
**Virtual DOM** – Faster updates by rendering only changed parts of the UI.  
**Declarative UI** – The UI updates automatically when the state changes.

**Example:**

import React from 'react';

function Welcome() {

return <h1>Welcome to React!</h1>;

}

export default Welcome;

**2. What are the advantages of using React?**

**Improves Performance** – Uses Virtual DOM for efficient rendering.  
**Reusability** – Components can be reused across the project.  
**Strong Ecosystem** – Supports third-party libraries like Redux, Material-UI, etc.  
**Large Community** – React has a huge developer community and resources.

**3. Disadvantages of using React?**

Most of the code is written in JSX, meaning that HTML and CSS are part of the JavaScript code. This can be perplexing because most other frameworks like to keep HTML separate from the JavaScript code.

ReactJS has a huge file size.

**3. What is JSX, and how is it different from HTML?**

JSX (JavaScript XML) is a syntax extension that allows writing **HTML-like code** inside JavaScript files. It makes the UI code **more readable and easier to maintain**.

**Difference Between JSX and HTML:**

JSX uses className instead of class (because class is a reserved word in JavaScript).  
JSX uses {} to embed JavaScript expressions.

**Example:**

**HTML:**

<h1>Hello, World!</h1>

**JSX:**

const element = <h1>Hello, World!</h1>;

**4. What are components in React?**

React components are **reusable and independent** UI elements. Components help to break the UI into smaller, manageable parts.

**Types of Components:**

**Functional Components** – Simple JavaScript functions that return JSX.  
**Class Components** – JavaScript classes that extend React.Component.

**Example (Functional Component):**

function Greeting(props) {

return <h1>Hello, {props.name}!</h1>;

}

**5. What is the difference between functional and class components?**

| **Feature** | **Functional Component** | **Class Component** |
| --- | --- | --- |
| **Definition** | A JavaScript function | A JavaScript class that extends React.Component |
| **State Management** | Uses useState hook | Uses this.state |
| **Lifecycle Methods** | Uses hooks like useEffect | Uses built-in lifecycle methods (componentDidMount, etc.) |
| **Performance** | Faster and optimized | Slightly heavier due to class instances |
| **Usage** | Recommended in modern React | Used in older projects |

**Example (Class Component):**

import React, { Component } from 'react';

class Greeting extends Component {

render() {

return <h1>Hello, {this.props.name}!</h1>;

}

}

export default Greeting;