**DIGITAL NURTURE 4.0 JavaFSE**

**WEEK 6**

**React descriptive answers**

### **What is SPA (Single-Page Application) and its benefits?**

* A **Single-Page Application (SPA)** is a web app that **loads a single HTML page** and dynamically updates content without reloading the page.
* Uses **JavaScript frameworks like React, Angular, or Vue** to render content.
* **Benefits:**
  + Fast and smooth user experience (no full page reloads).
  + Efficient data loading (fetches only what’s needed via APIs).
  + Better for dynamic and interactive applications.

### **What is React and how does it work?**

* **React** is a JavaScript library developed by Facebook for building **user interfaces**, mainly for SPAs.
* **How it works:**
  + Uses a **component-based structure** (UI is split into reusable pieces).
  + Uses a **Virtual DOM** to efficiently update changes on the page.
  + React compares the Virtual DOM with the real DOM and updates only the changed parts (faster rendering).

### **SPA vs. MPA (Multi-Page Application)**

| **Feature** | **SPA** | **MPA** |
| --- | --- | --- |
| **Page Loading** | Loads a single page and updates dynamically | Reloads a new page for each request |
| **Speed** | Faster after first load | Slower (full page reload) |
| **SEO** | Harder to optimize (needs extra setup) | Better SEO support by default |
| **Complexity** | Needs client-side routing (React Router) | Simpler, traditional approach |

### **Pros & Cons of Single-Page Applications**

**Pros:**

* Fast, responsive UI
* Reduces server load
* Better user experience

**Cons:**

* SEO challenges
* Larger initial load time
* More complex setup (needs routing, state management)

### **What is Virtual DOM?**

* The **Virtual DOM** is a **lightweight copy of the real DOM** that React uses to track changes.
* React updates the Virtual DOM first, compares it (diffing) with the real DOM, and only updates the parts that changed.
* This makes rendering **faster and more efficient**.

### **Features of React**

* Component-based architecture
* Virtual DOM for performance
* One-way data binding
* JSX (JavaScript + HTML syntax)
* Strong ecosystem (hooks, state management)
* Cross-platform (React Native)

### **Explain React Components**

* Components are the **building blocks of a React application**.
* Each component is a **reusable piece of UI** that returns JSX (HTML + JavaScript).
* They can be **functional (simple)** or **class-based (with state and lifecycle methods)**.

### **Differences between Components and JavaScript Functions**

| **React Component** | **JavaScript Function** |
| --- | --- |
| Returns **JSX (UI)** | Returns a value (number, string, object, etc.) |
| Can manage **state and lifecycle** (class components) | Cannot manage state/lifecycle by itself |
| Used to build UI, reusable across app | Used for general logic or calculations |
| React automatically re-renders when state/props change | No automatic UI update mechanism |

### **Types of Components**

1. **Functional Component** –
   * Simple function returning JSX.
   * Can use React Hooks for state and lifecycle.
2. **Class Component** –
   * Uses ES6 class syntax.
   * Can use **state**, **constructor**, and **lifecycle methods** (componentDidMount, etc.).

### **Class Component**

* Defined as a **class that extends React.Component**.
* Must include a **render() method** to return JSX.
* Can use **state** and **lifecycle hooks**.

### **Function Component**

* A simple **JavaScript function** that returns JSX.
* Initially stateless, but can use **Hooks** (useState, useEffect) for state and lifecycle.
* Easier and preferred for most modern React apps.

### **Component Constructor**

* Used only in **class components**.
* Initializes **state** and binds event handlers.
* Syntax:

constructor(props) {

super(props);

this.state = { key: "value" };

}

### render() **Function**

* Present only in **class components**.
* Responsible for **returning the JSX** to be displayed.
* React automatically calls it whenever state/props change.