#### ****Objective Answers – Week6\_HandsOn1****

#### 1. Define SPA and its benefits

A **Single Page Application (SPA)** is a modern web application that interacts with the user by dynamically rewriting the current page, instead of loading entire new pages from a server. This means that a SPA loads a single HTML file and updates only portions of the content as needed, using JavaScript.

**Benefits of SPA** include:

* **Faster navigation**: Only relevant content is updated, avoiding full-page reloads.
* **Better user experience**: It feels more like using a desktop or mobile app.
* **Reduced server load**: Since only data (not full HTML pages) is sent after the initial load.
* **Caching**: Once loaded, SPA resources can be cached efficiently, speeding up repeated visits.

#### 2. Define React and identify its working

**React** is an open-source JavaScript library developed by Facebook used for building interactive user interfaces. It allows developers to create reusable UI components that reflect real-time data.

React works by maintaining a **Virtual DOM** – a memory-based representation of the real DOM. When the state or props of a component change, React compares the new Virtual DOM with the previous one, detects what actually changed, and then efficiently updates only those parts of the actual DOM. This approach leads to faster and smoother performance, especially in complex applications.

#### 3. Identify the differences between SPA and MPA

| **Feature** | **SPA (Single Page App)** | **MPA (Multi Page App)** |
| --- | --- | --- |
| Loading | Loads only once and updates dynamically | Loads a new page from the server for each route |
| Speed | Faster after initial load | Slower due to full-page reloads |
| Interactivity | Highly interactive, feels like a mobile app | Less interactive |
| Development Focus | Frontend-heavy with client-side routing | Backend-focused with server-side rendering |
| Examples | Gmail, Facebook, Twitter | Amazon, eBay, government portals |

#### 4. Explain Pros & Cons of Single Page Application

**Pros:**

* **Improved performance**: Only data is transferred, not entire HTML pages.
* **Better UX**: Seamless transitions between views.
* **Reusable components**: Especially when using frameworks like React.
* **Efficient client-side routing**: Fast navigation without page refresh.

**Cons:**

* **SEO challenges**: Since content loads dynamically, search engines may not index it well.
* **Initial load time**: The full app must be loaded at once on first visit.
* **Browser support**: Heavily depends on JavaScript, which must be enabled.
* **Security**: More vulnerable to XSS attacks if not properly handled.

#### 5. Explain about React

**React** is not a full-fledged framework but a flexible library focused on building UI components. It allows developers to break down the UI into small, reusable, and manageable pieces called **components**. React supports both **functional** and **class-based components**, though modern development prefers functional ones with hooks.

It promotes the concept of **“learn once, write anywhere”**, meaning React can be used for web, mobile (React Native), and even desktop apps.

React’s declarative approach makes code easier to debug and maintain, and its vibrant ecosystem (React Router, Redux, etc.) supports complex app development.

#### 6. Define virtual DOM

The **Virtual DOM (V-DOM)** is a key concept in React. It is a lightweight, in-memory representation of the real DOM elements. React creates this V-DOM whenever there is a change in the state or props of a component.

Instead of updating the real DOM immediately (which is slow), React first updates the Virtual DOM. Then, it compares the new V-DOM with the old one using a technique called **“diffing”** and updates only the changed parts in the actual DOM. This makes React applications faster and more efficient than traditional DOM manipulation.

#### 7. Explain Features of React

Some major features of React include:

* **Component-Based Architecture**: UI is broken down into small, independent, reusable components.
* **JSX Syntax**: JavaScript + HTML syntax makes writing UI components more intuitive.
* **Virtual DOM**: Ensures efficient rendering by updating only changed elements.
* **One-Way Data Binding**: Ensures predictable data flow from parent to child components.
* **React Hooks**: Enable functional components to manage state and lifecycle features.
* **Unidirectional Flow**: Data flows in one direction, improving clarity and debugging.
* **React Native**: Allows code reuse for building mobile apps on Android and iOS.