WEEK-6

REACT

EXERCISE:1

Exercise 1:

1. Define SPA and its Benefits

SPA (Single Page Application) is a web application that loads a single HTML page and dynamically updates content as the user interacts with the app, without reloading the entire page.

Benefits of SPA:

- **Faster navigation** since only necessary content updates.
- **Smooth user experience** similar to desktop or mobile apps.
- **Reduces server load** as most resources are loaded once.
- **Efficient data handling** using APIs (like REST or GraphQL).

2. Define React and Identify Its Working

React is a JavaScript library developed by Facebook for building fast, interactive user interfaces, especially for SPAs.

How React Works:

- React uses a **component-based** architecture.
- Components are **reusable** and **encapsulated** units of UI.
- It maintains a **Virtual DOM** to update only the changed parts of the real DOM efficiently.
- Uses **JSX** (JavaScript XML) for writing HTML-like code in JavaScript.

3. Differences Between SPA and MPA

Feature	SPA (Single Page App)	MPA (Multi Page App)
Page Loading	Loads a single page, updates dynamically	Loads a new page from server on every request
Speed	Faster interactions after initial load	Slower due to full page reloads
User Experience	Seamless and smooth	May feel slower and less dynamic
Server Communication	API calls for data	Full page reload on every request
Example	Gmail, Facebook, React Apps	Amazon, Wikipedia, traditional websites

4. Pros & Cons of Single Page Applications

Pros:

- Smooth, app-like experience
- Faster response after first load
- Easier to debug with developer tools
- Good for mobile performance

Cons:

- Initial loading time can be higher
- SEO challenges (can be solved with SSR)
- JavaScript-dependent – may not work well if JS is disabled
- Security considerations (like XSS)

5. Explain About React

React is a powerful JavaScript library that allows developers to:

- Build reusable UI components.

- Use a **virtual DOM** for efficient rendering.
- Manage state and props to control how components behave.
- Leverage **hooks** (like useState, useEffect) for functional components.
- Use a strong ecosystem (React Router, Redux, etc.) for building complex apps.

6. Define Virtual DOM

The **Virtual DOM (VDOM)** is a lightweight copy of the real DOM maintained by React.

How It Works:

1. When a component state changes, React creates a new Virtual DOM tree.
2. It compares the new VDOM with the previous one (diffing).
3. Only the differences are updated in the real DOM (patching), improving performance.

7. Features of React

- **Component-Based**: UI is broken into independent, reusable components.
- **JSX**: JavaScript + XML syntax for creating UI elements.
- **Virtual DOM**: Efficient rendering by minimizing direct DOM updates.
- **Unidirectional Data Flow**: Data flows from parent to child using props.
- **Hooks**: Functions like useState, useEffect to manage state and side effects.
- **Declarative UI**: Describe how UI should look for a given state.
- **Strong Ecosystem**: Tools like Redux, React Router, etc.

CODE:

App.js:

```
import logo from './logo.svg';
import './App.css';

function App() {
  return (
    <div className="App">
      <header className="App-header">

        <h1>This is my React project</h1>

      </header>
    </div>
  );
}

export default App;
```

OUTPUT: