1**)Define SPA and its benefits ?**

**SPA** stands for **Single Page Application**.

* A web application or website that interacts with the user by **dynamically rewriting the current page**, rather than loading entire new pages from the server.
* **Examples:** Gmail, Google Maps, Facebook, GitHub.
* **Built with:** HTML, CSS, JavaScript frameworks like **React, Angular, Vue**, etc.
* **Benefit:** Faster, smoother user experience since only the content changes—not the whole page.

**2)Define React and identify its working**

**React** is a **JavaScript library** used to build **user interfaces**, especially for **Single Page Applications (SPAs)**.  
It was developed by **Facebook** and is used to create fast, dynamic, and interactive website

React allows developers to build **reusable components** that update efficiently when data changes.

**3)Identify the differences between SPA and MPA**

|  |  |
| --- | --- |
| **SPA vs MPA** |  |
| | **Feature** | **SPA (Single Page Application)** | **MPA (Multi Page Application)** | | --- | --- | --- | | **Definition** | Loads a single HTML page and dynamically updates content without reloading the whole page | Each interaction or page requires loading a new HTML page from the server | | **Page Reload** | no full page reload (uses AJAX/JavaScript for updates) | full page reload for every new page/view | | **Speed** | Fast after initial load (as only parts of the page update) | Slower, since each request loads an entire new page | | **User Experience** | Smooth and dynamic (like a mobile app) | Traditional web feel with flickering/loading | | **Routing** | Handled on the client side (e.g., React Router) | Handled on the server side (each page has a separate route) | |  |

**4) Pros and Cons of Single-Page Application (SPA)**

Pros of SPA

| **Advantage** | **Explanation** |
| --- | --- |
| **1. Fast and Responsive** | Only the content that changes is updated, not the whole page. This leads to a smoother user experience. |
| **2. Better UX** | Feels like a native mobile app — no flickering or full reloads. Ideal for modern interfaces. |
| **3. Efficient Data Loading** | Uses APIs (like REST/GraphQL) to load only required data, reducing bandwidth usage. |
| **4. Easy to Debug** | Tools like React Developer Tools or Vue DevTools make debugging simple. |
| **5. Code Reusability** | Components in frameworks like React or Vue are reusable and modular. |

Cons of SPA

| **Disadvantage** | **Explanation** |
| --- | --- |
| **1. Poor SEO Support** | Search engines may struggle to index dynamic content without extra setup like SSR (Server-Side Rendering). |
| **2. Initial Load is Slow** | The entire JS app is loaded at the start, which can make the first load slower. |
| **3. Security Risks** | Being a client-heavy app, SPAs are more exposed to XSS (Cross-Site Scripting) attacks. |
| **4. Complex State Management** | Managing app-wide state (like using Redux or Context API) can be tricky as the app grows. |
| **5. Browser History Issues** | Without proper routing setup (like React Router), back/forward navigation may not work as expected. |

**5) What is React?**

**React** is a **JavaScript library** used to build **user interfaces (UI)**, especially for **Single Page Applications (SPAs)**. It helps developers build fast, interactive, and component-based web applications.

**6)What is Virtual DOM?**

**Definition:**

The **Virtual DOM** is an **in-memory representation** of the actual DOM elements. React uses it to determine what has changed in the UI and updates **only the changed parts** — instead of reloading the entire page.

**Features of React**

**1. Component-Based Architecture**

* React apps are built using **components** — reusable and independent pieces of UI.
* Each component manages its own state and logic.
* Promotes **code reuse** and **modularity**.

**2. JSX (JavaScript XML)**

* React uses **JSX**, a syntax extension that lets you write HTML-like code inside JavaScript.
* It makes code more readable and concise.

**jsx**

Copy code

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

**3. Virtual DOM**

* React uses a **virtual copy** of the real DOM to track changes efficiently.
* Improves **performance** by minimizing actual DOM manipulation.

**4. One-Way Data Binding**

* Data flows from **parent to child components** only.
* This **unidirectional data flow** makes data management predictable and easy to debug.

**5. Declarative UI**

* You describe *what* the UI should look like.
* React updates the DOM to match that **declaration** automatically when data changes.

**6. State Management**

* React components can maintain **internal state**.
* For global state, tools like **Redux** or **Context API** are often used.

**7. React Router**

* Enables **navigation and routing** within a single-page application.
* Helps build **multi-view** applications without full page reloads.

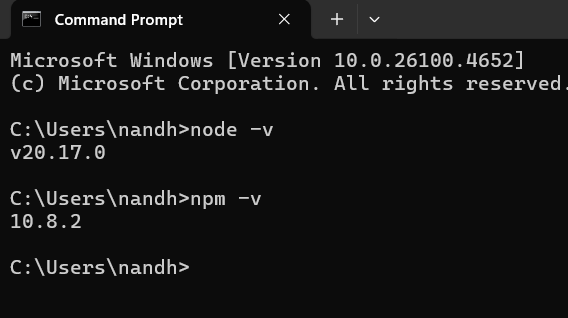
**8. Lifecycle Methods / Hooks**

* Class components use lifecycle methods (e.g., componentDidMount).
* Functional components use **React Hooks** (like useState, useEffect) for managing logic.

**9. Tooling Ecosystem**

* **Create React App (CRA)**: A tool to set up React projects quickly.
* **React Developer Tools**: Browser extension for debugging.
* Works well with modern tools like Webpack, Babel, Tailwind CSS, etc.

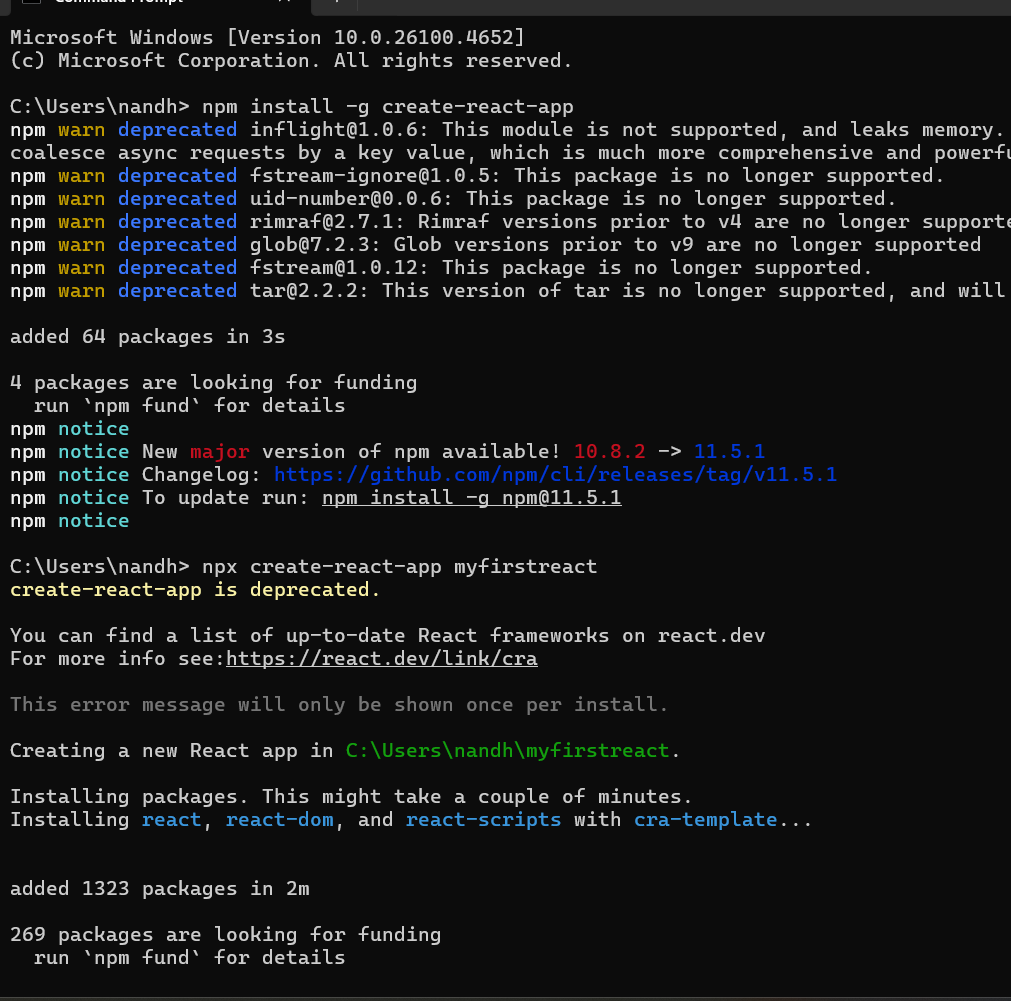
**2) To create a new React app, Install Nodejs and Npm from the following link:**



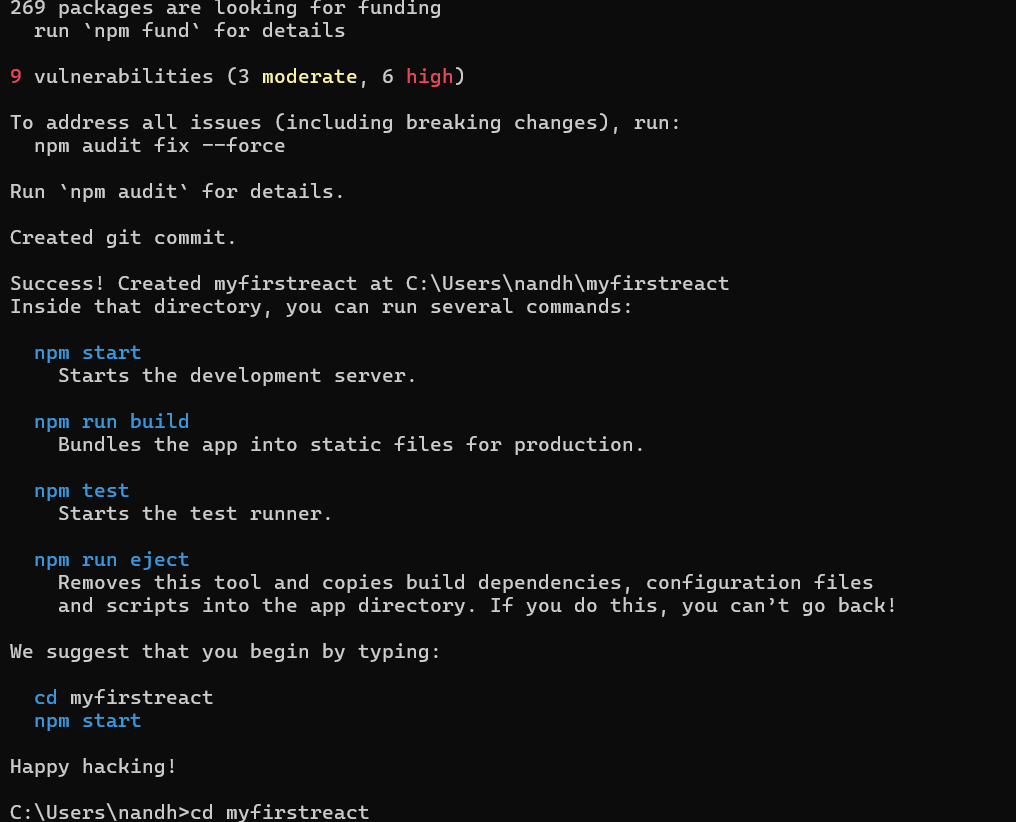
**2)Install Create-react-app by running the following command in the command prompt:**



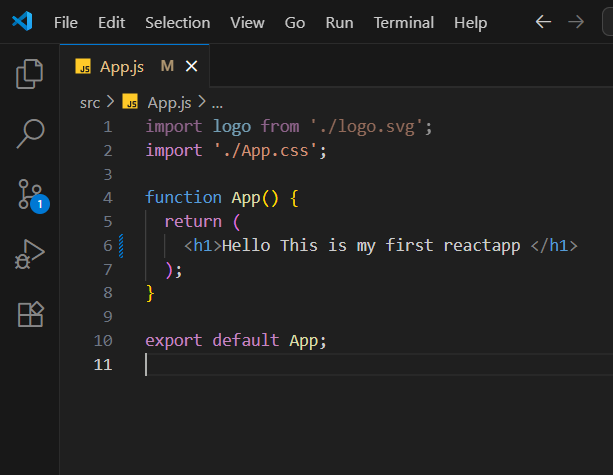
**To create a React Application with the name of “myfirstreact”, type the following command:**



**2)To create a React Application with the name of “myfirstreact”, type the following command:**

****

**Once the App is created, navigate into the folder of myfirstreact by typing the following command:**



3)I have opened a new browser window in the address bar

**Here is the output**

