1. Whatare various ways to add images into our App? Explain with code examples

Using the public folder

You can store images in the public folder (e.g., public/images) and reference them using a relative path from the public root. This does not require importing the image.

```
function App() {
  return (
    <div>
      <h1>Image from Public Folder</h1>
      <img src="/images/my-image.jpg" alt="Example from public folder" />
    </div>
  );
}
```

Importing images from the src folder

You can place images inside the src folder and import them into your component file. This method allows webpack to process and optimize the images during the build.

```
import React from 'react';
import myImage from './assets/my-image.jpg'; // Adjust path based on your file structure

function App() {
  return (
    <div>
      <h1>Imported Image</h1>
      <img src={myImage} alt="Example from src folder" />
    </div>
  );
}
```

Using an external image URL (e.g., from an API or CDN)

You can directly use image URLs in the src attribute, whether it's from an API response or a hosted source.

```
function App() {
  return (
    <div>
      <h1>Image from External URL</h1>
      <img src="https://example.com/image.jpg" alt="External image" />
    </div>
  );
}
```

2. What would happen if we do console.log(useState())?

useState() returns an array: [stateValue, setStateFunction].If you do console.log(useState()) directly, you're creating an extra state, which can break hook order Always store the result in variables, then log those.

3. How will useEffect behave if we don't add a dependency array ?

If you **do not provide a dependency array** to the useEffect hook, the effect will run **after every render** — both the initial render and every update that re-renders the component. If you put **expensive operations** (like API calls or animations) inside an effect **without a dependency array**, they'll run every time — which can cause **performance issues** or **unwanted side effects**.

4. What is an SPA, single page application ?

SPA (Single Page Application) is a modern web development approach where the entire application runs within a single HTML page. Instead of reloading the page every time the user navigates to a different route, SPAs use client-side routing to dynamically update the content. In React, this is typically handled using libraries like react-router-dom. When the route changes, only the corresponding component is rendered, and the page does not refresh. This leads to faster navigation, a smoother user experience, and better performance compared to traditional multi-page applications.

5. What is difference between Client Side Routing and Server Side Routing ?

Client-Side Routing is when all the routing logic is handled in the browser (on the client side) without reloading the page. In React, libraries like react-router-dom allow you to define routes that map to different components. When the user navigates to a new route, React dynamically loads the corresponding component and updates the URL, but the browser doesn't reload — the entire app still runs inside a single HTML file.

On the other hand, **Server-Side Routing** is the traditional approach where each route corresponds to a separate HTML page. When a user clicks a link or visits a new route, a request is sent to the server, which responds with a completely new HTML file. This causes a full page reload every time a route changes.