**React Portfolio Interview Q&A + Concepts**

### ✨ Common Interview Questions & Answers

#### Q: Why did you choose Vite over CRA or Next.js?

**A:** I chose Vite because it’s incredibly fast, especially during development. It offers instant server start without bundling, lightning-fast Hot Module Replacement (HMR), and optimized production builds using Rollup. It’s lighter than Create React App and less opinionated than Next.js, which gave me more flexibility.

#### Q: What is the dist folder?

**A:** dist stands for “distribution.” It’s the folder that Vite outputs after running npm run build. It contains the final, production-ready static assets like minified HTML, CSS, and JavaScript files. This is the folder you deploy to Netlify or other static hosting services.

#### Q: What is Rollup?

**A:** Rollup is a modern JavaScript module bundler used by Vite under the hood for production builds. It’s known for producing smaller, faster bundles by eliminating unused code (tree-shaking) and optimizing ES modules.

#### Q: What’s the difference between useEffect and useLayoutEffect?

**A:** - useEffect runs **after** the component renders and the browser paints the UI. It’s ideal for fetching data, setting up subscriptions, or modifying the DOM **without affecting layout**. - useLayoutEffect runs **before** the browser paints. Use it when you need to **measure or mutate layout** synchronously (e.g., measuring element positions, animations).

#### Q: What is the lifecycle of a React functional component?

**A:** 1. **Initialization**: Component function is invoked. 2. **Render**: JSX returns virtual DOM. 3. **Commit Phase**: React applies DOM changes. 4. **Effects Run**: useEffect or useLayoutEffect is called. 5. **Update**: Triggered when state/props change. 6. **Cleanup**: Effects are cleaned up before the next run or unmount.

#### Q: How is your project structured?

**A:** I structured it with modular React components: Home, About, Projects, Experience, and Contact, each in their own file. These are imported and rendered in App.tsx, which also controls navigation and scroll behavior.

#### Q: Why use react-router-dom?

**A:** It enables clean client-side routing in a single-page app. I used it to allow anchor links and prevent full page reloads. It provides a smoother experience and lets users jump to sections instantly.

#### Q: What is a CMS?

**A:** CMS stands for **Content Management System**. It lets non-developers manage website content via a UI without touching code. Examples: WordPress, Sanity, Contentful. In the future, I might use a CMS to manage project data or blog content.

#### Q: What’s in your package.json and why?

**A:** It includes script commands for development and production (npm run dev, npm run build), version-controlled dependencies like react, vite, tailwindcss, and typing support for TypeScript.

### 💡 Personal Experience & Lessons

#### Q: What did you learn while building this?

I learned how to structure and deploy a complete single-page React site using the modern Vite + Tailwind + TypeScript stack. It gave me confidence in working with a professional frontend toolchain.

#### Q: What would you do differently?

I’d start with a visual mockup in Figma to save time with layout tweaks. It would’ve helped me make more intentional design choices.

#### Q: What was the hardest bug you fixed?

The scroll position tracking logic. Getting the active section to update cleanly during scroll was tricky. I ended up using a distance-based method in useEffect to track the closest section.

#### Q: What are you most proud of?

It’s my first fully deployed project—and I built it solo from scratch. That’s huge for me.

#### Q: What do you want to add next?

Dark mode toggle, scroll animations, and maybe CMS integration for portfolio entries or blog posts.

Let me know if you’d like a condensed version of this for a resume, cover letter, LinkedIn post, or voiceover script.