

Complete React Developer Interview Handbook

Prepared for React Developers with 1-2 Years Experience

React Fundamentals

Q1. What is React?

Answer: React is a JavaScript library for building user interfaces using reusable components.

Q2. What is Virtual DOM?

Answer: A lightweight copy of the DOM used by React to efficiently update the UI.

Q3. What is JSX?

Answer: A syntax extension that allows HTML-like code inside JavaScript.

Q4. What are Components?

Answer: Reusable building blocks of a React application.

React Hooks

Q1. *useState*

Answer: Manages local component state.

Q2. *useEffect*

Answer: Handles side effects such as API calls and subscriptions.

Q3. *useMemo*

Answer: Memoizes expensive calculations.

Q4. *useCallback*

Answer: Memoizes functions to prevent unnecessary re-renders.

Q5. *useRef*

Answer: Stores mutable values without causing re-renders.

State Management

Q1. Redux

Answer: Centralized state management library.

Q2. Redux Toolkit

Answer: Official recommended way to write Redux logic.

Q3. Context API

Answer: Built-in React feature for sharing state globally.

Routing

Q1. React Router

Answer: Handles navigation between pages.

Q2. Protected Routes

Answer: Restrict access based on authentication.

API Integration

Q1. Axios vs Fetch

Answer: Axios provides interceptors and automatic JSON transformation.

Q2. Error Handling

Answer: Use try/catch or .catch().

Q3. Loading States

Answer: Display loaders while fetching data.

JavaScript ES6+

Q1. let vs const vs var

Answer: Scope and mutability differences.

Q2. Promises

Answer: Handle asynchronous operations.

Q3. Async/Await

Answer: Cleaner syntax for promises.

Q4. Closures

Answer: Functions retaining access to outer scope variables.

Q5. Destructuring

Answer: Extract values from objects and arrays.

HTML & CSS

Q1. Flexbox

Answer: One-dimensional layout system.

Q2. Grid

Answer: Two-dimensional layout system.

Q3. Responsive Design

Answer: Supports multiple screen sizes.

Q4. Semantic HTML

Answer: Improves accessibility and SEO.

Performance Optimization

Q1. React.memo

Answer: Avoids unnecessary component re-renders.

Q2. Code Splitting

Answer: Loads code only when needed.

Q3. Lazy Loading

Answer: Loads components on demand.

Q4. Optimization Techniques

Answer: Memoization, pagination, virtualization.

Testing

Q1. Jest

Answer: JavaScript testing framework.

Q2. Unit Testing

Answer: Tests individual units of code.

Q3. Component Testing

Answer: Tests React components.

Debugging

Q1. React DevTools

Answer: Inspect component hierarchy and state.

Q2. Browser DevTools

Answer: Analyze network, performance, and console logs.

Scenario Questions

Q1. Share data between siblings?

Answer: Lift state up or use Context/Redux.

Q2. Prevent prop drilling?

Answer: Use Context API or Redux.

Q3. Optimize large lists?

Answer: Pagination or virtualization.

Project-Based Questions

Q1. Describe your project

Answer: Explain architecture, tech stack, role, challenges, and achievements.

Q2. Authentication

Answer: JWT-based authentication and protected routes.

Q3. API Architecture

Answer: Reusable service layer using Axios.

HR Questions

Q1. Tell me about yourself

Answer: Focus on experience, skills, and achievements.

Q2. Why should we hire you?

Answer: Highlight React, Node.js, problem-solving, and teamwork.

Q3. Where do you see yourself in 5 years?

Answer: Growth into senior/full-stack responsibilities.