1️⃣ React – Basics & Fundamentals (10 Questions)

1. What is React and why is it used?
2. Difference between React and traditional MVC frameworks.
3. What is JSX? How does it work?
4. Difference between functional and class components.
5. What are props and state?
6. What is Virtual DOM and how does it improve performance?
7. What is one-way data binding?
8. What are keys in React lists and why are they important?
9. What is the role of render() in React?
10. Explain the React component lifecycle (high level).

2️⃣ React – Algorithm Solving & DSA (UI / Logic-Oriented) (10 Questions)

These are **coding-round style React questions** commonly asked.

1. Reverse a string in a React component.
2. Remove duplicate values from an array and render the list.
3. Implement search filter for a list.
4. Sort table data on column click.
5. Implement pagination logic in React.
6. Debounce an input field without libraries.
7. Find max/min value from API response.
8. Group array of objects by a property.
9. Implement simple autocomplete logic.
10. Optimize rendering of large lists.

3️⃣ React – Intermediate (Frequently Asked Interview Questions) (10 Questions)

1. What are hooks? Why were they introduced?
2. Explain useState and useEffect.
3. How does dependency array work in useEffect?
4. Difference between controlled and uncontrolled components.
5. What is lifting state up?

6. Difference between useRef and useState.
7. What is Context API? When should you use it?
8. What is memoization in React?
9. Explain React.memo, useMemo, and useCallback.
10. How does React handle forms?

4️⃣ React – Advanced (Architecture & Ecosystem) (10 Questions)

1. Explain React reconciliation algorithm.
2. What is Fiber architecture?
3. What are concurrent features in React?
4. Explain Suspense and lazy loading.
5. What are error boundaries?
6. What is Server-Side Rendering (SSR)?
7. Difference between CSR, SSR, and SSG.
8. What is hydration in React?
9. How does React handle state updates internally?
10. How does React scale for large applications?

5️⃣ React State Management (10 Questions)

1. When should you use local state vs global state?
2. Explain Redux architecture.
3. Difference between Redux and Context API.
4. What are actions, reducers, and store?
5. What is middleware in Redux?
6. Explain Redux Thunk vs Redux Saga.
7. What is immutability and why is it important?
8. How does Redux Toolkit simplify Redux?
9. How do you handle async API calls?
10. How do you prevent unnecessary re-renders?

6️⃣ React Routing, Security & Performance (10 Questions)

1. How does React Router work?
2. Difference between BrowserRouter and HashRouter.
3. How do you protect routes?
4. How do you handle authentication in React?
5. How do you store JWT securely?
6. What is code splitting?
7. How do you improve React performance?
8. What is lazy loading?
9. How do you handle XSS in React?
10. How do you optimize bundle size?

7️⃣ Design Patterns in React (10 Questions)

1. What design patterns are commonly used in React?
2. Explain Container-Presenter pattern.
3. What is Higher-Order Component (HOC)?
4. Explain Render Props pattern.
5. What is Compound Component pattern?
6. How is Observer pattern used in React?
7. Explain Flux architecture.
8. How is Dependency Injection achieved in React?
9. What is Atomic Design?
10. How do design patterns improve maintainability?

8️⃣ React Capstone Project – Interview Questions

A. Architecture & Design (10 Questions)

1. Explain your React project architecture.
2. Why did you choose this folder structure?
3. How did you manage shared components?
4. How did you handle scalability?
5. How did you integrate APIs?
6. How did you handle authentication & authorization?

7. How did you manage environment configurations?
8. How did you implement state management?
9. How did you handle role-based access?
10. What would you refactor if given a chance?

B. Coding & Implementation (10 Questions)

1. Explain a complex component you built.
2. How did you handle performance issues?
3. How did you handle API errors globally?
4. How did you manage side effects?
5. How did you handle form validations?
6. How did you implement reusable components?
7. How did you manage large datasets?
8. How did you write unit tests?
9. Explain a critical bug you fixed.
10. What is the most challenging React feature you used?

C. Production, DevOps & Testing (10 Questions)

1. How was your React app deployed?
2. How did you manage environment variables?
3. What build tools did you use?
4. How did you optimize production builds?
5. How did you handle logging in frontend?
6. How did you monitor performance?
7. How did you handle browser compatibility?
8. What CI/CD pipeline did you use?
9. How did you ensure security in production?
10. How did you handle rollback and hotfixes?