1. What is Redux and what role does it play in JavaScript applications?

2. Explain the principle of "single source of truth" in Redux and its significance.

3. Why is state immutability important in Redux?

4. How do you install and set up Redux in a React application?

5. What are Redux actions and how are they used to manage application state?

6. Describe the role of reducers in Redux and how they contribute to state management.

7. What is a Redux store and how does it maintain the application state?

8. Differentiate between presentational and container components in React-Redux applications.

9. How do presentational components differ from container components in terms of responsibilities?

10. Explain the concept of higher-order components (HOCs) and their usage in React-Redux applications.

11. Provide examples of scenarios where higher-order components are useful in React-Redux.

12. What is the React Context API and how does it facilitate data sharing between components?

13. Describe the process of sharing data between components using React Context.

14. How does React Context differ from Redux in terms of state management?

15. When would you choose Redux over React Context for state management in a React application?

16. How does Redux help in maintaining a predictable state container in JavaScript apps?

17. What are the benefits of using Redux in large-scale applications?

18. Explain how Redux enhances the scalability and maintainability of React applications.

19. How do you handle asynchronous operations in Redux?

20. Discuss the role of middleware in Redux and provide examples of popular middleware libraries.