1.What is React?

React is an open-source JavaScript library. It is used for building user interface (UI). React simplifies the creation of SPA by using reusable components.

2.what are the key features of reacts?

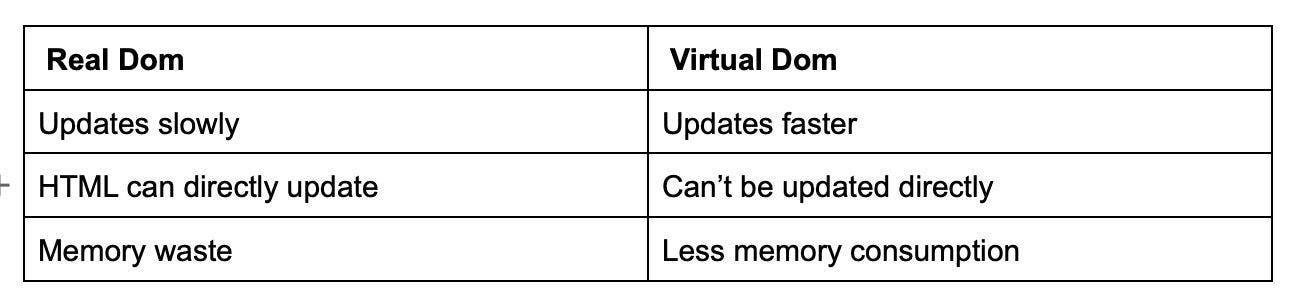
* Virtual DOM
* Component based architecture.
* Reusability & Composition.
* JSX (JavaScript XML)
* React Hooks.

3.What is Virtual DOM in React?

React uses a virtual DOM to efficiently update the UI without re-render the entire page, which helps improve performance and make the application more responsive.

4.Differents btw DOM and Virtual DOM?

|  |  |
| --- | --- |
| **DOM** | **Virtual DOM** |
| DOM is actual representation of the webpage | Virtual DOM is lightweight copy of the DOM |
| Re-renders the entire pages when updates occur | Re-renders only the changed parts efficiently. |
| Can be slower, especially with frequent updates | Optimized for faster rendering. |
| Suitable for static websites and simple application | Ideal for dynamic and complex single page application with frequent updates |



4. What are React Components?

In react, a component is a **reusable building block** for creating user interfaces.

5. what is Single Page Application?

SPA is a web application that have only one single web page. And whenever user do some action on the website, then in response content is dynamically updated without refreshing or loading a new page.

6. What are the 5 advantages of React?

* Simple to build SPA by using components.
* React is cross platform and open source (Free to use)
* Lightweight and very fast (Virtual DOM)
* Large Community and Ecosystem
* Testing is easy.

7. what is babel?

Babel in react is used to transpile JSX syntax into regular JavaScript which browser can understand.

8. What is Fragment?

Fragment is way to group multiple children elements. And using a fragment prevents the addition of unnecessary nodes to the DOM.

9. Difference btw Functional and Class Components?

|  |  |
| --- | --- |
| **Functional Components** | **Class Components** |
| Functional components are declared as a JS function | Class components are defined using JS Class |
| Stateless components, but with the help of hooks they can now manage the state | Stateful component by using the lifecycle methods. |
| Lifecycle method do not have | Lifecycle method have |
| No this keyword | This.props |
| Do not have render method | Render method have |

10. What is Prop Drilling?

Prop drilling is the process of **passing down props** through multiple layers of components.

Parent ------🡪 Parent Child ----------🡪 ParentGrantChid -----🡪

11. Why to avoid Prop Drilling? How many ways?

Maintenance, Complexity, Debugging.

We can avoid prop drilling using Context API, using Redux, Call back function and custom Hooks.

Root APP Root APP Data

Parent 1 Parent2 Parent 1 Parent2

Child 1 child 2 Child 1 child 2

12. What is Routing in react?

Routing allows you to create a single page web application with Navigation without the need for a full-page refresh.

13. What is React Router?

React router is a library for handling routing and enables navigation and rendering of different components based on the URL

14. How to implement Routing in React?

Import ( Routes, Route, Link) from react-router-dom.

15. What **are React Hooks**?

React Hooks are inbuilt function provided by react that allow functional components to use state and lifecycle features.

We need to import react library like,

Import React, { useState } from react

16. Types of React Hooks?

**useState**: Manages state in functional components.

**useEfect**: Performs side effects in functional components (e.g., data fetching, subscriptions).

**useContext**: Accesses context values in functional components.

**useReducer**: Manages more complex state logic using a reducer function (like Redux reducers).

**useCallback**: Memoizes callback functions to prevent unnecessary re-renders of child components.

**useMemo**: Memoizes the result of a computation to avoid recalculating on every render.

**useRef**: Provides a way to access and manipulate DOM elements or persist mutable values without causing re-renders.

**useLayoutEffect**: Similar to useEffect, but fires synchronously after all DOM mutations. Useful for reading layout and synchronously re-rendering.

**useTransition**: Allows you to manage the transition state for updates that can be deferred, improving responsiveness.

**useDeferredValue**: Defers updates to a value until the browser is idle, allowing for smoother transitions.

**useImperativeHandle**: Customizes the instance value exposed when using ref in parent components.

17. What is useState() Hooks?

useState hooks enables functional components to **manage state**.

useState() function ;accept the initial state value as the parameter and returns an array with two elements.

1. The first element is the current state value (count).
2. Second element is the function that is used to update the state (setCount)

The concept of assign array elements to individual variable is called **array destructuring**

EX: const [count, setCount] = useState(0)

18. What is useEffect in React Hooks?

useEffect Hook in React is used **to perform side effects** in functional components.

For examples: Data fetching from API, subscriptions, or any other operations that needs to be performed after the component has been rendered.

It will accept two parameters 1. Effect function 2. Dependency array.

19. What is Dependency Array in useEffect hooks?

One of the arguments that the **useEffect** accepts, after the callback function is the dependency array. This array defines the list of variables that if changed will trigger the callback function.

1. **Skipping the argument:**

useEffect(() => {  
 // Runs in each re-render of the component  
})

1. Empty Array:

useEffect(() => {  
// Runs only when the component is registered  
}, [])

1. Array of dependencies

useEffect(() => {  
// Runs if x or y changes  
}, [x, y])

20. What is useContext Hooks?

useContext in React provides a way to pass data from parent to child components without using props.

21. What is createContext methods? What are Provider and Consumer properties?

CreateContext() function returns an object with provider and Consumer properties.

Provider property is responsible **for providing the context value** to all its child components.

Consumer property can be used **to consume the context value** in child components.

22. Controlled and uncontrolled components?

|  |  |
| --- | --- |
| **Controlled Components** | **Uncontrolled components** |
| Values are controlled by React State. | Value are not controlled by react state. |
| Event handlers update react state | Value can be accessed directly from the DOM |
| Do not depend on useRef() | Commonaly uses useRef() to access from element |
| Re renders on state changes. | Less rerendering since are not directly tied to React state. |

23. What is Props and State ?

|  |  |
| --- | --- |
| **Props** | **State** |
| Props allow you to pass data from one component to other components as an argument. | state is used to manage data within a component |
| Props can be accessed by the child component. | State cannot be accessed by child components. |
| Props are immutable | State is mutable. |
| Props make components reusable. | State cannot make components reusable. |
| Props are read-only. | State changes can be asynchronous |

