1. What is Restful API?
representational state transfer and was created by computer scientist Roy Fielding

2. What is REST API example?
For example, a REST API would use a GET request to retrieve a record, a POST request to create one, a PUT request to update a record, and a DELETE request to delete one.
All HTTP methods can be used in API calls.
A well-designed REST API is similar to a website running in a web browser with built-in HTTP functionality.

3. What is react js?
React JS is a front end JavaScript library for building web and mobile user interfaces (TK)
It was developed by Facebook in 2011
React allows developers to build reusable UI components
It has the support of a large, open source community

4.What are react components? Why are components so important to react?
When it comes to using React, everything boils down to components.
Components are the building materials React uses to create website and application UI's.
Components break a UI down into reusable parts (one of React's core competencies).
React then renders each UI component as needed (separately from the others), which is a big part of React's fast performance speeds.

5. What's the main difference between props and state?
"State" describes a default data value in a React component that can change over time (usually based on user actions that call for changes in a UI).
"Props" (or properties) describe the way a React component is configured. Props do not change.
In summary: state is mutable (changeable based on user actions) while props are not

6.When would you use a class component over a functional component?
Functional components are the most basic kind of React component, defined by the components (unchanging) props.
Class components are more complicated React components that allow developers to execute component lifecycle methods and manage a component's state.
This means class components are used over functional components when you need to manage state or use component lifecycle methods

7. What are react events?
Events are reactions that are triggered by specific user actions like clicking on a UI button, hovering a mouse over a UI object, using keyboard commands with the UI, etc.

8. What is JSX?
JSX is an HTML-like syntax that let's developers write HTML style code in JavaScript, in case you prefer the readability of HTML to raw JavaScript
Developers do NOT need to use JSX to build websites or applications with React, but it can be a helpful tool for reducing overall code complexity (and Facebook encourages using it in their official React documentation)
If you have experience using JSX to build a website or app, mention some of the specific pros or cons you've encountered related to the process

9. What are virtual doms and how do they work?
When web browsers render HTML documents and turn them into a website or application on a screen, they create a representational tree of how the site or app is arranged called a Document Object Model (DOM).
Without React, your website or app will use HTML to update its DOM in order to make things "change" on screen without users needing to manually refresh a page.
React JS takes a different approach by creating a Virtual DOM—a copy of the site's "actual" DOM.

React uses this copy to determine what parts of the actual DOM need to change based
on a user's action. React then takes the change data from the Virtual DOM and
selectively updates the actual DOM (versus reloading the entire thing).
Over time, this leads to significant performance improvements for the website or application.

10. How would you debug an issue in react code? What debugging tools have you used?

Linters (eslint, jslint)
Debuggers (React Developer Tools)

11. React js vs react native - what's the difference?

Again, React JS is a front end, open source JavaScript library used for building UIs.
React Native, on the other hand, is an open source, MOBILE framework that allows developers
to use React on platforms like Android and iOS. "Native" is a reference to the fact that
React Native integrates React components with the native capabilities of these
mobile-specific platforms.

12. What are some of the major advantages to using react when building UIs?

Increased application performance via the Virtual DOM model
Improved coding efficiency with JSX
The ability to reuse components across multiple projects
Flexibility and extensibility through add-on tools provided by React's open source community

13. What are some of react's limitations?

React JS is a JavaScript library and not a full-blown framework,
meaning it might not be full service enough for some project
React's library is extremely large and can take additional time and
experience (past the initial learning phase) to really understand
JSX adds a new coding dimension for developers who haven't used before
(though it does have a gentle learning curve due to its similarity to HTML)

14. What is redux?

Redux helps developers build web applications that perform consistently and
are easy to test. It also includes live code editing and debugging tools.
Redux can be used with React or any other UI library.
React helps developers divide UIs into multiple components,
but doesn't have a specific way to keep track of state data.
Redux is a library that compliments React by providing a stable way to track this data.

15. What are the advantages of redux?

Redux adds a solid structure to React code, making your code easier to maintain,
and your intended coding results more predictable.
Redux includes developer tools that allow you to track your web application's performance
in real time – From actions to state changes, developers can track everything going on
in the application in real time.
Like React, Redux has strong community support and robust ecosystem.

16. What are the features of React?

JSX, Components, Virtual DOM
One-way data-binding: React's one-way data binding keeps everything modular and fast.
A unidirectional data flow means that when designing a React app,
you often nest child components within parent components.
High performance: React updates only those components that have changed,
rather than updating all the components at once. This results in much faster web applications.

17. Can web browsers read JSX directly?

Web browsers cannot read JSX directly. This is because they are built to only read
regular JS objects and JSX is not a regular JavaScript object

For a web browser to read a JSX file, the file needs to be transformed into a regular
JavaScript object. For this, we use Babel

18. Why use React instead of other frameworks, like Angular?

1) Easy creation of dynamic applications: React makes it easier to create dynamic
web applications because it provides less coding and provides more functionality,
whereas, with JavaScript applications, code tends to get complex very quickly.
2) Improved performance: React uses virtual DOM, which makes web applications perform faster.
Virtual DOM compares its previous state and updates only those components in the real DOM,
whose states have changed, rather than updating all the components — like conventional
web applications.
3) Reusable components: Components are the building blocks of any React application,
and a single app usually consists of multiple components.
These components have their own logic and controls, and they can be reused through the
application, which, in turn, dramatically reduces the development time of an application.
4) Unidirectional data flow: React follows a unidirectional data flow.
This means that when designing a React app, we often nest child components within
parent components. And since the data flows in a single direction, it becomes easier to
debug errors and know where the problem occurs in an application at the moment.
5) Dedicated tools for easy debugging: Facebook has released a chrome extension that we can use
to debug React applications. This makes the process of debugging React to web applications
faster and easier.

19. What is the difference between the ES6 and ES5 standards?

Components and function
exports vs export
require vs import

20. What are synthetic events in React?

Synthetic events combine the response of different browser's native events into one API,
ensuring that the events are consistent across different browsers.
The application is consistent regardless of the browser it is running in.
Here, preventDefault is a synthetic event.

21. Explain how lists work in React

We create lists in React as we do in regular JavaScript.
Lists display data in an ordered format
The traversal of lists is done using the map() function

22. Why is there a need for using keys in Lists?

Keys are very important in lists for the following reasons:
A key is a unique identifier and it is used to identify which items have changed,
been updated or deleted from the lists
It also helps to determine which components need to be re-rendered instead of re-rendering
all the components every time. Therefore, it increases performance, as only the
updated components are re-rendered

23. What are forms in React?

React employs forms to enable users to interact with web applications.

Using forms, users can interact with the application and enter the required
information whenever needed. Form contain certain elements, such as text fields,
buttons, checkboxes, radio buttons, etc
Forms are used for many different tasks such as user authentication, searching,
filtering, indexing, etc

24. What is an arrow function and how is it used in React?

An arrow function is a short way of writing a function to React.
It is unnecessary to bind ‘this’ inside the constructor when using an arrow function.
This prevents bugs caused by the use of ‘this’ in React callbacks.

25. How is React different from Angular?

Angular React
Author: Google, Facebook
Architecture: Complete MVC, View layer of MVC
DOM: Real DOM, Virtual DOM
Data-Binding: Bi-directional, Uni-directional
Rendering: Client-Side, Server-Side
Performance: Comparatively Slow, Faster due to Virtual DOM

26. What are components in React?

Components are the building blocks of any React application, and a single app
usually consists of multiple components. A component is essentially
a piece of the user interface.
It splits the user interface into independent, reusable parts that can be processed separately.

There are two types of components in React:
Functional Components: These types of components have no state of their own and
only contain render methods, and therefore are also called stateless components.
They may derive data from other components as props (properties).

Class Components: These types of components can hold and manage their own state and
have a separate render method to return JSX on the screen.
They are also called Stateful components as they can have a state.

27. What is the use of render() in React?

It is required for each component to have a render() function.
This function returns the HTML, which is to be displayed in the component.
If you need to render more than one element, all of the elements must be inside
one parent tag like <div>, <form>.

28. What is a state in React?

The state is a built-in React object that is used to contain data or information about
the component. The state in a component can change over time, and whenever it changes,
the component re-renders.
The change in state can happen as a response to user action or system-generated events.
It determines the behavior of the component and how it will render.

29. What are props in React?

Props are short for Properties. It is a React built-in object that stores the value
of attributes of a tag and works similarly to HTML attributes.
Props provide a way to pass data from one component to another component.
Props are passed to the component in the same way as arguments are passed in a function.

30. What are the differences between state and props?

State Props
1) Use: Holds information about the components,
Allows to pass data from one component to other components as an argument
2) Mutability: Is mutable, Are immutable
3) Read-only: Can be changed, Are read-only
4) Child components: Child Components cannot access, Child component can access
5) stateless components: Cannot have state, can have props

31. What is higher-order component in React?

A higher-order component acts as a container for other components.
This helps to keep components simple and enables re-usability.
They are generally used when multiple components have to use a common logic.

32. What are differences between class and functional components?

Class Components, Functional Components
State: Can hold or manage state, Cannot hold or manage state
Simplicity: Complex as compared to the stateless component, Simple and easy to understand
Lifecycle methods: Can work with all lifecycle methods, Does not work with any lifecycle method
Reusability: Can be reused, Cannot be reused

33. Explain the lifecycle methods of components

getInitialState(): This is executed before the creation of the component.
componentDidMount(): Is executed when the component gets rendered and placed on the DOM.
shouldComponentUpdate(): Is invoked when a component determines changes to the DOM and returns
a "true" or "false" value based on certain conditions.
componentDidUpdate(): Is invoked immediately after rendering takes place.
componentWillUnmount(): Is invoked immediately before a component is destroyed and unmounted permanently.

34. What are the components of Redux?

Store: Holds the state of the application.
Action: The source information for the store.
Reducer: Specifies how the application's state changes in response to actions sent to the store.

35. What is the Flux?

Flux is the application architecture that Facebook uses for building web applications.
It is a method of handling complex data inside a client-side application and manages
how data flows in a React application.
There is a single source of data (the store) and triggering certain actions is the only way
to update them. The actions call the dispatcher, and then the store is triggered and
updated with their own data accordingly.
When a dispatch has been triggered, and the store updates, it will emit a change event that
the views can rerender accordingly.

36. How is Redux different from Flux?

Redux, Flux
1) Redux is an open-source JavaScript library used to manage application State,
Flux is an architecture and not a framework or library
2) Store's state is immutable, Store's state is mutable
3) Can only have a single-store, Can have multiple stores
4) Uses the concept of reducer, Uses the concept of the dispatcher

37. What is React Router?

React Router is a routing library built on top of React, which is used to create routes
in a React application.

38. Why do we need to use React Router?

It maintains consistent structure and behavior and is used to develop single-page
web applications.
Enables multiple views in a single application by defining multiple routes in the
React application.

39. What do you understand by Virtual DOM? Explain its working.

A virtual DOM is a lightweight JavaScript object which originally is just the copy of real DOM.
It is a node tree that lists the elements, their attributes and content as Objects and their

properties. React’s render function creates a node tree out of the React components.
It then updates this tree in response to the mutations in the data model which is caused
by various actions done by the user or by the system.
This Virtual DOM works in three simple steps.
1) Whenever any underlying data changes, the entire UI is re-rendered in Virtual DOM
representation.
2) Then the difference between the previous DOM representation and the new one is calculated.
3) Once the calculations are done, the real DOM will be updated with only the things that
have actually changed.

40. Differences between stateful and stateless components.

Stateful Component, Stateless Component
1) Stores info about component's state change in memory,
Calculates the internal state of the components
2) have authority to change state, Do not have the authority to change state
3) Contains the knowledge of past, current and possible future changes in state
Contains no knowledge of past, current and possible future state changes.
4) Stateless components notify them about the requirement of the state change, then they
send down the props to them,
They receive the props from the Stateful components and treat them as callback functions.

41. What are the different phases of React component's lifecycle?

1) Initial Rendering Phase: This is the phase when the component is about to start
its life journey and make its way to the DOM.
2) Updating Phase: Once the component gets added to the DOM, it can potentially update
and re-render only when a prop or state change occurs. That happens only in this phase.
3) Unmounting Phase: This is the final phase of a component’s life cycle in which the
component is destroyed and removed from the DOM.

42. What is an event in React?

In React, events are the triggered reactions to specific actions like mouse hover,
mouse click, key press, etc. Handling these events are similar to handling events
in DOM elements. But there are some syntactical differences like:
1) Events are named using camel case instead of just using the lowercase.
2) Events are passed as functions instead of strings.
The event argument contains a set of properties, which are specific to an event.
Each event type contains its own properties and behavior which can be accessed via
its event handler only.

43. What are synthetic events in React?

Synthetic events are the objects which act as a cross-browser wrapper around the browser’s
native event. They combine the behavior of different browsers into one API. This is done to
make sure that the events show consistent properties across different browsers.

44. What do you understand by refs in React?

Refs is the short hand for References in React. It is an attribute which helps to store
a reference to a particular React element or component, which will be returned by the
components render configuration function. It is used to return references to a particular
element or component returned by render(). They come in handy when we need DOM measurements
or to add methods to the components.

45. List some of the cases when you should use Refs.

Following are the cases when refs should be used:

1) When you need to manage focus, select text or media playback
2) To trigger imperative animations
3) Integrate with third-party DOM libraries

46. What do you know about controlled and uncontrolled components?
controlled components, uncontrolled components
1. They do not maintain their own state, They maintain their own state
2. Data is controlled by the parent component, Data is controlled by the DOM
3. They take in the current values through props and then notify the changes via callbacks,
Refs are used to get their current values

47. What are higher order components(HOC)?
Higher Order Component is an advanced way of reusing the component logic. Basically, it's a pattern that is derived from React's compositional nature. HOC are custom components which wrap another component within it. They can accept any dynamically provided child component but they won't modify or copy any behavior from their input components.
You can say that HOC are 'pure' components.

48. What can you do with HOC?
HOC can be used for many tasks like:

Code reuse, logic and bootstrap abstraction
Render High jacking
State abstraction and manipulation
Props manipulation

49. What are Pure Components?
Pure components are the simplest and fastest components which can be written. They can replace any component which only has a render(). These components enhance the simplicity of the code and performance of the application.

50. What is the significance of keys in React?
Keys are used for identifying unique Virtual DOM Elements with their corresponding data driving the UI. They help React to optimize the rendering by recycling all the existing elements in the DOM. These keys must be a unique number or string, using which React just reorders the elements instead of re-rendering them.
This leads to increase in application's performance.

51. What were the major problems with MVC framework?
Following are some of the major problems with MVC framework:

1) DOM manipulation was very expensive
2) Applications were slow and inefficient
3) There was huge memory wastage
4) Because of circular dependencies, a complicated model was created around models and views

52. Explain Flux.
Flux is an architectural pattern which enforces the uni-directional data flow.
It controls derived data and enables communication between multiple components using a central Store which has authority for all data. Any update in data throughout the application must occur here only. Flux provides stability to the application and reduces run-time errors.

53. What is Redux?
Redux is one of the most trending libraries for front-end development in today's marketplace.
It is a predictable state container for JavaScript applications and is used for the entire applications state management. Applications developed with Redux are easy to test and can run in different environments showing consistent behavior.

54. What are the three principles that Redux follows?

1) Single source of truth: The state of the entire application is stored in an object/ state tree within a single store. The single state tree makes it easier to keep track of changes over time and debug or inspect the application.
2) State is read-only: The only way to change the state is to trigger an action. An action is a plain JS object describing the change. Just like state is the minimal representation of data, the action is the minimal representation of the change to that data.
3) Changes are made with pure functions: In order to specify how the state tree is transformed by actions, you need pure functions. Pure functions are those whose return value depends solely on the values of their arguments.

55. What do you understand by "Single source of truth"?

Redux uses 'Store' for storing the application's entire state at one place. So all the component's state are stored in the Store and they receive updates from the Store itself. The single state tree makes it easier to keep track of changes over time and debug or inspect the application.

56. List down the components of Redux

Redux is composed of the following components:

1) Action – It's an object that describes what happened.
2) Reducer – It is a place to determine how the state will change.
3) Store – State/ Object tree of the entire application is saved in the Store.
4) View – Simply displays the data provided by the Store.

57. How are Actions defined in Redux?

Actions in React must have a type property that indicates the type of ACTION being performed. They must be defined as a String constant and you can add more properties to it as well. In Redux, actions are created using the functions called Action Creators.

58. Explain the role of Reducer.

Reducers are pure functions which specify how the application's state changes in response to an ACTION. Reducers work by taking in the previous state and action, and then it returns a new state. It determines what sort of update needs to be done based on the type of the action, and then returns new values. It returns the previous state as it is, if no work needs to be done.

59. What is the significance of Store in Redux?

A store is a JavaScript object which can hold the application's state and provide a few helper methods to access the state, dispatch actions and register listeners. The entire state/ object tree of an application is saved in a single store. As a result of this, Redux is very simple and predictable. We can pass middleware to the store to handle the processing of data as well as to keep a log of various actions that change the state of stores. All the actions return a new state via reducers.

60. Why is switch keyword used in React Router v4?

Although a <div> is used to encapsulate multiple routes inside the Router. The 'switch' keyword is used when you want to display only a single route to be rendered amongst the several defined routes. The <switch> tag when in use matches the typed URL with the defined routes in sequential order. When the first match is found, it renders the specified route. Thereby bypassing the remaining routes.

61. Why do we need a Router in React?

A Router is used to define multiple routes and when a user types a specific URL, if this URL matches the path of any 'route' defined inside the router, then the user is redirected to that particular route. So basically, we need to add a Router library to our app that allows creating multiple routes with each leading to us a unique view.

62. List down the advantages of React Router.

1) Just like how React is based on components, in React Router v4, the API is 'All About Components'. A Router can be visualized as a single root component (<BrowserRouter>) in which we enclose the specific child routes (<route>).
2) No need to manually set History value: In React Router v4, all we need to do is wrap our routes within the <BrowserRouter> component.
3) The packages are split: Three packages one each for Web, Native and Core. This supports the compact size of our application. It is easy to switch over based on a similar coding style.

63. What are refs used for in React?

Refs are an escape hatch which allow you to get direct access to a DOM element or an instance of a component. In order to use them you add a ref attribute to your component whose value is a callback function which will receive the underlying DOM element or the mounted instance of the component as its first argument.

64. What does it mean for a component to be mounted in React?

It has a corresponding element created in the DOM and is connected to that.

65. What is the purpose of using super constructor with props argument?

A child class constructor cannot make use of this reference until super() method has been called. The same applies for ES6 sub-classes as well.
The main reason of passing props parameter to super() call is to access this.props in your child constructors.

66. Does useState Hook update immediately?

React useState and setState don't make changes directly to the state object; they create queues to optimize performance, which is why the changes don't update immediately.
The process to update React state is asynchronous for performance reasons.
To perform side effects after state has change, you must use the useEffect

67. What are React Hooks?

Hooks are a new addition in React 16.8. They let you use state and other React features without writing a class. With Hooks, you can extract stateful logic from a component so it can be tested independently and reused. Hooks allow you to reuse stateful logic without changing your component hierarchy. This makes it easy to share Hooks among many components or with the community.

68. What are advantages of using React Hooks?

Primarily, hooks in general enable the extraction and reuse of stateful logic that is common across multiple components without the burden of higher order components or render props.
Hooks allow to easily manipulate the state of our functional component without needing to convert them into class components.

Hooks don't work inside classes (because they let you use React without classes).
By using them, we can totally avoid using lifecycle methods, such as componentDidMount, componentDidUpdate, componentWillUnmount. Instead, we will use built-in hooks like useEffect .

69. What do these three dots(...) in React do?

That's property spread notation. It was added in ES2018 (spread for arrays/iterables was earlier, ES2015).
Spread notation is handy not only for that use case, but for creating a new object with most (or all) of the properties of an existing object — which comes up a lot when you're updating state, since you can't modify state directly

70. What is prop drilling and how can you avoid it?

When building a React application, there is often the need for a deeply nested component to use data provided by another component that is much higher in the hierarchy.
The simplest approach is to simply pass a prop from each component to the next in the hierarchy from the source component to the deeply nested component.
This is called prop drilling.

The primary disadvantage of prop drilling is that components that should not otherwise be aware of the data become unnecessarily complicated and are harder to maintain.

To avoid prop drilling, a common approach is to use React context.
This allows a Provider component that supplies data to be defined,
and allows nested components to consume context data via either a Consumer component or a useContext hook.

71. What is StrictMode in React?

React's StrictMode is sort of a helper component that will help you write better react components, you can wrap a set of components with <StrictMode /> and it'll basically:

Verify that the components inside are following some of the recommended practices and warn you if not in the console.
Verify the deprecated methods are not being used, and if they're used strict mode will warn you in the console.
Help you prevent some side effects by identifying potential risks.

72. What is the difference betwen ShadowDOM and VirtualDOM?

Virtual DOM
Virtual DOM is about avoiding unnecessary changes to the DOM, which are expensive performance-wise, because changes to the DOM usually cause re-rendering of the page.
Virtual DOM also allows to collect several changes to be applied at once,
so not every single change causes a re-render, but instead re-rendering only happens once after a set of changes was applied to the DOM.
Shadow DOM
Shadow dom is mostly about encapsulation of the implementation.
A single custom element can implement more-or-less complex logic combined with more-or-less complex DOM. An entire web application of arbitrary complexity can be added to a page by an import and <body><my-app></my-app> but also simpler reusable and composable components can be implemented as custom elements where the internal representation is hidden in the shadow DOM like <date-picker></date-picker>

73. Why do class methods need to be bound to a class instance?

In JavaScript, the value of this changes depending on the current context.
Within React class component methods, developers normally expect this to refer to the current instance of a component, so it is necessary to bind these methods to the instance.
Normally this is done in the constructor

74. Describe Flux vs MVC?

Traditional MVC patterns have worked well for separating the concerns of data (Model), UI (View) and logic (Controller) — but MVC architectures frequently encounter two main problems:

1) Poorly defined data flow: The cascading updates which occur across views often lead to a tangled web of events which is difficult to debug.

2) Lack of data integrity: Model data can be mutated from anywhere, yielding unpredictable results across the UI.

With the Flux pattern complex UIs no longer suffer from cascading updates;

any given React component will be able to reconstruct its state based on the data provided by the store. The Flux pattern also enforces data integrity by restricting direct access to the shared data.

75. Describe how events are handled in React

In order to solve cross browser compatibility issues, your event handlers in React will be passed instances of SyntheticEvent, which is React's cross-browser wrapper around the browser's native event. These synthetic events have the same interface as native events you're used to, except they work identically across all browsers.

What's mildly interesting is that React doesn't actually attach events to the child nodes themselves. React will listen to all events at the top level using a single event listener. This is good for performance and it also means that React doesn't need to worry about keeping track of event listeners when updating the DOM.

76. Do hooks replace render props and higher-order components?

Often, render props and higher-order components render only a single child. React team thinks Hooks are a simpler way to serve this use case.

There is still a place for both patterns (for example, a virtual scroller component might have a renderItem prop, or a visual container component might have its own DOM structure). But in most cases, Hooks will be sufficient and can help reduce nesting in your tree.

77. How would you go about investigating slow React application rendering?

One of the most common issues in React applications is when components re-render unnecessarily. There are two tools provided by React that are helpful in these situations:

React.memo(): This prevents unnecessary re-rendering of function components
PureComponent: This prevents unnecessary re-rendering of class components
Both of these tools rely on a shallow comparison of the props passed into the component—if the props have not changed, then the component will not re-render. While both tools are very useful, the shallow comparison brings with it an additional performance penalty, so both can have a negative performance impact if used incorrectly. By using the React Profiler, performance can be measured before and after using these tools to ensure that performance is actually improved by making a given change.

78. React Hooks: Explain why and when would you use useMemo()?

Why:

In the lifecycle of a component, React re-renders the component when an update is made. When React checks for any changes in a component, it may detect an unintended or unexpected change due to how JavaScript handles equality and shallow comparisons. This change in the React application will cause it to re-render unnecessarily.

Additionally, if that re-rendering is an expensive operation, like a long for loop, it can hurt performance. Expensive operations can be costly in either time, memory, or processing.

When:

Optimal if the wrapped function is large and expensive.

How:

Memoization is an optimization technique which passes a complex function to be memoized. In memoization, the result is "remembered" when the same parameters are passed-in subsequently.

useMemo takes in a function and an array of dependencies.
The dependency's list are the elements useMemo watches: if there are no changes,
the function result will stay the same. Otherwise, it will re-run the function.
If they don't change, it doesn't matter if our entire component re-renders,
the function won't re-run but instead return the stored result.

79. React Hooks: What's the difference between useCallback and useMemo in practice?

With useCallback you memoize functions, useMemo memoizes any computed value

80. What is the difference between using constructor vs getInitialState in React?

The difference between constructor and getInitialState is the difference between
ES6 and ES5 itself. You should initialize state in the constructor when using ES6 classes,
and define the getInitialState method when using React.createClass

81. When is it important to pass props to super(), and why?

The only one reason when one needs to pass props to super() is when you want to
access this.props in constructor:
Note that passing or not passing props to super has no effect on later uses of
this.props outside constructor.

82. When whould you use StrictMode component in React?

I've found it especially useful to implement strict mode when I'm working on new code bases
and I want to see what kind of code/components I'm facing.
Also if you're on bug hunting mode, sometimes it's a good idea to wrap with <StrictMode />
the components/blocks of code you think might be the source of the problem.

83. Why would you need to bind event handlers to this?

Binding is not something that is specifc to React, but rather how this works in Javascript.
When you define a component using an ES6 class, a common pattern is for an event handler
to be a method on the class. In JavaScript, class methods are not bound by default.
If you forget to bind this.someEventHandler and pass it to onChange, this will be undefined
when the function is actually called.
Generally, if you refer to a method without () after it, such as
onChange={this.someEventHandler}, you should bind that method.

84. What is Jest?

Jest is a JavaScript unit testing framework created by Facebook based on Jasmine.
It offers automated mock creation and a jsdom environment.
It is also used as a testing component.

85. What is dispatcher?

A dispatcher is a central hub of app where you will receive actions and broadcast
payload to registered callbacks.

86. What is meant by callback function? What is its purpose?

A callback function should be called when setState has finished, and the component is re-rendered.

As the setState is asynchronous, which is why it takes in a second callback function.

87. Explain the Presentational segment

A presentational part is a segment which allows you to renders HTML.
The segment's capacity is presentational in markup.

88. Explain yield catchphrase in JavaScript

The yield catchphrase is utilized to delay and resume a generator work,
which is known as yield catchphrase.

89. Pure components in React js
Pure components are the fastest components which can replace any component with only a render().
It helps you to enhance the simplicity of the code and performance of the application.

90. What are children prop?
Children props are used to pass component to other components as properties.
You can access it by using {props.children}

91. Explain error boundaries?
Error boundaries help you to catch Javascript error anywhere in the child components.
They are most used to log the error and show a fallback UI.

92. What is the use of empty tags <></>?
Empty tags are used in React for declaring fragments.

93. What are reacted portals?
Portal allows you to render children into a DOM node. CreatePortalmethod is used for it.

94. What is Context?
React context helps you to pass data using the tree of react components.
It helps you to share data globally between various react components.

95. What is the use of Webpack?
Webpack is basically is a module builder. It is mainly runs during the development process.

96. What is Babel in React js?
Babel, is a JavaScript compiler that converts latest JavaScript like ES6, ES7 into plain
old ES5 JavaScript that most browsers understand.

97. When should you use the top-class elements for the function element?
If your element does a stage or lifetime cycle, we should use top-class elements.

98. How can you share an element in the parsing?
Using the State, we can share the data.

99. Explain the term reconciliation
When a component's state or props change then rest will compare the rendered
element with previously rendered DOM and will update the actual DOM if it is needed.
This process is known as reconciliation.

100. How can you re-render a component without using setState() function?
You can use forceUpdate() function for re-rending any component.

101. Explain the term 'Restructuring'
Restructuring is extraction process of array objects. Once the process is completed,
you can separate each object in a separate variable.

102. Explain the Mounting and Demounting
The process of attaching the element to the DCOM is called mounting.
The process of detaching the element from the DCOM is called the demounting process.

103. What is the use of 'prop-types' library?
'Prop-types' library allows you to perform runtime type checking for props and
similar object in a recent application.

104. List down some of the methods in a react-dom package

Important methods for react-dom packages are:
render()
hydrate()
createPortal()
unmountComponentAtNode()
findDOMNode()