Name: Sushant G. Hire
Date: 13th February 2023

Q1. What is React JS and why do we use it?

Answer - React is a JavaScript library used for building user interfaces (UI) in web applications. Here are a few reasons why React is popular among developers:

- ★ Components-based: React allows developers to build UI using small, reusable components. This makes it easier to maintain and improve the code.
- ★ Virtual DOM: React uses a virtual DOM (Document Object Model) that provides a faster and more efficient way to update the UI compared to the traditional DOM.
- ★ Performance: React updates only the specific components that need to change, rather than reloading the entire page, which leads to better performance.
- ★ Declarative programming: React uses a declarative programming style, which makes it easier to understand and debug the code.
- ★ Popularity: React is one of the most popular front-end libraries and has a large community of developers who contribute to its development and offer support.

In simple words, React makes it easier and faster to build dynamic and interactive web applications.

Q2. What do you understand by JSX?

Answer - JSX stands for JavaScript XML. JSX allows us to write HTML in React. JSX makes it easier to write and add HTML in React.

- ★ Browsers cannot understand JSX.
- ★ Browsers can only read JavaScript objects but JSX is not a regular JavaScript object.
- ★ Thus to enable a browser to read JSX, first, we need to transform the JSX file into a JavaScript object using JSX transformers like Babel and then pass it to the browser.

Q3. What are functional components?

Answer - Functional Components are a simpler way of defining components in React. They are defined as JavaScript functions that return React elements. Functional components have the following characteristics:

- ★ No state: Unlike class components, functional components do not have a state object and cannot manage their own state.
- ★ Props as arguments: Functional components receive props as an argument and can use them to render any dynamic content.
- ★ Simplified syntax: Because functional components do not have a render method or lifecycle methods, their syntax is much simpler than class components.
- ★ Performance optimization: Functional components can be optimized more easily by React, as they are closer to the React elements they return.

Q4. What are states and props?

Answer - States and props are two important concepts in React.

- ★ Props: Props stands for "properties" and they are used to pass data from a parent component to its child components. Props are read-only, meaning that the child component cannot change the values of its props.
- ★ States: States, on the other hand, are used to store and manage dynamic data within a component. States can change over time and can be used to trigger re-renders of the UI when the state changes. Unlike props, states are mutable and can be changed within a component.

In simple terms, props are like inputs to a component and states are like variables that store data and can change over time. Both props and states are used to control the behavior and render the UI of a React component.

Q5. What do you understand by component-based architecture?

Answer - Component-based architecture is a design pattern used in React to build UI. In this pattern, a UI is divided into smaller, independent, and reusable components.

- ★ Each component has its own logic and render function, which returns a piece of the UI.

- ★ This approach makes it easier to manage and maintain complex UI, as well as makes it possible to reuse components across different parts of the application.

In simple terms, component-based architecture in React is a way of breaking down a UI into smaller, independent pieces called components. Each component has its own logic and render function and can be reused in multiple parts of the application. This makes it easier to manage and maintain the UI.

Q6. What do you understand by rendering and conditional rendering of elements/components?

Answer - Rendering and conditional rendering are two important concepts in React.

- ★ Rendering: Rendering refers to the process of creating a UI from the components and data. In React, this is done using the render function, which returns a piece of UI based on the component's state and props.

- ★ Conditional Rendering: Conditional rendering is a technique in React where the component only renders certain parts of the UI based on specific conditions. For example, a component may only render a button if a certain condition is met, or render a different UI based on the value of a prop. This is done using JavaScript control structures like if/else statements or conditional operators (ternary operators).

In simple terms, rendering is the process of creating a UI in React, and conditional rendering is a technique that allows a component to only render certain parts of the UI based on specific conditions. This makes it possible to show or hide different parts of the UI based on the component's state or props.

Q7. Explain handling events.

Answer - Handling events in React refers to the process of responding to user interactions with the UI, such as clicks, hover, etc. In React, events are handled using event handlers, which are functions that are called in response to a specific event.

Here's how to handle events in React in simple terms:

- ★ Declare an event handler: First, you need to declare a function that will be called in response to an event. This function will contain the logic to handle the event.

- ★ Attach the event handler to an element: Next, you need to attach the event handler to the element that will trigger the event. This is done using the onEvent syntax, where Event is the name of the event you want to handle (e.g. onClick, onHover, etc.).

- ★ Pass the event handler as a prop: Finally, you need to pass the event handler as a prop to the component that will trigger the event. This is done by including the event handler as an attribute in the component's JSX.

In summary, handling events in React involves declaring event handler functions, attaching them to elements in the UI, and passing them as props to the components that will trigger the events.

Q8. Explain Prop Drilling.

Answer - Prop drilling in ReactJS refers to the process of passing data from a parent component to a child component that is several levels down the component tree.

Prop drilling is when you need to pass information from a parent component to a child component that is several levels away, and you need to go through multiple components to do so.

Code File - https://stackblitz.com/edit/react-1yu86h?file=src/index.js

Q9. Explain controlled and uncontrolled components.

Answer - These are the 2 types of components:

- ★ In a controlled component, the value is set and updated through the React state. This means that the form is controlled by the React code and the user's inputs will update the state. Controlled components provide more control and structure for forms in React. It takes its current value through props and makes changes through callback functions like onClick() function.

- ★ In an uncontrolled component, the value is stored and updated directly by the DOM, without using the React state. This means that the form is controlled by the DOM and the user's inputs will update the value directly, bypassing the React code. Uncontrolled components are simpler and quicker to implement for basic use cases.

Q10. Explain different types of CSS.

Answer - There are three main types of CSS styles: inline, internal, and external (also known as "module level").

- ★ Inline CSS: This type of CSS is written directly in the HTML element using the "style" attribute. It is the most specific form of styling and overrides any other styles that may be defined elsewhere.

- ★ Internal CSS: This type of CSS is defined within the "head" section of an HTML document, using the "style" tag. All elements on the page will be affected by the styles defined in the internal CSS.

- ★ External CSS: This type of CSS is defined in a separate file with a ".css" extension, and is linked to the HTML document using the "link" tag in the "head" section. This allows the same styles to be reused across multiple pages, making it easier to maintain and update the styling of a website.

In simple words, the difference between the three types of CSS is where they are located and how they affect the styling of a web page. Inline CSS is the most specific, internal CSS affects the whole page, and external CSS can be reused across multiple pages.

Q11. Explain the lists and keys.

Answer - Lists and keys are two important concepts in React when working with dynamic data.

- ★ Lists: Lists are used to render multiple items in a UI. For example, you may have a list of items in an e-commerce app that you want to render in a table or as a list of cards. In React, you can use the map function to loop over an array of data and render each item as a component.

- ★ Keys: Keys are unique identifiers that are assigned to each item in a list. They are used by React to keep track of which items have changed, added, or removed. This helps React optimize the rendering process and update the UI efficiently.

In simple terms, lists are used to render multiple items in a UI, and keys are unique identifiers that help React keep track of the items in a list and update the UI efficiently.

Q12. Explain local storage and browser storage.

Answer - Local storage and browser storage are two options for storing data in a web browser. Here are the main differences between them:

- ★ Persistence: Local storage is persistent, meaning that the data stored in local storage will remain even after the browser is closed, while browser storage (e.g. session storage) is not persistent and the data is lost once the browser is closed.

- ★ Capacity: Local storage has a larger capacity compared to browser storage, with most browsers allowing up to 5 MB of data storage in local storage.

- ★ Access: Both local storage and browser storage can be accessed from JavaScript within a web page, but browser storage is only accessible within the same window or tab that created it.

- ★ Data Types: Both local storage and browser storage can store strings, but local storage can also store complex data types such as arrays and objects using JSON.stringify() and JSON.parse().

In simple terms, local storage is a way of storing data in a web browser that is persistent and has a larger capacity, while browser storage is a way of storing data that is not persistent and has a smaller capacity. Both can be accessed from JavaScript within a web page, but local storage can store more complex data types.

Q13. Explain Recoil.

Answer - Recoil is a state management library for React that allows us to manage and optimize the state of our components in a very efficient manner. Basically, it provides us with several methods to store the shared state that multiple components can use without passing data through props. Its core concepts include -

- ★ Atoms - Atoms are the basic units of the state. They are updatable and subscribable and represent a single piece of data that can be read and updated by the developer.

- ★ Selectors - Selectors are a way to derive a new state from an existing state. They can be used to combine or transform multiple Atoms into a single derived state.

- ★ The useRecoilState hook - The useRecoilState hook is basically a hook that allows us to read and update a Recoil state. It returns us an array with the current value of the atom and a setter function in order to update it.

- ★ The useSetRecoilState hook - The useSetRecoilState hook is a hook that provides a setter function in order to update the value of a Recoil state that we want to update. It takes the Recoil state as an argument and returns us the setter function, which is then used to update the value of that state. It is helpful when we need to update the state outside of the component that uses that particular state.

- ★ The useRecoilValue hook - The useRecoilValue hook is a hook that allows us to read a Recoil state. It returns the current value of the Atom.

- ★ RecoilRoot - RecoilRoot is a component that wraps up our entire React application and serves as the root of the Recoil state tree.

In conclusion, Recoil is a very powerful state management library for React that provides us with several different methods to manage the state of your application in a centralized and efficient way. Its core concepts include atoms, selectors, and hooks such as useRecoilState, useRecoilValue, useSetRecoilState, etc., allowing us to read, update, and observe changes to the state. Recoil offers a variety of other properties and functions to handle more advanced use cases.

Q14. Explain routing with react-router-dom.

Answer - Routing in React refers to the process of mapping URLs to components, which are then displayed in the UI. React Router is a popular library for implementing routing in React applications.

- ★ Installing React Router: You can install React Router as a dependency in your React project using npm or yarn.

- ★ Defining Routes: To define routes in your React application, you use the Route component provided by React Router. Each Route component maps a specific URL path to a React component that should be displayed when that URL is visited.

- ★ Rendering Routes: The routes are rendered using the BrowserRouter component from the react-router-dom library. This component acts as a wrapper for your React application and provides the functionality for managing the URL and routing.

- ★ Navigating between routes: You can navigate between routes in your React application using the Link component provided by React Router. The Link component allows you to define links that, when clicked, will change the URL and trigger the appropriate component to be displayed.

In simple terms, routing in React allows you to map URLs to components, which are then displayed in the UI. React Router is a popular library for implementing routing in React applications, and it provides components for defining routes, rendering routes, and navigating between routes.

Q15. Explain children props.

Answer - The children prop in React refers to the content that is contained within a component's opening and closing tags. In other words, it represents the elements that are nested inside a component.

- ★ A component can access its children prop via the props object, just like any other prop.
- ★ In the component's render method, you can use the props.children to render the content that was passed as the children prop.
- ★ Using the children prop can make your components more flexible and reusable.
- ★ For example, you can create a component that takes in some content as the children prop and renders it in a specific way, such as with a border or in a specific layout.
- ★ This way, you can use the same component multiple times with different content, without having to repeat the same code.

Q16. How to deal with npm packages?

Answer - NPM (Node Package Manager) is a package manager for JavaScript that is widely used in the React community to manage dependencies and packages. Here's how to deal with NPM packages in React:

- ★ Installing packages: To install an NPM package in your React project, you can use the npm install command followed by the package name. For example, to install the lodash package, you can run npm install lodash.
- ★ Importing packages: Once you have installed an NPM package, you can import it into your React components using the import statement. For example, to import the lodash package, you can add the following line to your React component: import _ from 'lodash';.

- ★ Updating packages: To update an NPM package in your React project, you can use the npm update command followed by the package name. For example, to update the lodash package, you can run npm update lodash.

- ★ Removing packages: To remove an NPM package from your React project, you can use the npm uninstall command followed by the package name. For example, to remove the lodash package, you can run npm uninstall lodash.

In simple terms, NPM is a package manager for JavaScript that allows you to install, import, update, and remove packages in your React projects. You can use the npm command line tool to manage NPM packages in your React projects.

Q17. Explain the Fetch method to fetch an API.

Answer - The fetch method in React allows you to retrieve data from a remote source, like an API, and use it in your React component.

- ★ The fetch method is used to retrieve data from a remote source.

- ★ The data is then stored in the state of your React component.

- ★ The fetch method makes an API request to a specified URL.

- ★ The response data from the API is returned and can be manipulated and displayed in your React component.

- ★ Fetch is a convenient way to get data from an API and use it in your React application.

Q18. Explain the Material UI library.

Answer - Material UI is a popular React library that provides pre-designed UI components and styles based on Google's Material Design guidelines. It makes it easier to develop a consistent and visually appealing user interface for your React applications.

- ★ Components: Material UI provides a large collection of pre-designed UI components such as buttons, icons, forms, and more, that you can use in your React applications.

- ★ Styles: Material UI also provides a set of pre-designed styles and themes based on the Material Design guidelines, so you don't have to design and style your components from scratch.

- ★ Customization: Material UI components can be customized to match your specific requirements and design needs.

- ★ Interoperability: Material UI components are designed to work well with other popular React libraries, making it easier to integrate with your existing React projects.

In simple terms, Material UI is a React library that provides pre-designed UI components and styles based on the Material Design guidelines. It makes it easier to develop a consistent and visually appealing user interface for your React applications.

Q19. Explain the import and export of components and data.

Answer - The import and export are used in React to include code from one file into another file. They are part of the ES6 (ECMAScript 6) module system, which provides a way to split your code into reusable and modular units.

- ★ The import statement: The import statement is used to include code from another file.

- ★ The export statement: The export statement is used to make values and functions from a file available for import in another file.

- ★ Default export: A file can also have a default export, which is the value that is exported when no name is specified.

A file can have both named exports and a default export, but it can only have one default export.

Q20. Explain media queries in CSS.

Answer - Media Queries in CSS allow you to apply different styles to your website based on the dimensions of the device or the size of the screen. They are used to make your website responsive so that it looks good on different devices and screen sizes.

- ★ Max-width: The max-width media query property allows you to specify the maximum width at which a specific style should be applied. For example, you can use the max-width property to apply a different style to your website when the screen size is below a certain width.

- ★ Min-width: The min-width media query property allows you to specify the minimum width at which a specific style should be applied. For example, you can use the min-width

property to apply a different style to your website when the screen size is above a certain width.

When to use them:

- ★ You can use max-width and min-width media queries when you want to apply different styles based on the width of the screen.
- ★ You can use max-width to apply styles for small screen sizes, such as mobile devices, and min-width to apply styles for larger screen sizes, such as desktops.

In simple terms, media queries in CSS allow you to apply different styles to your website based on the size of the screen. The max-width and min-width properties allow you to specify the maximum and minimum width at which a specific style should be applied, respectively. They are used to make your website responsive and to apply different styles for different screen sizes.

Q21. Explain how to use GitHub to manage their code.

Answer - GitHub is a web-based platform that allows you to store and manage your code online. It provides version control, collaboration features, and a user-friendly interface for managing your code.

Here are the basic steps for using GitHub to manage your code:

Create a GitHub account: Sign up for a free GitHub account on the GitHub website.

- ★ Create a repository: A repository is a place to store your code and track changes. To create a repository, click the "New repository" button on the GitHub homepage and enter a name and description for your repository.
- ★ Clone the repository: To make a local copy of the repository on your computer, you can use the "Clone" button on the GitHub repository page to copy the repository's URL. Then, you can use Git to clone the repository by running the git clone command in your terminal or command prompt, followed by the URL.
- ★ Make changes: You can now make changes to the code in the local copy of the repository on your computer. You can add, modify, and delete files as needed.
- ★ Commit changes: When you're ready to save your changes, you can use Git to commit the changes. A commit is like a snapshot of your code at a specific point in time. To commit changes, you can use the git commit command in your terminal or command prompt, followed by a message describing the changes.

- ★ Push changes: After committing changes, you can push the changes to the remote repository on GitHub by using the git push command in your terminal or command prompt. This will upload your changes to GitHub, making them accessible to others.

- ★ Collaborate: If you're working on a project with others, you can use GitHub to collaborate on the code. You can use the pull request feature to suggest changes to the code, and others can review and approve or reject the changes. You can also use GitHub issues to track bugs and feature requests.

These are the basic steps for using GitHub to manage your code. The platform has many other features, but these steps should get you started.

Q22. Explain DOM and Virtual DOM.

Answer - The following is a brief information regarding DOM and Virtual DOM:

DOM (Document Object Model): The DOM is a tree-like representation of the structure of a web page, including all its elements, attributes, and content. When a web page is loaded, the browser creates a DOM and uses it to render the page on the screen.

Virtual DOM: The Virtual DOM is a virtual representation of the DOM used by React to update the actual DOM efficiently. When a change is made to the state of a React component, React first updates the Virtual DOM, which is a lightweight and fast in-memory representation of the actual DOM.

- ★ Speed: The Virtual DOM is faster than the actual DOM because it minimizes the number of changes made to the actual DOM, reducing the time it takes to update the page.

- ★ Abstraction: The Virtual DOM provides an abstraction layer between the React components and the actual DOM, making it easier to work with and update the page.

- ★ Update efficiency: The Virtual DOM allows React to update the actual DOM more efficiently by only updating the parts of the page that have actually changed, instead of redrawing the entire page.

In simple terms, the DOM is a representation of a web page used by the browser to render the page on the screen, while the Virtual DOM is a virtual representation of the DOM used by React to update the actual DOM efficiently. The Virtual DOM is faster, provides an abstraction layer, and allows React to update the actual DOM more efficiently.

Q23. Explain some important higher-order functions of JavaScript.

Answer - Higher-order functions are functions that can either accept another function as an argument or return a function as output. These functions are a powerful tool in JavaScript, as they allow you to write concise, reusable, and abstract code.

Here are some of the most important higher-order functions in JavaScript, along with examples and explanations:

- ★ map: This function allows you to apply a function to each element in an array and return a new array with the results. For example:
  ```
  const numbers = [1, 2, 3, 4, 5];
  const doubledNumbers = numbers.map(number => number * 2);
  console.log(doubledNumbers); // [2, 4, 6, 8, 10]
  ```

- ★ filter: This function allows you to filter an array based on a condition and return a new array with the elements that meet the condition. For example:
  ```
  const numbers = [1, 2, 3, 4, 5];
  const evenNumbers = numbers.filter(number => number % 2 === 0);
  console.log(evenNumbers); // [2, 4]
  ```

- ★ reduce: This function allows you to reduce an array to a single value by applying a function to each element and accumulating the result. For example:
  ```
  const numbers = [1, 2, 3, 4, 5];
  const sum = numbers.reduce((acc, number) => acc + number, 0);
  console.log(sum); // 15
  ```

- ★ sort: This function allows you to sort an array in ascending or descending order. For example:
  ```
  const numbers = [5, 3, 2, 4, 1];
  const sortedNumbers = numbers.sort((a, b) => a - b);
  console.log(sortedNumbers); // [1, 2, 3, 4, 5]
  ```

- ★ forEach: This function allows you to iterate over an array and perform an action for each element. For example:
  ```
  const numbers = [1, 2, 3, 4, 5];
  numbers.forEach(number => console.log(number));
  ```

These are just a few examples of higher-order functions in JavaScript. Understanding and using these functions can greatly improve the quality and readability of your code.

Q24. Explain the async and await.

Answer - async/await is a way to handle asynchronous code in JavaScript, including in React.

- ★ Asynchronous code: Asynchronous code refers to code that runs independently of the main program and does not block the execution of other code. For example, fetching data from a server or reading a file are asynchronous operations.

- ★ async keyword: The async keyword is used to declare a function as asynchronous. When an async function is called, it returns a promise that is resolved when the function's code is complete.

- ★ await keyword: The await keyword is used inside an async function to wait for a promise to be resolved before continuing with the rest of the code.

For example, if you have an asynchronous function that fetches data from a server, you can use the await keyword to wait for the data to be returned before updating the state of your React component.

In simple terms, async/await is a way to handle asynchronous code in JavaScript, including in React. The async keyword is used to declare a function as asynchronous, and the await keyword is used to wait for a promise to be resolved before continuing with the rest of the code.

Code File - https://stackblitz.com/edit/react-sf1dvi?file=src/App.js

Q25. Explain useState, useEffect, useRef, and useMemo.

Answer - React provides several hooks that you can use in your functional components to manage state, perform side effects, and other tasks. Here's a brief explanation of each hook:

- ★ useState - useState is a hook that allows you to add state to your functional components. State is a way of managing data that can change over time. The useState hook returns an array with two elements, the first being the current value of the state, and the second being a function to update the state.

  Code File - https://stackblitz.com/edit/react-we5hc5?file=src/App.js

- ★ useEffect - useEffect is a hook that allows you to perform side effects in your functional components. Side effects can include things like making an API request, setting up a subscription, or updating the title of a page. The useEffect hook takes two arguments: a

function that contains the side effect, and a dependency array that tells React when to re-run the effect.

Code File - https://stackblitz.com/edit/react-zcqm16?file=src/App.js

★ useRef - useRef is a hook that allows you to store a mutable value in your component that will persist across re-renders. The useRef hook returns a ref object that you can use to access the value.

Code File - https://stackblitz.com/edit/react-xujh9c?file=src/App.js

★ useMemo - useMemo is a hook that allows you to memoize a value. Memoization is a way of caching the result of a computation so that it only needs to be re-computed when the inputs to the computation change. The useMemo hook takes two arguments: a function that computes the value, and a dependency array that tells React when to re-compute the value.

Code File - https://stackblitz.com/edit/react-ivndix?file=src/App.js

Q26. Explain different types of positions in CSS.

Answer - There are 5 main types of positions. Remember the acronym S.F.A.R.S. (sounds like "*as far as..*") and begin your answer like "As far as I know of positions, there are main 5 positions in CSS, namely -

- ❖ S - Static
- ❖ F - Fixed
- ❖ A - Absolute
- ❖ R - Relative
- ❖ S - Sticky

★ Static: This is the default position value, and it means that the element will follow the normal flow of the page. No position properties (e.g., top, right, bottom, left) are needed for a static element.

Code File - https://stackblitz.com/edit/react-po16rk?file=src/App.js

★ Relative: A relative positioned element is positioned relative to its normal position. When you set the position to relative, you can then use the top, right, bottom, and left properties to offset the element from its normal position.

Code File - https://stackblitz.com/edit/react-ct4nbx?file=src/App.js

★ Absolute: An absolute positioned element is positioned relative to the nearest positioned ancestor, but if there is no positioned ancestor, it will be positioned relative to the initial containing block, which is usually the viewport or the body element.

Code File - https://stackblitz.com/edit/react-6efvcf?file=src/App.js

★ Fixed: A fixed-positioned element is positioned relative to the viewport, and it will not move even if the page is scrolled.

Code File - https://stackblitz.com/edit/react-ispqt4?file=src/App.js

★ Sticky: A sticky positioned element is a hybrid of relative and fixed positioning. It is positioned relative to its normal position until a certain point (defined by top, right, bottom, or left) is reached, then it becomes fixed.

Code File - https://stackblitz.com/edit/react-bon5q7?file=src/App.js