- **ReactJS is a JavaScript library.**
- **ReactJS is used to build web applications (user interfaces).**
- **ReactJS was released by Facebook.** *(Now maintained by Meta.)*
- **The current version is React 19.x.** ✔

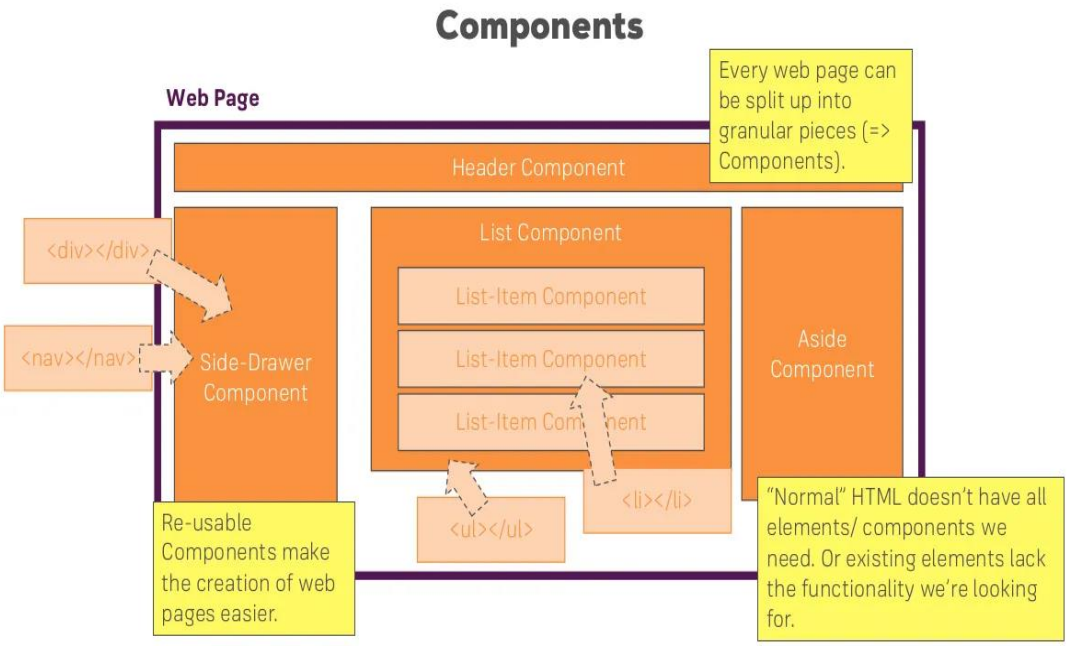
ReactJS – Features

1. Components in React

- A **component** is a small, reusable piece of UI in a React application.
- Each part of a webpage can be built as a **separate component**.
- Components make the application **modular, maintainable, and reusable**.
- One component can be included inside another (**component composition**).
- React provides **two types of components**:
  - **Functional Components**
  - **Class Components**

Differences Between Class Components and Functional Components

Class Components	Functional Components
1. Created using **ES6 classes**.	1. Created using **JavaScript functions**.
2. Use **lifecycle methods** such as componentDidMount, componentDidUpdate, etc.	2. Use **Hooks** like useEffect to handle lifecycle events.
3. Manage state using this.state.	3. Manage state using **useState** and other Hooks.
4. Require **this keyword** to access props and state.	4. Do **not** use the this keyword.
5. Slightly **slower and heavier** because they involve class instantiation.	5. **Faster and lightweight** due to simple function structure.
6. Older approach; **not preferred** for new React apps.	6. **Recommended** approach to modern React.



2) JSX

- **JSX stands for JavaScript XML.**
- In JSX, we write **HTML inside JavaScript**.
- JSX is used to build **React applications**.
- **Babel** is used to convert JSX into **equivalent JavaScript** that browsers can understand.

Differences Between JSX and TSX

JSX	TSX
1. JSX stands for **JavaScript XML**.	1. TSX stands for **TypeScript XML**.
2. Used to write **HTML inside JavaScript**.	2. Used to write **HTML inside TypeScript**.
3. **No type checking**.	3. **Type checking available** because it uses TypeScript.
4. File extension is **.jsx**.	4. File extension is **.tsx**.
5. Suitable for **small-scale** applications.	5. Suitable for **medium and large-scale** applications.

6. Errors are mostly caught **at runtime**.	6. Errors are caught **during development/compile time**.

Extra Interview Notes

- TSX is preferred in modern enterprise applications because **TypeScript reduces bugs** and improves code quality.
- JSX is simpler and easier for beginners but less safe in large codebases.
- Both JSX and TSX are eventually compiled into **plain JavaScript**.

3) Expression

- An **expression** is used to display **dynamic data** in React.
- Expressions are also called **interpolation**.
- Expressions are written using **curly braces {}** inside JSX.
- We can use:

✓ Variables

✓ Functions

✓ Math operations

✓ Conditional operators

✓ Boolean expressions

Examples

jsx Copy code

```jsx
<h1>{"Welcome"}</h1>
<p>{10 + 10}</p>

{
  10 > 9 ? <h1>{"ReactJS"}</h1> : <h1>{"NodeJS"}</h1>
}
```

4) State

- **State** is used to store and manage **component data**.
- State is **mutable** (it can change).
- In **functional components**, state is created using the **useState() hook**.
- When state changes, **react re-renders** the component automatically.

State in Functional Components (useState Hook)

Basic Example

jsx Copy code

```jsx
const [x, setX] = useState("CareerIT");

{x}          // Output: CareerIT

setX("NodeJS");

{x}          // Output: NodeJS
```

State With Numbers

jsx Copy code

```jsx
const [num1, setNum1] = useState(200);
const [num2, setNum2] = useState(100);

{num1 + num2}    // 300
{num1 - num2}    // 100
```

State With Boolean

```jsx
const [flag, setFlag] = useState(true);

{
  flag ? num1 : num2      // Output: 200
}

{
  !flag ? num1 : num2     // Output: 100
}
```

State With Arrays

```jsx
const [arr, setArr] = useState([10, 20, 30, 40, 50]);

{
  arr.map((element, index) => {
    return <p key={index}>{element} --- {index}</p>
  })
}
```

State With Objects

```jsx
const [obj, setObj] = useState({
  frontend: "ReactJS",
  backend: "NodeJS"
});

{obj.frontend}   // ReactJS
{obj.backend}    // NodeJS
```

State with Array of Objects (Employee Example)

```jsx
const [emps, setEmps] = useState([
  { eno: 111, ename: "Emp1", esal: 10000 },
  { eno: 222, ename: "Emp2", esal: 20000 },
  { eno: 333, ename: "Emp3", esal: 30000 }
]);

<table>
  <tbody>
    {
      emps.map((element, index) => {
        return (
          <tr key={index}>
            <td>{element.eno}</td>
            <td>{element.ename}</td>
            <td>{element.esal}</td>
          </tr>
        );
      })
    }
  </tbody>
</table>
```

Comments

Comments are used to explain code and make it easier to understand.

Each language has its own comment syntax.

1) HTML Comments

```html
<!-- HTML comment -->
```

2) CSS Comments

```css
/*
   CSS comment
*/
```

3) JavaScript Comments

Single-line comment

```javascript
// This is a single-line comment
```

Multi-line comments

```javascript
/*
   This is a multi-line comment
*/
```

4) React Comments

React comments must be written **inside JSX** using curly braces {}.

```jsx
{/* This is a React comment */}
```

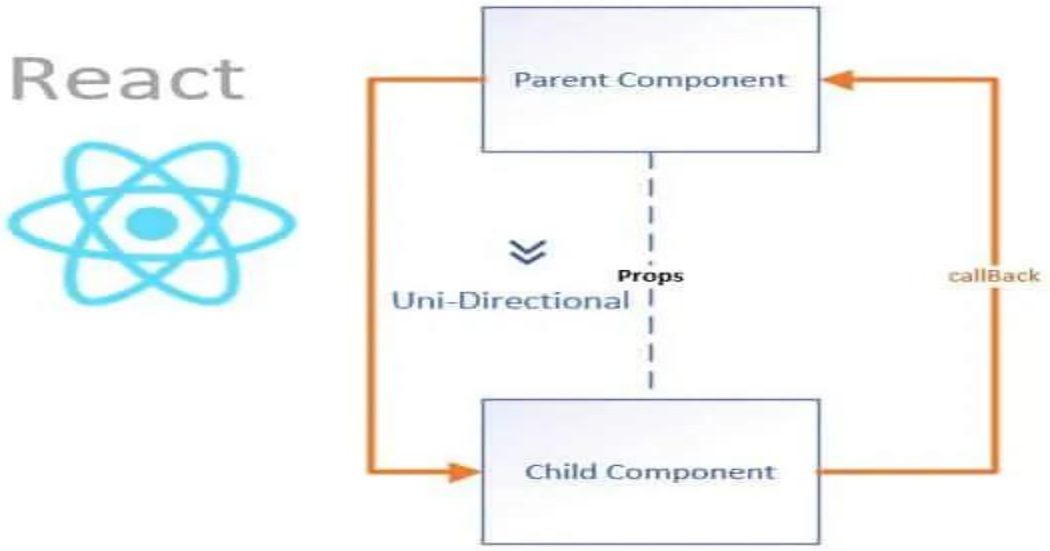
5) Props

- A child's component receives data from a parent's component using **props**.
- Props are **immutable** (cannot be changed by the child component).

Difference Between *State* and *Props*

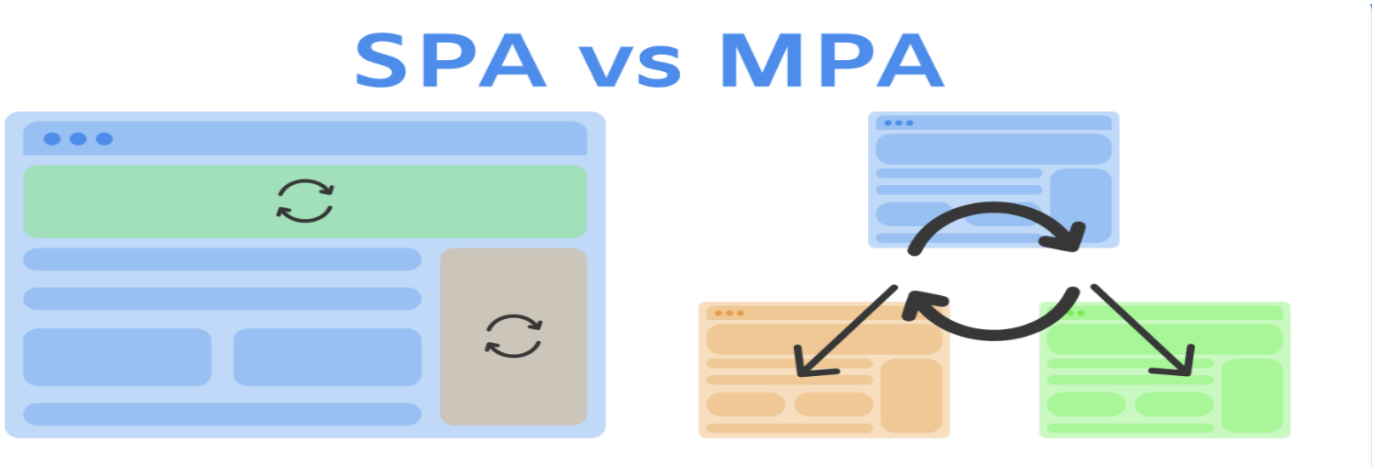
State	Props
State is used to manage data inside a component.	Props are used to pass data from a parent to a child's component.

State is **mutable** (can be changed using setState or useState).	Props are **immutable** (cannot be changed by the child).



6) Single Page Applications (SPA)

- Loading one component inside another **without refreshing the page** is called a Single Page Application.
- Moving from one component/page to another is called **Routing**.
- <Route> is a predefined component used to create a route.
- <Routes> is used to group multiple routes.
- <BrowserRouter> is used for **dynamic routing** (URL looks normal).
- <HashRouter> is used for **static routing** (URL contains # symbol).
- <Link> or <NavLink> components are used to create navigation links.
- useParams() hook is used to read route parameters.
- useNavigate() hook is used to navigate through code (events, buttons, etc.).
- / Is called the **default route**.
- * Is called the **wildcard route** (used for 404 pages).

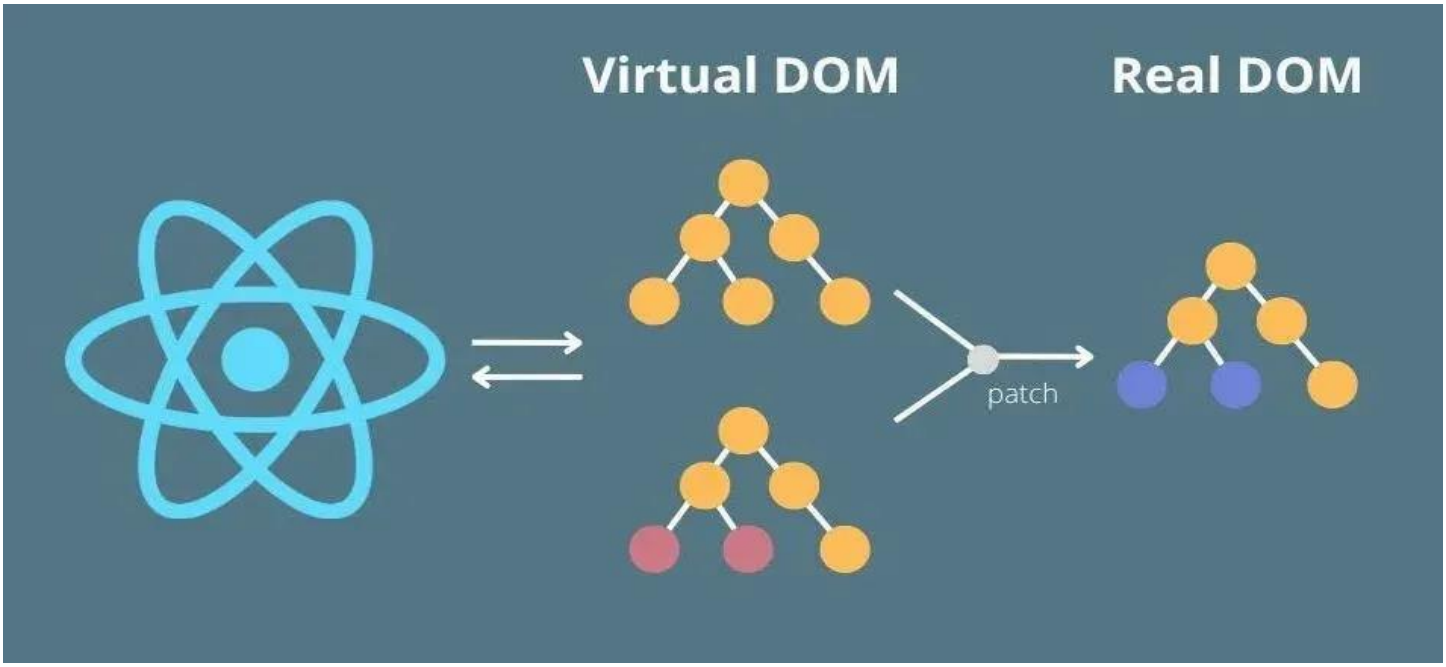


7) Babel

- Babel converts **modern JavaScript** into **older JavaScript**.
- Babel helps the project run in **all browsers**.
- Babel also converts **JSX** to normal JavaScript.
- Babel is called a **JavaScript Compiler / Transpiler**.

8) Virtual DOM

- The Virtual DOM is a copy of the real webpage kept in memory.
- When changes happen, React updates the **Virtual DOM first**, not the real DOM.
- The updated Virtual DOM is compared with the previous Virtual DOM.
- Only the **changed parts** are updated in the real DOM.
- This makes React applications **faster and more efficient**.



9) State Management

- Passing data through many layers of components is called **props drilling**.
- State management is used to **avoid props drilling**.
- Two main state management methods are:
  - **Context API**
  - **Redux**

Difference Between Context API & Redux

Context API	Redux
Provided by **React** (Facebook).	Provided by a **third-party library**.
Good for **small applications**.	Good for **medium and large applications**.
Best for **static or less-changing data**.	Best for **dynamic or frequently changing data**.
Harder to debug.	Easier to debug (DevTools available).
Build size does **not increase much**.	Build size **increases** because Redux adds extra files.

10) Hooks

Hooks add extra features to functional components.

Hooks were introduced in **React 16.8**

Common Hooks

- useState()
- useRef()
- useEffect()
- useParams()
- useNavigate()
- createContext()
- useContext()
- useCallback()
- useMemo()
- use() (new in modern React)

Software Installation

Download and Install Node.js

Website: https://nodejs.org/en/download

What is npm?

- **npm** stands for **Node Package Manager**.
- npm comes installed with **Node.js**.
- npm is used to **download, install, update, uninstall, and manage** JavaScript libraries/packages.

Difference Between npm and yarn

npm	yarn
Given by the **Node.js team**.	Given by **Facebook**.
Used to manage packages/libraries.	Also used to manage packages/libraries.
Slightly **slower** compared to Yarn.	Slightly **faster** compared to npm.
Needs **internet** to install previously downloaded packages.	Can install previously downloaded packages **without internet** (because of caching).

Installing Yarn

> npm install -g npm@11.6.4
> npm install -g yarn
> yarn -v

What is Vite?

- **Vite** is a fast-built tool used to create and run modern web applications.
- With Vite, you can create:
  - Angular applications
  - React applications
  - Vanilla JavaScript applications
  - Next.js applications (via plugins)
  - Many other frameworks
- Vite supports **HMR (Hot Module Replacement)**.

What is HMR?

- When you change code, **only the changed part** updates on the screen.
- The **whole page does not reload**.
- This makes development **faster and smoother**.

Create React Application (Using Vite)

Command:

npm create vite@latest

Questions During Setup:

Q1: Proceed? → Yes (press **y**)

Q2: Project name: → first-application (write in **small letters**)

Q3: Select a framework: → **React**

Q4: Select a variant: → **JavaScript**

Q5: Use rolldown-vite (Experimental)? → No

Q6: Install with npm and start now? → Yes

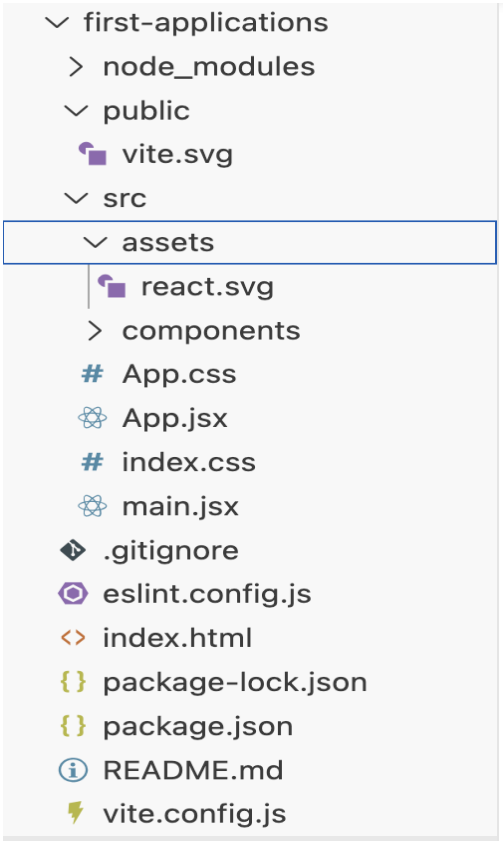
Q7: Press o to open automatically

—or—

Open manually: **http://localhost:5173/**

Note: Default port is **5173**.

Directory Structure



1) node_modules

- Contains all installed **libraries/packages**.
- We **should not rename** the node_modules folder.
- To install packages:

```nginx
npm install
```

or

```css
npm i
```

2) public

- Files inside public are **directly accessible** by the browser.
- vite.svg is the default SVG file in this folder.

3) src

- Main folder used to write **application code**.

4) src/assets

- Stores **static resources**, such as:
  - CSS files
  - HTML files
  - Images
  - Fonts
- react.svg is the default file inside assets.

5) src/components

- Used to store **React components**.
- It is recommended to create a separate folder named components inside src.

6) src/App.jsx

src/App.css

- App.jsx is the **default main component** in the React application.
- App.css contains the **styles** for the App component.

7) src/index.css

- This is the **global style sheet**.
- Styles written here apply to **all components**.

8) src/main.jsx

- This is the **entry point** of the application.
- It renders the React app into:

which is located inside index.html.

9) index.html

- React application starts running from this file.
- Contains a where React loads your entire app.

10) package.json

- Contains details of all **libraries**, **scripts**, and **project information**.

11) package-lock. Json

- Locks the **exact versions** of all installed packages.
- Helps maintain the same versions for all developers.

✓Creating Your First Component in React

Comp1.jsx

```jsx
const Comp1 = () => {
    return (
        <>
            <h1>Welcome to</h1>
        </>
    )
}

export default Comp1;
```

Explanation

1) const Comp1 = () => { ... }

- You are creating a **function component** named **Comp1**.
- Components must start with a **capital letter**.

2) return (...)

- A component must **return JSX**.
- JSX looks like HTML but works inside JavaScript.

3) <>...</> (Fragment)

- <> and </> are called **React Fragments**.
- They allow you to return **multiple elements** without adding an extra .

4) <h1>Welcome </h1>

- This is a simple heading that will appear on the screen.

5) export default Comp1;

- This allows the component to be used in another file.
- Without exporting, you cannot import it elsewhere.

Using the Component in main.jsx

```jsx
import { StrictMode } from 'react'
import { createRoot } from 'react-dom/client'
import './index.css'
import App from './App.jsx'
import Comp1 from './components/Comp1.jsx'

createRoot(document.getElementById('root')).render(
  <StrictMode>
    <Comp1 />
  </StrictMode>,
)
```

Copy code

Explanation

import { StrictMode } from 'react'

- This imports **StrictMode**, a tool that helps find errors in development.
- It does not affect production.
- It is only for safety checks.

import { createRoot } from 'react-dom/client'

- createRoot() tells React where to show your application on the webpage.

import './index.css'

- This imports global CSS styles for the whole project.

import App from './App.jsx'

- This imports the default **App component**.
- In this example, we are not using it, but it is still available.

import Comp1 from './components/Comp1.jsx'

- This imports your **Comp1** component from the components folder.
- Now you can use <Comp1 /> anywhere inside JSX

✓ Rendering the Component

createRoot(document.getElementById('root')).render(...)

- React finds the inside **index.html**.
- React will put your application **inside that div**.

<StrictMode>

- Wraps your component for extra safety checks.
- Only runs in development mode.

<Comp1 />

- This tells React:

"Show the Comp1 component on the screen."

☑ MCQs

MCQ Set — React Basics

1. ReactJS is mainly used for building _____.
2. React was released by which company?
3. The current version of React _____.
4. Components in React make applications _____.
5. Which type of component uses Hooks?
6. Which component type uses lifecycle methods like componentDidMount?

MCQ Set — JSX vs TSX

7. JSX stands for _____.
8. TSX is mainly used in _____ applications.
9. Which one supports type checking: JSX or TSX?
10. Errors in JSX are mostly caught at _____.
11. The file extension for TSX is _____.
12. JSX is converted into JavaScript using _____.

MCQ Set — Expressions

13. Expressions in JSX are written inside _____.
14. Which of the following can be used inside expressions? Variables / Functions / Math / Conditional operators?
15. An expression in React is also called _____.

MCQ Set — State

16. State in React is _____ (mutable or immutable?).
17. In functional components, state is created using _____ hook.
18. Changing state causes React to _____ the component.
19. Which hook is used to store boolean, array, or object values?
20. Which value comes first in useState: state variable or setter function?

MCQ Set — Arrays & Objects in State

21. The map() function requires a _____ when rendering lists in React.
22. When working with arrays in state, which hook is used to update them?
23. React recommends giving each list item a unique _____.

MCQ Set — Comments

24. HTML comments start with _____ and end with _____.
25. CSS comments are written using _____.
26. React comments must be written inside _____ using curly braces.

MCQ Set — Props

27. Props allow which direction of data flow? Parent → Child OR Child → Parent?
28. Props in React are _____ (mutable or immutable?).
29. Props are used to _____ data between components.
30. Which re-renders first: Virtual DOM or Real DOM?

MCQ Set — SPA & Routing

31. Loading components without page refresh is called _____.
32. Moving between components is called _____.
33. is used to _____.

34. is used to _____.
35. is used for _____ routing.
36. uses a _____ symbol in the URL.
37. The wildcard route is represented by _____.
38. The default route is represented by _____.
39. useParams() hook is used to read _____.
40. useNavigate() hook is used for _____.

MCQ Set — Babel

41. Babel converts modern JavaScript into _____ JavaScript.
42. Babel is also called a JavaScript _____.
43. Babel also converts _____ into pure JavaScript.

MCQ Set — Virtual DOM

44. Virtual DOM is stored in _____.
45. React first updates which DOM? Virtual or Real?
46. Virtual DOM compares current and previous DOM using _____ process.
47. React updates only _____ parts of the real DOM.
48. Virtual DOM makes applications _____.

MCQ Set — State Management

49. Passing data through many levels of components is called _____.
50. State management helps avoid _____.
51. Context API is provided by _____.
52. Redux is suitable for _____ scale applications.
53. Context API is best for _____ data.
54. Redux debugging is _____ compared to Context API.

MCQ Set — Hooks

55. Hooks were introduced in React version _____.
56. Which hook is used to manage component state?
57. Which hook is used to reference DOM elements?
58. Which hook handles side effects?
59. Which hook is used to create global context?

60. Which hook is used to read values from context?
61. Which hook is used to memoize functions?
62. Which hook is used to memoize values?
63. Which hook is newly added to modern React?

MCQ Set — npm & Yarn

64. npm stands for _____.
65. npm comes pre-installed with _____.
66. Yarn was developed by _____.
67. Yarn is slightly _____ than npm.
68. npm requires _____ to install previously downloaded packages.

MCQ Set — Vite

69. Vite is a tool used for _____ and _____ applications.
70. Vite supports which advanced features?
71. Using Vite, you can create _____ (React / Angular / VanillaJS / NextJS?).
72. HMR stands for _____.
73. HMR updates only the _____ part of the screen.
74. Vite uses which default port?

MCQ Set — Directory Structure

75. The folder that contains all packages is _____.
76. The folder accessible directly by the browser is _____.
77. Static resources like images and fonts go inside _____.
78. Components are stored inside _____ folder.
79. The default main component is _____.
80. Global styles are written inside _____.
81. The entry point file is _____.
82. The HTML file that contains root div is _____.
83. The file that stores project dependencies is _____.
84. The file that locks exact library versions is _____.

MCQ Set — First Component

85. React components must return _____.

86. Fragments are written as _____.
87. Exporting a component is done using _____.
88. createRoot() is used to render components into _____.
89. StrictMode is used for _____ checking during development.
90. Components must start with a _____ letter.

☑ Fill in the Blanks

91. JSX stands for _____.
92. TSX is used inside _____.
93. State in React is _____.
94. Props are _____.
95. Expressions in JSX are written inside _____.
96. React updates the _____ DOM first.
97. Virtual DOM improves _____.
98. Passing data between many components is called _____.
99. Redux is good for _____ applications.
100. Context API is suitable for _____ data.
101. Babel converts JSX into _____.
102. Vite supports _____ (full form of HMR).
103. The entry file in Vite is _____.
104. The root div is located inside _____.
105. Components are stored in the _____ folder.
106. The global stylesheet in Vite is _____.
107. The default port of Vite is _____.
108. Yarn can install cached packages without _____.
109. React comments must be written using _____ braces.
110. React components return _____.

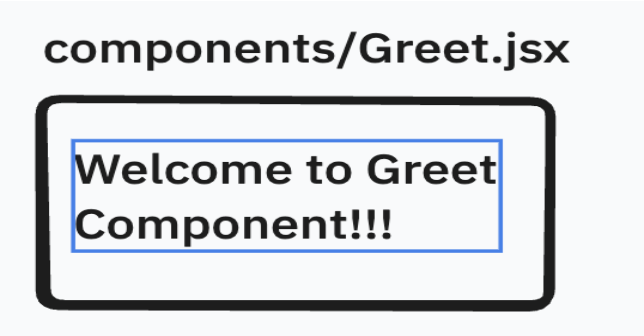
☑ One-Word Question Prompts

111. Who maintains React now?
112. Which language powers JSX?
113. What hook stores state?
114. What hook reads route parameters?
115. What hook handles side effects?
116. What tool converts JSX?

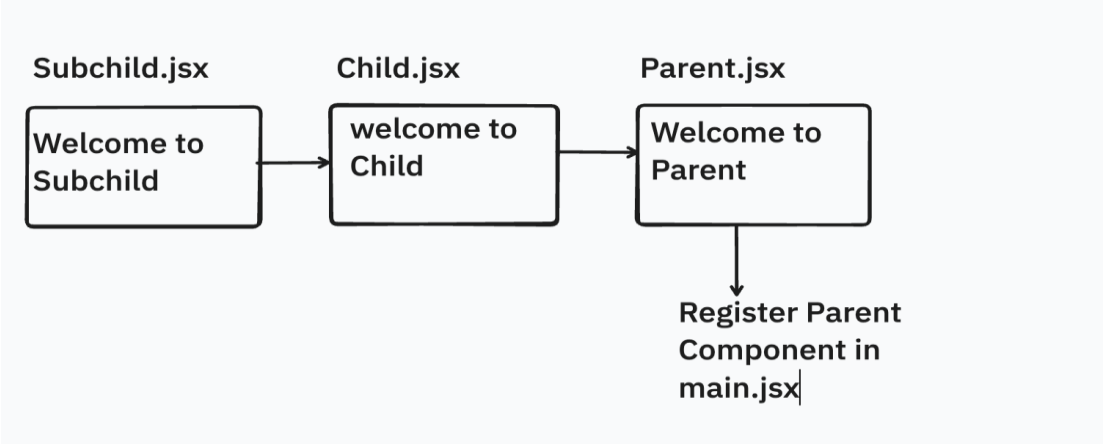
117. What DOM does React update first?
118. What symbol does HashRouter use?
119. What is used for global styles?
120. What is the entry file in Vite?
121. What folder contains packages?
122. What is the unique identifier needed in list rendering?
123. What is the feature that updates only changed screen parts?
124. What is the name of React's build tool discussed in the PDF?
125. What file stores project dependencies?

☑ Coding Questions

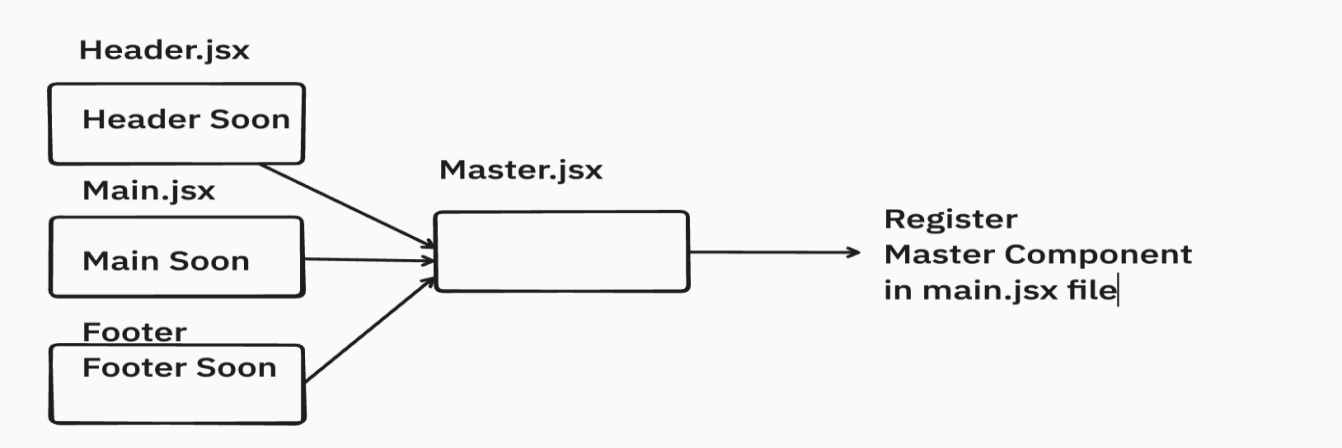
126.



127.



128.


Header.jsx
Header Soon
Main.jsx
Main Soon
Footer
Footer Soon
Master.jsx
Register
Master Component
in main.jsx file