1.Props and State:

I. Props are used to pass data from a parent component to a child component.

Example:

Imagine a Parent and a Child in a family:

Parent: The parent decides what gift to give to the child for their birthday.

Child: The child receives the gift, but doesn't have the power to choose what is inside the gift.
The child just enjoys what the parent gave them.
Gift: The gift is like props in React. It contains something (data) that is passed from the parent to the child.

II. State is used to manage internal component data.

Example:

- You're writing in your diary about your mood today. Initially, you might write "I feel happy."
- Later, after an event, you might write, "I feel sad."

2.Hooks:

In React, hooks are special functions that allow you to use React features (like state, lifecycle methods, etc.) in functional components**.** Before hooks, you could only use features like state and lifecycle methods in class components**.** But with hooks, you can do all of that in functional components (which are simpler and easier to write).

Example:

- useState (State Hook) = Hammer: Think of use State as your hammer. Whenever you need to store something (like data), you use use State to "nail it down" (store it) into your component. You can change or update that "nail" when needed by calling set State.

- useEffect (Effect Hook) = Measuring Tape: The useEffect hook is like your measuring tape. Before you start working on something, you need to measure and check if everything is in place. Similarly, useEffect lets you run code after your component renders — for example, making sure data is fetched from an API after the component has loaded.

3.LocalStorage:

In React, localStorage allows you to store data in the browser's local storage. The data saved in local storage persists even when the page is reloaded or the browser is closed and reopened. This is useful for saving things like user preferences, authentication tokens, or other data that should persist across page refreshes.

Example:

- You might keep a list of items in your box, like receipts or documents, that you need to refer to later.
- In React, you can store data like user settings, authentication tokens, or preferences that you want to persist after a user closes and reopens the application.

4.SessionStorage:

Session Storage in JavaScript is very similar to local Storage, with one key difference: session Storage only persists data for the duration of the page session (as long as the browser tab is open). Once the tab is closed, the data in session Storage is cleared.

Example:

You might use this box to store things that are only relevant for the current session, like a user's temporary settings**,** or items in a shopping cart that the user will check out soon but not need when they return later.

5.Const:

In React, const is a JavaScript keyword that you can use to declare variables that cannot be reassigned once they are assigned a value. It stands for "constant," meaning the variable's value remains the same throughout its lifetime (it cannot be re-assigned). However, the value stored in a const variable can still be mutable if it is an object or array (though the reference to that object or array cannot be changed).

Example:

- You have a container labeled “Office Supplies,” and you put pens, paper, and markers in it. You can add more items or change what’s inside the box, but the label on the box stays the same. The container’s contents can change, but you can’t change the label.
- In React or JavaScript, using const means that once you assign a value to a variable, you can’t change the variable to point to something else, but you can modify the value it holds (if it's an object or array).

6.ReactDOM :

React DOM is a package that serves as the glue between React and the DOM **(**Document Object Model) in web applications. It provides methods that allow React components to interact with the web page's DOM.

Example:

- Imagine you're watching a play at the theater. The director (ReactDOM) is in charge of making sure the actors (React components) perform correctly and that the audience (users) can see them in the right place.

- The actors don't know where they are on stage, but the director makes sure the audience sees them in the right spots — whether it's on the stage, in the spotlight, or moving to a new area of the theater. Similarly, ReactDOM tells the browser where and how to display the components so users can interact with them.

7.Fetch :

In React, fetch() is a JavaScript function that allows you to make HTTP requests (like GET, POST, PUT, DELETE) to interact with APIs and retrieve or send data. It is a built-in function available in most modern browsers and is commonly used in React to handle data fetching from external sources like REST APIs.

Example:

Imagine you're writing a letter asking someone to send you a list of available books at their library.

- You send the letter (make a fetch request).
- The person receives it and writes back to you, listing all the available books (the server or API responds with the data).
- You receive the letter, and now you have the list of books to look at and use in your React app.

8.Axios :

Axios is a popular JavaScript library that is often used in React applications to make HTTP requests. It is a promise-based HTTP client that works both in the browser and in Node.js environments. Axios is an alternative to the native fetch() API and provides additional features like automatic JSON parsing, request and response interceptors, and easier handling of errors.

Example:

Imagine you need to ask someone to send you a document from their office.

- You hire a courier (Axios**)** to deliver your request.
- The courier goes to the office (server), gets the document (data), and brings it back to you.
- You receive the document (data) and can now use it in your application (like displaying it to a user).

9.Charts :

Creating charts in a React application is a common way to visualize data. There are several libraries available to help you integrate charts, and one of the most popular libraries for creating charts in React is Recharts. Other libraries, like Chart**.**js and Victory, are also commonly used in React.

Example:

Imagine you are an artist tasked with drawing a picture that represents the monthly sales of a business.

- You have the data (like how much was sold each month).
- You use a canvas (like a digital screen or a piece of paper) to draw the chart.
- You follow instructions to decide that you'll use a bar chart to represent the sales for each month.
- You paint the bars with colors (representing the sales for each month), and the finished artwork (chart) is ready to show the sales data.

10.Pages :

In React, pages typically refer to the different views or components that are displayed to the user when they navigate through a web application. These pages are often created using React Router, which allows for client-side navigation and helps in rendering different components based on the URL path.

Example:

Imagine you're reading a cookbook:

- Each chapter is a recipe. One chapter could be "Breakfast Recipes," another could be "Dinner Recipes," and another could be "Desserts."
- The table of contents helps you navigate directly to a chapter you're interested in.
- When you go to the Dessert chapter, you're taken directly to that page with all the relevant recipes, just like navigating to the Dessert page on a website.