Props

What are props?

In React, a "prop" is short for "property" and refers to a way to pass data from a parent component to a child component. A prop is essentially an input for a React component.

Where are they used :

1. Props are used to pass data to a component.
2. Props are used in React components to pass data from a parent component to a child component. Props can be used to customise a component's behaviour, content, or appearance.

Props are commonly used in the following scenarios:

1. Customising component content: Props can be used to pass text, images, or other data to a component to customise its content. For example, a Product component could receive props such as name, description, and price to display information about a product.
2. Controlling component behaviour: Props can be used to control a component's behaviour. For example, a Button component could receive a prop such as onClick to specify the function to be called when the button is clicked.
3. Styling component appearance: Props can be used to pass CSS classes or inline styles to a component to customise its appearance. For example, a Card component could receive a prop such as backgroundColor to specify the background colour of the card.
4. Dynamic rendering of components: Props can be used to render components dynamically based on data. For example, a List component could receive an array of objects as props, and use that data to dynamically render a list of items.

Overall, props allow for great flexibility in building reusable React components and creating dynamic and customizable user interfaces.

How are props used :

```
function Greeting(props) {
  return <h1>Hello, {props.name}!</h1>;
}

ReactDOM.render(
  <Greeting name="Alice" />,
  document.getElementById('root')
);
```

In this example, we define a functional component called Greeting that takes a single **prop** called **name**. Inside the function, we can access the name prop using the props argument. We return a **JSX** element that displays a personalised greeting using the name prop.

To use the Greeting component, we create an instance of it by passing in a prop called name with a value of "Alice". We then render the Greeting component by calling **ReactDOM.render**() and passing in the component instance and the target DOM node.

The output of this code will be a heading element that says "Hello, Alice!".

Functional components in React are designed to be pure functions, meaning they don't have any internal state and their output is solely determined by their input (i.e., their props). By using props to pass data into functional components, we can create reusable and composable components that can be easily customised and used in different parts of our application.

Some terms related to Props:

Prop Drilling: Prop Drilling is the process of passing props through multiple levels of nested components in order to make them available to a deeply nested child component. This can result in messy and hard-to-read code, so it's generally recommended to use a state management library like Redux or Context API to avoid prop drilling.

Default Props: Default Props are used to provide default values for props in case they are not passed in by the parent component. This can help prevent errors caused by missing props.

Example :

1. Customising component content:

```
function Product(props) {
  return (
    <div>
      <h2>{props.name}</h2>
      <p>{props.description}</p>
      <p>Price: ${props.price}</p>
```

```
    </div>
  );
}
ReactDOM.render(
  <Product
    name="iPhone 13"
    description="The latest iPhone from Apple"
    price={999.99}
  />,
  document.getElementById('root')
);
```

In this example, we have a **Product** component that takes three props (**name**, **description**, and **price**) to display information about a product.

Example 2 👍

Controlling component behaviour:

```
function Button(props) {
  return (
    <button onClick={props.onClick}>
      {props.label}
    </button>
  );
}

function handleClick() {
  console.log('Button clicked');
```

```
}

ReactDOM.render(
  <Button
    label="Click me"
    onClick={handleClick}
  />,
  document.getElementById('root')
);
```

In this example, we have a **Button** component that takes two props (**label** and **onClick**) to specify the text displayed on the button and the function to be called when the button is clicked.

SOME INTERVIEW QUESTIONS 👍

1. **What are React props and how are they used?**

2. **How do you pass props from a parent component to a child component in React?**

3. Can you provide an example of how to pass a function as a prop in React?

4. What is the difference between props and state in React?

5. How do you set default props in a React component?