Fundamental Concepts of React

→ React is a popular Javascript Library for building user interfaces, particularly single-page applications. Developed & maintained by Facebook, it emphasizes a component-based architecture, allowing developers to create reusable UI components.

→ JSX (JavaScript XML)

- JSX is a syntax extension for JavaScript that allows you to write HTML-like code within JavaScript. It makes it easier to create & visualize the structure of the UI.

Code Example:

```
import React from 'react';
import ReactDOM from 'react-dom';

const element = <h1> Hello, World! </h1>;
ReactDOM.render( element, document.getElementById ('root'));
```

→ Components

. Components are the building blocks of a React application. They encapsulate part of the UI & behaviour into a reusable piece.

Code Example:

```
function Welcome (props)
{
    return <h1> Hello, {props.name} </h1>;
}
const element = <Welcome name = "Sara" />;
ReactDOM.render (element, document.getElementById('root'));
```

⇒ The 'Welcome' function is a functional component that takes 'props' as an argument & returns a JSX element. In this case it renders a 'h1' element with a personalized greeting. Components can be composed together to build complex UIs.

→ State

- State is a built-in object used to contain data or information about the component. A component's state can change over time, typically in response to user actions.

Code Example:

```
class Clock extends React.Component {
  constructor(props) {
    super(props);
    this.state = { date: new Date() };
  }

  componentDidMount() {
    this.timerID = setInterval(() => this.tick(), 1000);
  }

  componentWillUnmount() {
    clearInterval(this.timerID);
  }

  tick() {
    this.setState({
      date: new Date()
    });
  }
```

```
Render()
{
 return (
   <div>
     <h1> Hello, world! </h1>
     <h2> It is { this.state.date.toLocaleTimeString()}. </h2>
   </div>
   );
 }
}

ReactDOM.render(<Clock />, document.getElementById('root'));
```

→ Props

- Props are read-only attributes used to pass data from a parent component to a child component.

Code Example:

```
function App()
{
   return (
     <div>
     <Welcome name="Himanshu" />
     <Welcome name="Aman" />
```

```
    <Welcome name="Ayush" />
    </div>
  );
}

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

ReactDOM.render(<App />, document.getElementById('root'));
```

Here, the 'App' component renders three 'Welcome' components, each with different 'name' prop.

The 'Welcome' component receives the 'name' prop & renders it as 'h1' element.

Props allow components to be reusable & dynamic.