1. What is React?

React is a JavaScript library for building user interfaces, particularly single-page applications. It was developed by Facebook and released as open-source. React allows developers to create reusable UI components and their state efficiently.

Code Example 1:
```jsx
// HelloComponent.jsx
import React from 'react';

const HelloComponent = () => {
  return (
    <div>
      <h1>Hello, React!</h1>
      <p>This is a basic React component.</p>
    </div>
  );
};

export default HelloComponent;
```

Code Example 2:
```jsx
// App.jsx
import React from 'react';
import HelloComponent from './HelloComponent';

const App = () => {
  return (
    <div>
      <h1>My React App</h1>
      <HelloComponent />
    </div>
  );
};

export default App;
```

2. Why use React?

React offers several advantages for building modern web applications:

- **Component-Based Architecture**: React allows developers to break down UI into reusable components, making code more modular and maintainable.
- **Virtual DOM**: React uses a virtual DOM for efficient updates to the actual DOM, resulting in better performance.
- **Declarative Syntax**: React's declarative approach simplifies UI development by allowing developers to describe how the UI should look based on state changes.

Code Example 1:
```jsx
// CounterComponent.jsx
import React, { useState } from 'react';

const CounterComponent = () => {
  const [count, setCount] = useState(0);

  const increment = () => {
    setCount(count + 1);
  };

  return (
    <div>
      <h2>Counter: {count}</h2>
      <button onClick={increment}>Increment</button>
    </div>
  );
};

export default CounterComponent;
```

Code Example 2:
```jsx
// App.jsx
import React from 'react';
import CounterComponent from './CounterComponent';

const App = () => {
  return (
    <div>
      <h1>React Counter App</h1>
      <CounterComponent />
    </div>
  );
};

export default App;
```

3. React Features

React comes with several features that make it a powerful choice for building web applications:

- **JSX**: JSX is a syntax extension for JavaScript that allows developers to write HTML-like code within JavaScript.
- **State Management**: React provides a built-in state management system that allows components to manage their internal state and re-render when state changes.
- **Lifecycle Methods**: React components have lifecycle methods that allow developers to execute code at different stages of a component's life, such as when it's mounted or updated.

These examples provide a basic understanding of React, its advantages, and some of its key features.