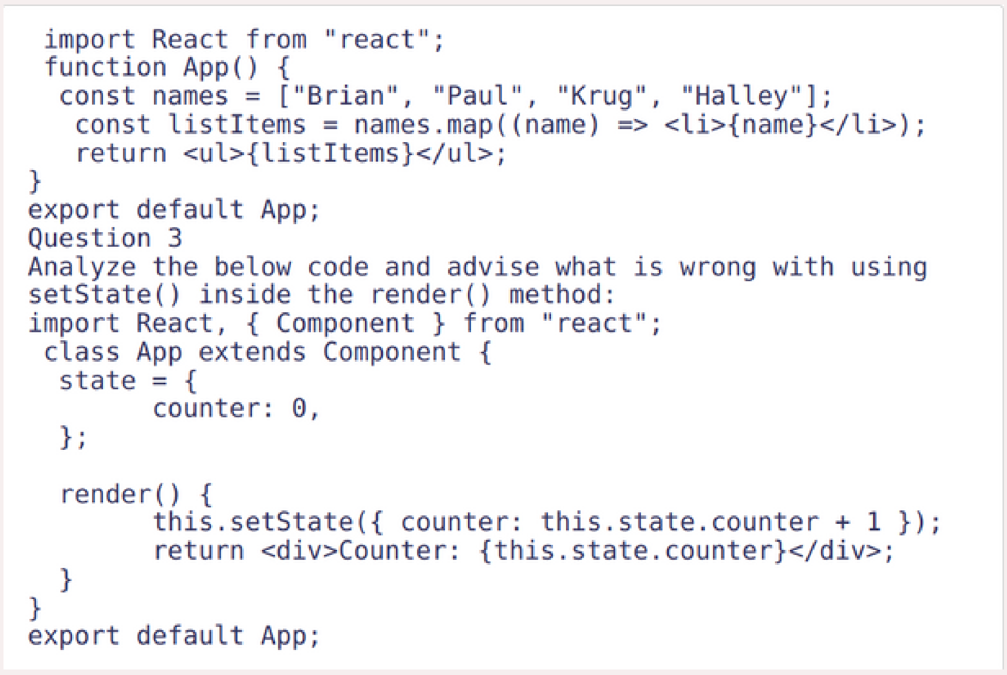
FIND THE ISSUE IN THE BELOW CODE SNIPPET AFTER RENDERING THE LIST OF NAMES

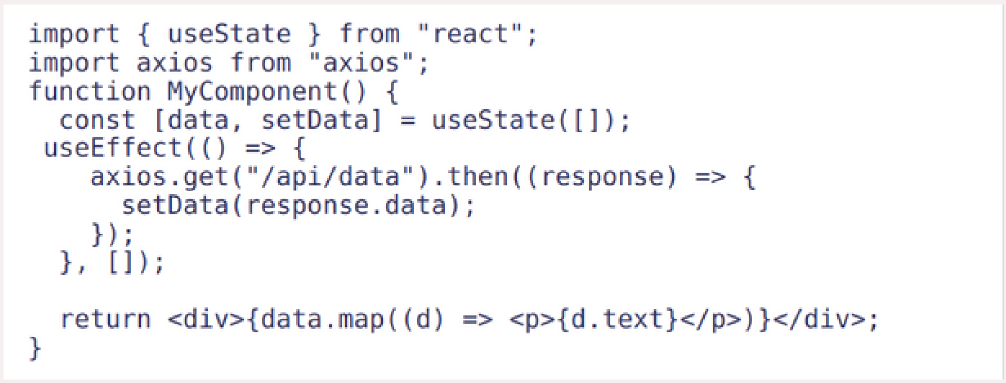
sujith Kumar R



```
 import React from "react";
 function App() {
  const names = ["Brian", "Paul", "Krug", "Halley"];
   const listItems = names.map((name) => <li>{name}</li>);
   return <ul>{listItems}</ul>;
}
export default App;
Question 3
Analyze the below code and advise what is wrong with using
setState() inside the render() method:
import React, { Component } from "react";
 class App extends Component {
  state = {
        counter: 0,
  };

  render() {
        this.setState({ counter: this.state.counter + 1 });
        return <div>Counter: {this.state.counter}</div>;
  }
}
export default App;
```

SEE THE BELOW CODE SNIPPET AND ADVISE, WILL THERE BE ANY ISSUE MAKING A REST API CALL IN A COMPONENT'S USEEFFECT HOOK?



```
import { useState } from "react";
import axios from "axios";
function MyComponent() {
  const [data, setData] = useState([]);
 useEffect(() => {
    axios.get("/api/data").then((response) => {
      setData(response.data);
    });
  }, []);

  return <div>{data.map((d) => <p>{d.text}</p>)}</div>;
}
```

WHAT ARE HIGHER-ORDER COMPONENTS (HOCS)?

Higher-Order Components (HOCs) are a pattern in React for reusing component logic. HOCs are functions that take a component as an argument and return a new component with additional props or behavior.

Example:

```
function withLogging(WrappedComponent) {
  return function LoggingComponent(props) {
    console.log('Props:', props);
    return <WrappedComponent {...props} />;
  };
}
```

In this example, the withLogging function takes a WrappedComponent as an argument and returns a new LoggingComponent that logs the props and renders the WrappedComponent with the same props.

To use the HOC, you can wrap your component like this

```
import React from 'react';
import withLogging from './withLogging';

function MyComponent(props) {
  return <p>Hello, {props.name}!</p>;
}

export default withLogging(MyComponent);
```

This will create a new component that includes the logging behavior, and you can use it just like the original component.