Q1-Explain what the simple List component does.

This is a React component that renders a list of items, with each item being represented by a <li> element. The list items are passed in as an array of objects, each object having a `text` property that represents the text to be displayed for that item.

The component consists of two sub-components, “SingleListItem” and “ListComponent”.

“SingleListItem” is a functional component that renders a single list item. It receives four props:

- “index”: a number representing the index of the item in the list
- “isSelected”: a boolean indicating whether the item is currently selected or not
- “onClickHandler”: a function that will be called when the item is clicked
- “text”: a string representing the text to be displayed for the item

“ListComponent” is also a functional component that renders the entire list. It receives one prop:

- “items”: an array of objects representing the list items

Inside “ListComponent”, the “useEffect” hook is used to reset the “selectedIndex” state variable whenever the `items` prop changes. This is to ensure that if the list is re-rendered with new items, any previously selected item will be unselected.

The “handleClick” function is a callback that will be called whenever an item is clicked. It takes an “index” parameter representing the index of the clicked item and sets the “selectedIndex” state variable to that index.

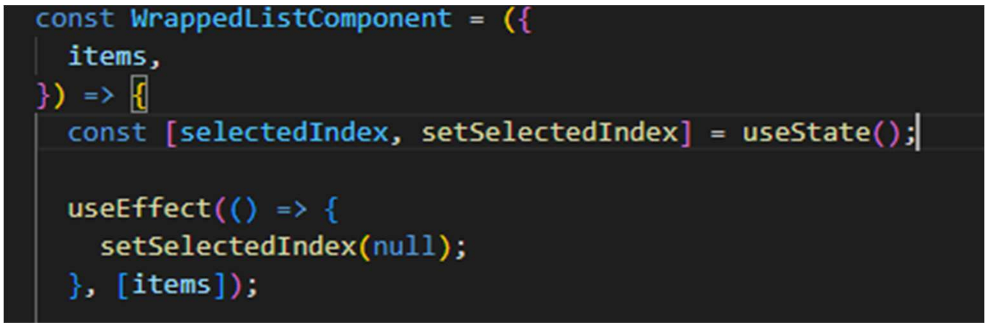
Finally, the “items” array is mapped to an array of “SingleListItem” components, with each component being passed the necessary props to render the corresponding list item. The “memo” function is used to optimize performance by memoizing the components and preventing unnecessary re-renders when props have not changed.

Overall, this component provides a basic list functionality that allows users to select one item at a time. When an item is selected, it is highlighted with a green background colour, otherwise it is displayed with a red background colour.

Q2- What problems / warnings are there with code?

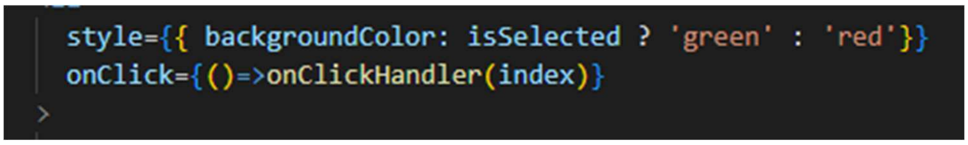
There are a few issues with the code:

1- **useState usage:** In the WrappedListComponent, the state is declared incorrectly. The line const [setSelectedIndex, selectedIndex] = useState(); should be changed to const [selectedIndex, setSelectedIndex] = useState(null);



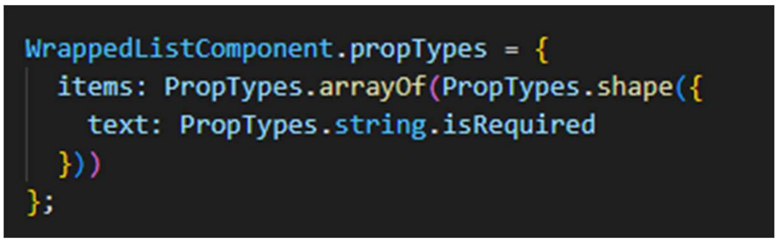
```
const WrappedListComponent = ({
  items,
}) => {
  const [selectedIndex, setSelectedIndex] = useState();

  useEffect(() => {
    setSelectedIndex(null);
  }, [items]);
```

2- **onClickHandler usage**: In the WrappedSingleListItem, the onClickHandler prop is being called immediately with the index argument instead of passing a function to be called when the element is clicked. To fix this, we can change onClick={onClickHandler(index)} to onClick={() => onClickHandler(index)}.



```
  style={{ backgroundColor: isSelected ? 'green' : 'red'}}
  onClick={()=>onClickHandler(index)}
>
```

3- **PropTypes usage:** In the WrappedListComponent, the items prop is not defined correctly in the propTypes object. Instead of items: PropTypes.array(PropTypes.shapeOf({ text: PropTypes.string.isRequired })), it should be defined as items: PropTypes.arrayOf(PropTypes.shape({ text: PropTypes.string.isRequired })). This will ensure that the items prop is an array of objects with a text property that is a required string.



```
WrappedListComponent.propTypes = {
  items: PropTypes.arrayOf(PropTypes.shape({
    text: PropTypes.string.isRequired
  }))
};
```

4- **defaultProps usage:** In the WrappedListComponent, the items prop is given a default value of null. This may cause issues in the items.map function later in the code. Instead, we can give it a default value of an empty array by changing items: null to items: [].

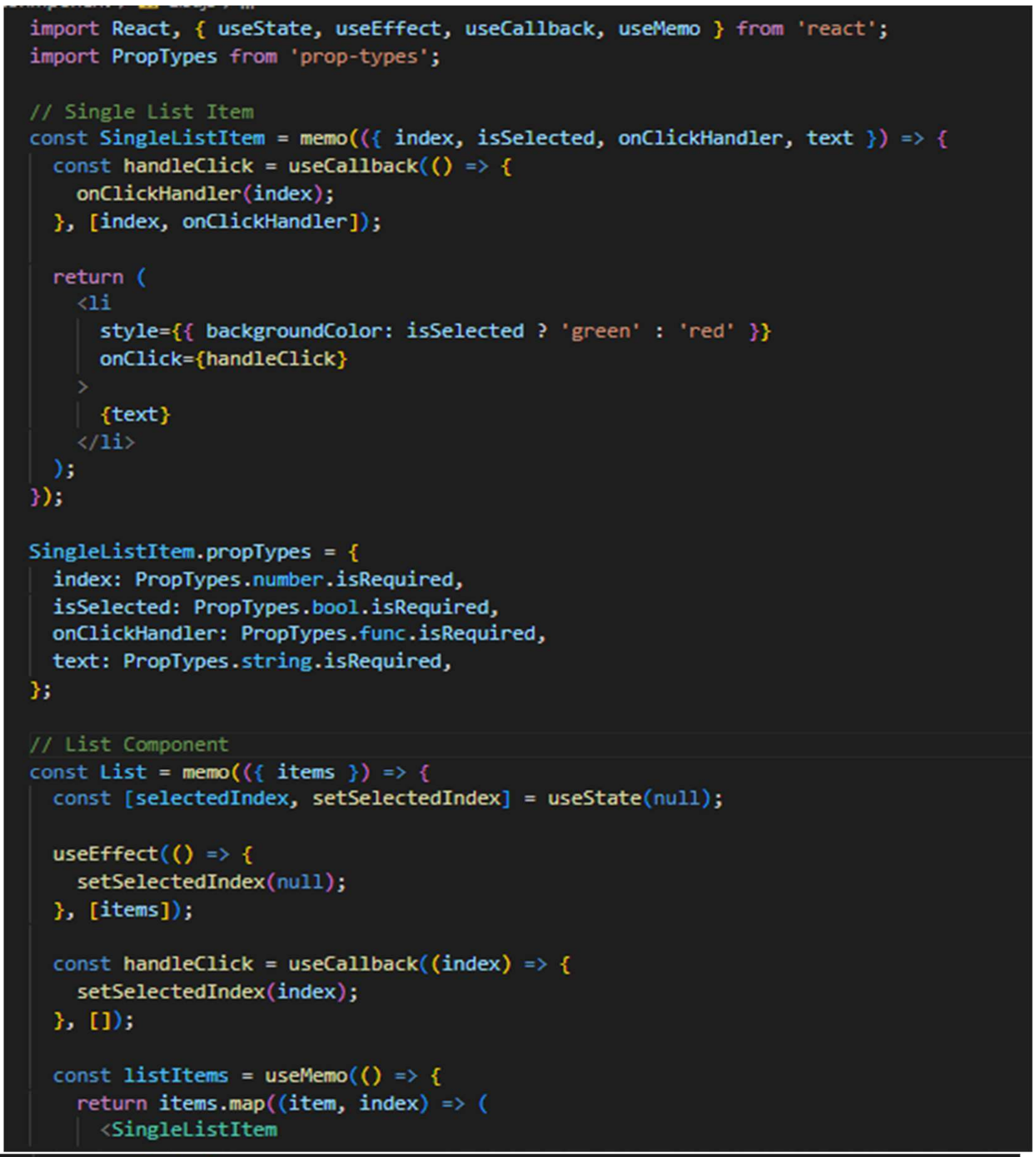
Q3-Please fix, optimize, and/or modify the component as much as you think is necessary.

```
import React, { useState, useEffect, useCallback, useMemo } from 'react';
import PropTypes from 'prop-types';

// Single List Item
const SingleListItem = memo(({ index, isSelected, onClickHandler, text }) => {
  const handleClick = useCallback(() => {
    onClickHandler(index);
  }, [index, onClickHandler]);

  return (
    <li
      style={{backgroundColor: isSelected ? 'green' : 'red' }}
      onClick={handleClick}
    >
      {text}
    </li>
  );
});

SingleListItem.propTypes = {
  index: PropTypes.number.isRequired,
  isSelected: PropTypes.bool.isRequired,
  onClickHandler: PropTypes.func.isRequired,
  text: PropTypes.string.isRequired,
};

// List Component
const List = memo(({ items }) => {
  const [selectedIndex, setSelectedIndex] = useState(null);
  useEffect(() => {
    setSelectedIndex(null);
  }, [items]);

  const handleClick = useCallback((index) => {
    setSelectedIndex(index);
  }, []);

  const listItems = useMemo(() => {
    return items.map((item, index) => (
      <SingleListItem
        key={index}
        onClickHandler={handleClick}
        text={item.text}
        index={index}
        isSelected={selectedIndex === index}
      />
    ));
  }, [handleClick, items, selectedIndex]);

  return <ul style={{ textAlign: 'left' }}>{listItems}</ul>;
});

List.propTypes = {
  items: PropTypes.arrayOf(
    PropTypes.shape({
      text: PropTypes.string.isRequired,
    }).isRequired
  ),
};
```

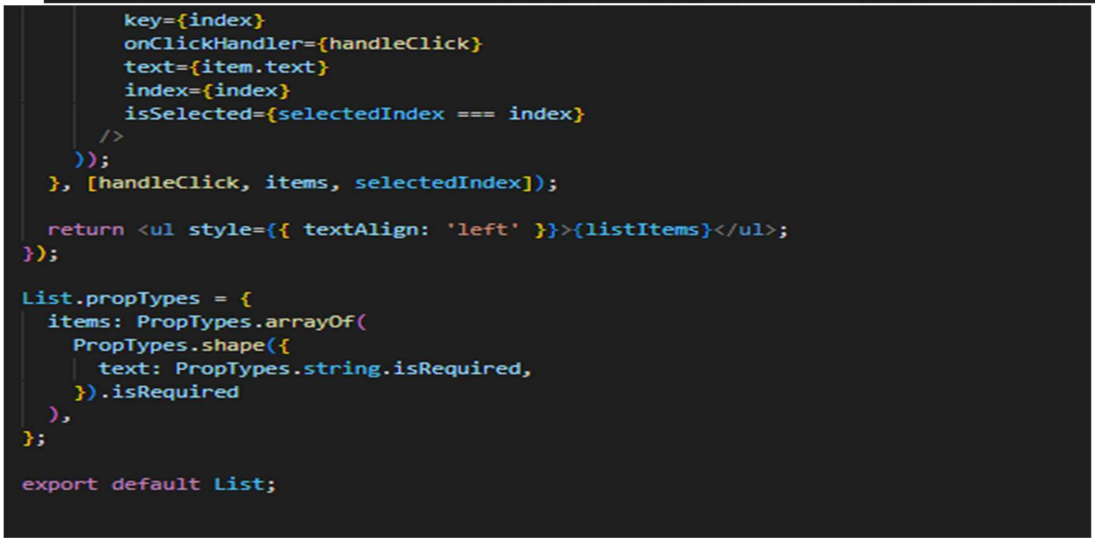
```
export default List;
```

Changes made:

- In the SingleListItem component, the onClick handler is updated to use a useCallback hook to memoize the function and prevent unnecessary re-renders. Also, the propTypes definitions are updated to require the index, isSelected, onClickHandler, and text props.
- In the List component, the setSelectedIndex and selectedIndex state variables are defined correctly. The useEffect hook is updated to reset the selectedIndex state variable when the items prop changes. The handleClick function is also updated to use useCallback hook for memoization. Finally, a useMemo hook is added to memoize the list items to prevent unnecessary re-renders.
- The propTypes definitions for the List component are updated to require an array of objects with a text property.
- The defaultProps for WrappedListComponent are removed since they are not necessary.



```
import React, { useState, useEffect, useCallback, useMemo } from 'react';
import PropTypes from 'prop-types';

// Single List Item
const SingleListItem = memo(({ index, isSelected, onClickHandler, text }) => {
  const handleClick = useCallback(() => {
    onClickHandler(index);
  }, [index, onClickHandler]);

  return (
    <li
      style={{ backgroundColor: isSelected ? 'green' : 'red' }}
      onClick={handleClick}
    >
      {text}
    </li>
  );
});

SingleListItem.propTypes = {
  index: PropTypes.number.isRequired,
  isSelected: PropTypes.bool.isRequired,
  onClickHandler: PropTypes.func.isRequired,
  text: PropTypes.string.isRequired,
};

// List Component
const List = memo(({ items }) => {
  const [selectedIndex, setSelectedIndex] = useState(null);

  useEffect(() => {
    setSelectedIndex(null);
  }, [items]);

  const handleClick = useCallback((index) => {
    setSelectedIndex(index);
  }, []);

  const listItems = useMemo(() => {
    return items.map((item, index) => (
      <SingleListItem
```



```
        key={index}
        onClickHandler={handleClick}
        text={item.text}
        index={index}
        isSelected={selectedIndex === index}
      />
    ));
  }, [handleClick, items, selectedIndex]);

  return <ul style={{ textAlign: 'left' }}>{listItems}</ul>;
});

List.propTypes = {
  items: PropTypes.arrayOf(
    PropTypes.shape({
      text: PropTypes.string.isRequired,
    }).isRequired
  ),
};

export default List;
```