Equbal Fahmi 12018038 LPU

1.Explain what the simple `List` component does.

The List component is a simple React component that renders an unordered list (<ul>) of items. It takes an array of items as a prop and renders a SingleListItem component for each item in the array.

The SingleListItem component is a child List component that represents a single item in a list. Renders a list item (<li>) with the item text and a background color that changes depending on whether the item is selected or not.

The List component also keeps track of the currently selected item by maintaining the index of the selected item in the component's state. When you click an item, the handleClick function updates the state with the index of the item you clicked, causing the isSelected prop of the corresponding SingleListItem component to update and the item's background color to change accordingly.

In summary, the List component is a reusable component that can render a list of items with the ability to select and highlight one item at a time.

2.What problems / warnings are there with code?

There are several issues and warnings with this code that need to be addressed:

1. The `isSelected` prop in the `SingleListItem` component is not passed correctly. It should be passed as a boolean value, but is passed as the state variable `selectedIndex`. To solve this, change `isSelected={selectedIndex}` to `isSelected={selectedIndex === index}`.

2. The `items` prop is not properly validated in the `ListComponent` component. The PropTypes `array` validator should be wrapped in `PropTypes.shape()` to determine the shape of the array elements. Replace `items: PropTypes.array(PropTypes.shapeOf({text: PropTypes.string.isRequired}))` with `items: PropTypes.arrayOf(PropTypes.shape({text: PropTypes.string.isRequired}))`.

3. The `SingleListItem` component does not use the `index` prop to set the key for the list item. This can cause issues with optimizing React's rendering performance. To

solve this, add a `key` support with an `index` value to the ,<li>` element: `<li key={index} ...`.

4. Prop `onClickHandler` in `SingleListItem` component is not used correctly. A function should be passed, but it is called immediately with the `index` argument. To solve this, change `onClick={onClickHandler(index)}` to `onClick={() => onClickHandler(index)}`.

5. The `index` pad is not validated in the `SingleListItem` component. Add `index: PropTypes.number.isRequired` to the `propTypes` object.

6. In the `ListComponent` component, the `setSelectedIndex` function should be `const [selectedIndex, setSelectedIndex] = useState(null);`, not `const [setSelectedIndex, selectedIndex] = useState();`. The order of destructed values should be `[selectedIndex, setSelectedIndex]`.

7. The `ListComponent` component creates a new `handleClick` function on each render of the component. To solve this, use the `useCallback` hook to remember the function: `const handleClick = useCallback(index => { setSelectedIndex(index); }, []);`.

Addressing these issues will make the code more robust and performant.

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

Here's a modified version of the code that addresses the issues mentioned before and makes some optimizations and improvements:

```
import React, { useState, useEffect, useCallback, memo } from 'react';

import PropTypes from 'prop-types';
```

```
// Single List Item
const SingleListItem = memo(({ index, isSelected, onClickHandler, text }) => {
  return (
    <li
      style={{ backgroundColor: isSelected ? 'green' : 'red' }}
      onClick={onClickHandler}
    >
      {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);
  },
  [setSelectedIndex]
);

return (
  <ul style={{ textAlign: 'left' }}>
    {items.map(({ text }, index) => (
      <SingleListItem
        key={index}
        onClickHandler={() => handleClick(index)}
        text={text}
        index={index}
        isSelected={selectedIndex === index}
      />
    ))}
  </ul>
```

```
  );
});

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

List.defaultProps = {
  items: [],
};

export default List;
```

Changes made:

Moved the definition of the SingleListItem component into its own function definition.

Changed isSelected prop validation to isRequired.

Removed unnecessary function call in onClick handler for SingleListItem component.

Added key support to the SingleListItem component to improve rendering performance.

Changed the onClickHandler support to use the arrow function to avoid creating a new function each time it is rendered.

Changed the order of destructuring values in the useState hook in the List component.

Used a useCallback hook to remember the handleClick function and added setSelectedIndex to its dependencies array.

Added default props for List component.

Rearranging the PropTypes validators and adding a trailing comma to each line for better readability.