Name: Yash Kumar Role: Frontend Developer

Email: kumaryash1212@gmail.com

1.) Explain what the simple List component does.

On websites, lists are primarily used to display menus and are used to present data in an ordered format. Similar to how we create lists in JavaScript, lists in React can be created.

It displays a list of items in linear fashion either vertically or horizontally. To use the Simple List component in React, we can create a functional or class component that renders a container element, such as an unordered list **<ul>** or a div ** **, and maps through an array of data to render individual list items. List navigation is usually accomplished using the map() function.

The Simple List component in React can be customized and extended in various ways to suit different design and functionality requirements. For example, we can add additional content elements, such as images or descriptions, to each list item, or we can include interactive elements, such as buttons or checkboxes, to enable users to take actions on the items.

2.) What problems / warnings are there with code?

- The useState hook in **ListComponent** is being used incorrectly. Instead of **const [setSelectedIndex, selectedIndex] = useState();** it should be **const [selectedIndex, setSelectedIndex] = useState();**

- We can add **?** after items in **items.map((item, index)** as:
  items?.map((item, index)
  to avoid a runtime error if the items array is null or undefined.

  Before adding the optional chaining operator "**?**", if we did not pass any value to the "items" prop

```
▶Uncaught TypeError: Cannot read properties of null (reading 'map')
    at WrappedListComponent (list.js:38:1)
    at renderWithHooks (react-dom.development.js:16305:1)
    at updateFunctionComponent (react-dom.development.js:19588:1)
    at beginWork (react-dom.development.js:21601:1)
    at HTMLUnknownElement.callCallback (react-dom.development.js:4164:1)
    at Object.invokeGuardedCallbackDev (react-dom.development.js:4213:1)
    at invokeGuardedCallback (react-dom.development.js:4277:1)
    at beginWork$1 (react-dom.development.js:27451:1)
    at performUnitOfWork (react-dom.development.js:26557:1)
    at workLoopSync (react-dom.development.js:26466:1)
```

- We can add **key={index}** prop with a unique value as index in to **SingleListItem** in **ListComponent** as react uses keys to identify which items have changed, been added, or been removed from a list.

Before adding key:

```
▶Warning: Each child in a list should have a unique "key" prop.

Check the render method of `WrappedListComponent`. See https://reac
information.
    at WrappedSingleListItem (http://localhost:3000/static/js/bundl
    at WrappedListComponent (http://localhost:3000/static/js/bundle
    at div
    at App
```

- The **PropTypes** definition for **items** is not correct. Instead of **PropTypes.array(PropTypes.shapeOf({ text: PropTypes.string.isRequired }))**, it should be **PropTypes.arrayOf(PropTypes.shape({ text: PropTypes.string.isRequired }))**.

- In **WrappedSingleListItem**, the onClick handler should be a function that calls the **onClickHandler** prop with the index value like this:
  onClick = {() => onClickHandler(index)}.
  Otherwise, it will be immediately invoked when the component is rendered.

- **isSelected** prop passed to **SingleListItem** in **ListComponent** should be a Boolean value, but **selectedIndex** is being passed, which is a number. It should be changed to:
  isSelected = {selectedIndex === index}.

3.) **Please fix, optimize, and/or modify the component as much as you think is necessary.**
File attached along with this doc in the GitHub repository.

List.js

```
import React, { useState, useEffect, memo } from "react";
import PropTypes from "prop-types";

// Single List Item
const WrappedSingleListItem = ({ index, isSelected, onClickHandler, text }) =>
{
  return (
    <li
      style={{ backgroundColor: isSelected ? "green" : "red" }}
      onClick={() => onClickHandler(index)}
    >
      {text}
    </li>
  );
};

WrappedSingleListItem.propTypes = {
  index: PropTypes.number,
```

```
  isSelected: PropTypes.bool,
  onClickHandler: PropTypes.func.isRequired,
  text: PropTypes.string.isRequired,
};

const SingleListItem = memo(WrappedSingleListItem);

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

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

  const handleClick = (index) => {
    setSelectedIndex(index);
  };

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

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

WrappedListComponent.defaultProps = {
  items: null,
};

const List = memo(WrappedListComponent);

export default List;
```

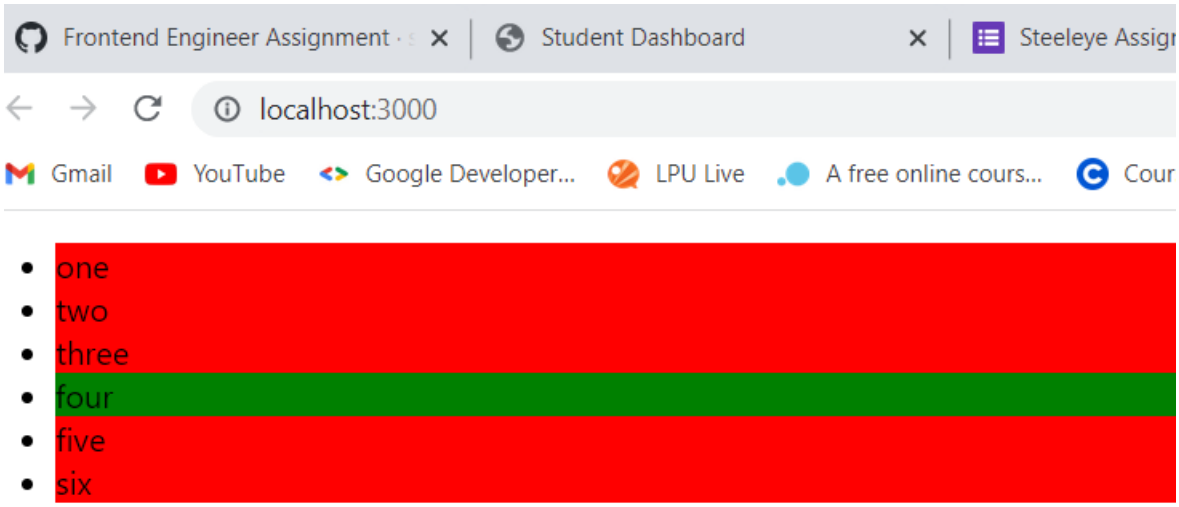
App.js

```
import "./App.css";
import List from "./list";

function App() {
  const itemsList = [
    { text: "one", },
    { text: "two", },
    { text: "three", },
    { text: "four", },
    { text: "five", },
    { text: "six", },
  ];
  return (
    <div className="App">
      <List items={itemsList} />
    </div>
  );
}

export default App;
```

Output:



Deployment Link: https://steeleye-assignment-gamma.vercel.app/