STEELEYE ASSIGNMENT

Lovely Professional University

Front End

Name – Gautam Mishra

College Registration No. : 12010046

Application Link -> https://gauatmmishra-steeleye-frontend.netlify.app/

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

Ans :

The given code has a React component called **"List"** that renders a list of items with selectable items. It consists of two main parts:

1. **WrappedSingleListItem Component**: The WrappedSingleListItem component is a functional component that represents a single item in the list. It receives the following props:

- **index:** A number that represents the index of the item in the list.
- **isSelected:** A boolean that indicates whether the item is currently selected or not.
- **onClickHandler:** A function that handles the click event on the list item.
- **text:** A string that represents the text content of the list item.

Inside the component, an HTML **li** element is rendered with the following attributes:

- **style:** The background color of the list item is determined based on the value of isSelected prop. If **isSelected** is true, the background color is set to 'green', indicating that the item is selected, otherwise it is set to 'red'.
- **onClick:** The **onClickHandler** prop is called with the index prop as an argument when the list item is clicked.
- **text:** The text prop is used as the text content of the list item.

SingleListItem Component: The SingleListItem component is a memoized version of the **WrappedSingleListItem** component, created using the memo function provided by React. Memoization is a performance optimization technique that helps to prevent unnecessary renders of a component if its props have not changed. This can improve the performance of the component by avoiding unnecessary re-renders when the parent component updates.

2. **WrappedListComponent Component:** The WrappedListComponent component is the main component that renders the list of items. It receives the following prop:

- **items:** An array of objects that represent the items in the list. Each object should have a text property which is a string representing the text content of the item.

Inside the component, the **useState** hook is used to manage the state of the selected item's index, with the **selectedIndex** state variable initialized as undefined (since no item is selected initially).

The **useEffect** hook is also used to reset the selected index to null whenever the items prop changes. This is achieved by providing items as a dependency array to the **useEffect** hook, so that whenever items prop changes, the effect will be triggered and the selected index will be reset to null.

The component renders an HTML **ul** element with left-aligned text. For each item in the items prop, it renders a **SingleListItem** component with the necessary props passed down, including the **onClickHandler** to handle item selection. The selected item is determined by comparing the **selectedIndex** state with the current index in the map function. If **selectedIndex** is equal to the current index, it means the item is selected, and the **isSelected** prop of the **SingleListItem** component is set to true, otherwise it is set to false.

Finally, the **WrappedListComponent** is memoized using the **memo** function for performance optimization. This means that the component will only re-render if its props (items) or state (selectedIndex) change, avoiding unnecessary re-renders and improving performance.

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

Ans:

- ✓ The prop type definition for the items prop in **WrappedListComponent** is incorrect. It should be **PropTypes.arrayOf** instead of ***PropTypes.array***, and the shape

definition should be wrapped in parentheses: ***PropTypes.arrayOf(PropTypes.shape({ ... })).***

- ✓ In the **WrappedSingleListItem** component, the **onClick** handler is not defined correctly. It should be a function that is called when the li element is clicked. Instead, the **onClick** handler is being immediately called with the index argument. This will cause the ***onClick handler*** to be called during rendering, which is not what we want.

- ✓ In the **WrappedListComponent** component, the ***setSelectedIndex*** hook is not being used correctly. The **setSelectedIndex** function should be called to update the state, but it is being passed as the initial state value to ***useState***. This will cause s**etSelectedIndex** to be undefined, which will cause errors when trying to call it.

- ✓ The ***isSelected*** prop in **WrappedSingleListItem** should be a **boolean** indicating whether the current item is selected. However, it is being passed the **selectedIndex** state value, which is a number. This will cause errors when trying to set the **backgroundColor** style of the li element based on the ***isSelected*** prop.

- ✓ The **selectedIndex** state value in **WrappedListComponent** should be initialized to null instead of undefined. This will ensure that it has the correct type (number) and value when it is first used.

- ✓ The items prop in **WrappedListComponent** should have a default value of '**[]'** instead of null. This will ensure that the component does not throw an error when it is first rendered

- ✓ The index prop in **WrappedSingleListItem** is not actually used in the component. It can be removed.

- ✓ The **onClickHandler** prop in **WrappedSingleListItem** should have a default value of an empty function ***(() => {})*** instead of being marked as required. This will ensure that the component does not throw an error when it is first rendered.

- ✓ The text prop in **WrappedSingleListItem** should have a default value of an empty string **('')** instead of being marked as required.

This will ensure that the component does not throw an error when it is first rendered.

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

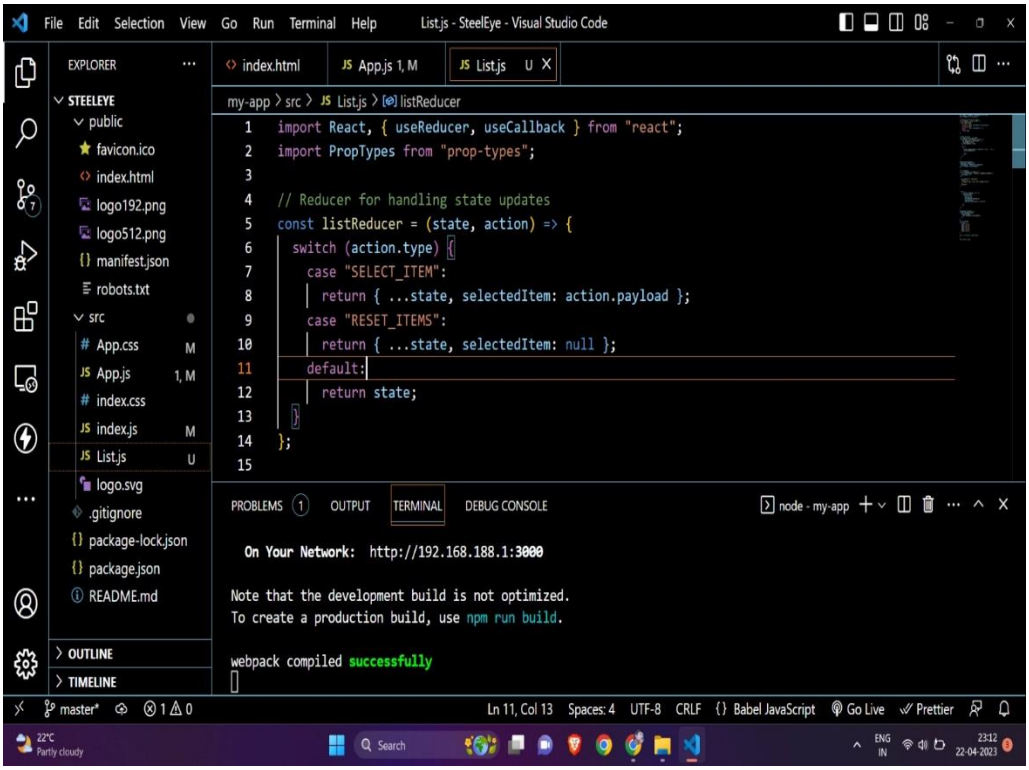
Ans :

Optimized Code/modified code -

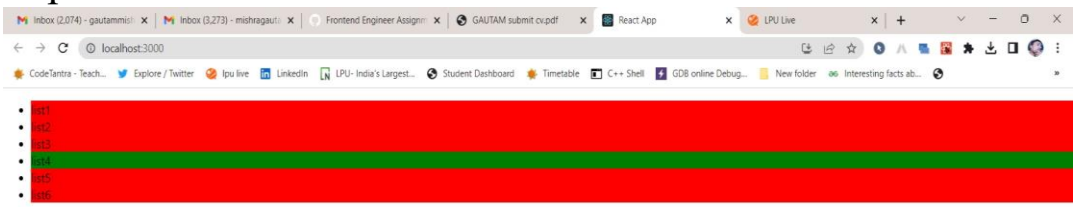
```
import React, { useReducer, useCallback } from "react";
import PropTypes from "prop-types";

// Reducer for handling state updates
const listReducer = (state, action) => {
  switch (action.type) {
    case "SELECT_ITEM":
      return { ...state, selectedItem: action.payload };
    case "RESET_ITEMS":
      return { ...state, selectedItem: null };
    default:
      return state;
  }
};

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

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

SingleListItem.propTypes = {
  isSelected: PropTypes.bool,
```

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

// List Component
const ListComponent = ({ items }) => {
  const [{ selectedItem }, dispatch] = useReducer(listReducer, {
    selectedItem: null
  });

  const handleClick = useCallback(
    (text) => {
      dispatch({ type: "SELECT_ITEM", payload: text });
    },
    [dispatch]
  );

  const resetItems = useCallback(() => {
    dispatch({ type: "RESET_ITEMS" });
  }, [dispatch]);

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

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

export default ListComponent;
```

After removing warnings:


File Edit Selection View Go Run Terminal Help List.js - SteelEye - Visual Studio Code
EXPLORER
index.html
App.js 1, M
List.js U
my-app > src > List.js > listReducer
STEELEYE
public
favicon.ico
index.html
logo192.png
logo512.png
manifest.json
robots.txt
src
App.css M
App.js 1, M
index.css
index.js M
List.js U
logo.svg
.gitignore
package-lock.json
package.json
README.md
OUTLINE
TIMELINE
1 import React, { useReducer, useCallback } from "react";
2 import PropTypes from "prop-types";
3
4 // Reducer for handling state updates
5 const listReducer = (state, action) => {
6 switch (action.type) {
7 case "SELECT_ITEM":
8 return { ...state, selectedItem: action.payload };
9 case "RESET_ITEMS":
10 return { ...state, selectedItem: null };
11 default:
12 return state;
13 }
14 };
15
PROBLEMS 1 OUTPUT TERMINAL DEBUG CONSOLE
node - my-app
On Your Network: http://192.168.188.1:3000
Note that the development build is not optimized.
To create a production build, use npm run build.
webpack compiled successfully
master* 1 0
Ln 11, Col 13 Spaces: 4 UTF-8 CRLF {} Babel JavaScript Go Live Prettier
22°C Partly cloudy
Search
ENG IN
23:12 22-04-2023

Output:


localhost:3000
list1
list2
list3
list4
list5
list6


21°C Partly cloudy
Search
ENG IN
23:35 22-04-2023