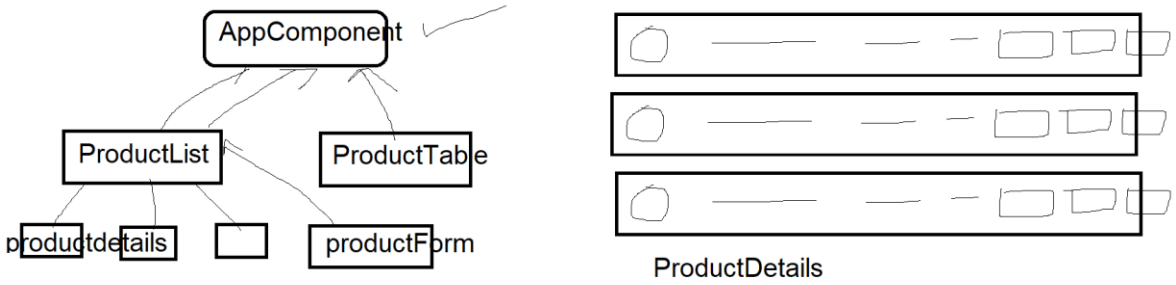

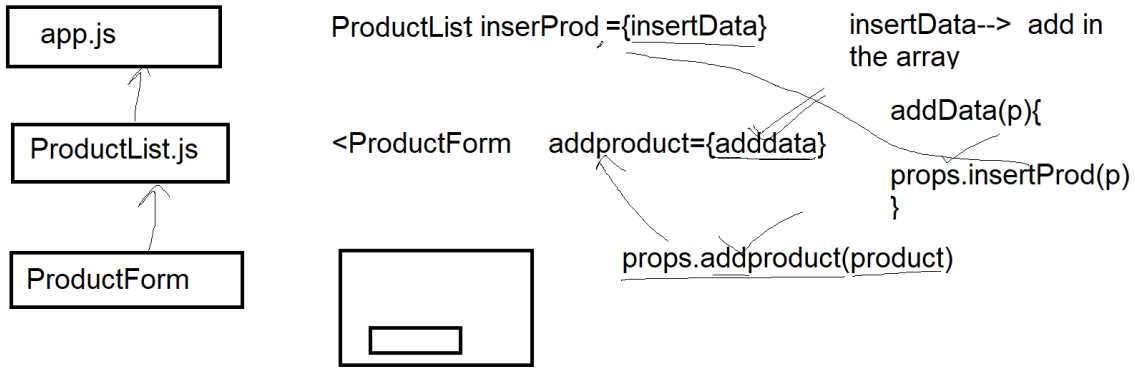

Step

1. Add <BrowserRouter> in index.js, unless you add this, Routing does not work.
   Because the component in which you add <Routes> has tobe enclosed in <BrowserRouter>, also add import for Browser router in index.js
2. Add ProductManagement System in App.js as header
3. Design MainNavbar component and add it in app.js
   a. From bootstrap library find the navbar formatting and paste it in return() of MainNavBar
   b. Replace all <a> with NavLink and href with to, in MainNavbar
      i. We are designing SPA and we do not want to refresh the page, and anchor tag href property refresh and reload the page to stop that, hence we usereact-router-dom NavLink component and to property.
4. Add Routes for each component.
   a. Add all Routes entry in App.js
   b. And design all components, to just display name of each component.
5. Then design a class ProductService in ProductSerivce.js in service folder, add array of products by name prodarr in the constructor and initialize it.
   And also add getAllProducts method to return the array, later change this to axios.get function
6. If you want to store Product objects by using constructor then add Product class, in Product.js in src folder.(This step is optional)
7. To design ProductTable component, we need to retrieve product array from ProductService class, hence
   a. Create a prodarr as a state and initialize it to empty array(use useState hook)

    b. To inialize the array call getAllProducts function from ProductService class, in useEffect initialization hook (useEffect(()=>{},[]), empty array indicates, this useEffect will get executed only once in lifetime of the component.
    c. Then select bootstrap template for display table and replace all tr tags with map function to convert array into tr tags
    d. Also add key in each tr tag, and initialize it to product id, this helps react virtul DOM to identify the component uniquely, and it improves the performance.
8. Design ProductList component using same steps as ProductTable, add view and delete and edit buttons for each row
    a. To design each row we are using ProductDetails component
    b. In Product detail component, we are adding bootstrap 12 grid column to arrange the row contents.
9. To add functionality on buttons in ProductDetails component
    a. Wrap edit button in Link component, it's a built-in component, by react-router-dom, when you click on the button and if you just want to change the url , then use Link component, assign url to, to property of Link component, if you want to pass the data, then assign it to state property, this state object will be available in new component, in location object,
    b. Wrap view button also in which Link Component, because we want to change the link, assign url to , to property.
    c. Since in delete button we want to add logic, hence use onClick event and call function deleteProduct and pass id as a parameter.
    d. Add deleteProduct function, in the function use ProductService.deleteProduct, to actually delete the product from the array, so add deleteproductbyid function in ProductService class.
10. To design ProductForm component
    a. copy From template from the bootstrap library and do the change is Id, name and type propery, classe all input tags, and change class to className.
    b. To perform two-way binding, add value and onChange event in each text box,
    c. For storing values from the form ass formdetails object as state, and initialize all its properties to "".
    d. Add onclick event on Add product button, check all the text boxes are filled, other wise show error message, otherwise call ProductService.addProduct function and pass the object, to add in the array, or in the database using axios.
    e. To change the url back to /list, use navigate function (use useNavigate hook).
11. To design Productedit, copy all the contents of ProdiuctForm, and change Add product button to Update Product button, and add onclick event, to update the product.
    a. To update data we need to show thw data in the form as soon as form appears in the browser, hence initialize formdetails in useEffect hook to the data coming via Link from ProductDetails (useLocation will give access to location object)
    b. Once you initialize form details, all data will be visible in the form, change Productid text box to readOnly
    c. In onclick event of the button call updateProduct to modify the product in the arry or send put request to webservice to update database. Add updateProduct function in ProductService class also.