**Day 2 :14 Jul. 24**

**HOC :** Higher Order Component design pattern.

HOC is a function that takes a component as argument and return that function with new component with new enhancement functionality. This pattern mainly we use to do reusability component logic.

Pts

1. Input : HOC accept as a component (ie component can be functional style or class style). Typically that component we call as Wrapped Component as arguments.
2. Logic : it add some logic
   1. Props manipulation
   2. State management
   3. Life cycle methods. (class components )
   4. Side effects.
3. Output : The HOC return new component that wrap the original component which contains some additional functionality.

HOC concept using string manipulation

create-react-app hoc-string-manipulation

string operation : HOC responsible to convert string into upper case, lower case, substring, search the contents.

React router : it is use to navigate from one page to another page.

Context api

HOC is private router.

**create-react-app react-login-private-router-app**

cd **react-login-private-router-app**

**npm install react-router-dom**

login component it contains login page

logout component responsible to provide the button do to logout.

home component public component you can access without login.

dashboard component private/protected it can allow to open only if we do login.

Render props design pattern

If we are using class style component class provide one the life cycle method ie render which help to render or send JSX(DOM) element on DOM.

In function style component we no need to use render function.

A component with a render props take a function that return a React Element or React DOM Element ie JSX and calls it instead of implementing its own render logic.

In simple way passing the props as function using render from parent to child components.

Use cashes

1. Sharing statefull data logic ie such as fetching data, handling user input or managing timer.
2. Conditional rendering
3. Composition : has a relationship

**create-react-app react-render-props-app**

Compound component design pattern.

In compound component design pattern we can use more than one component to share the state data implicitly and allow to communicate with each others in the background. We can share the data between one component to another component using props.

In this design pattern multiple component works together to have shared state and handler the business logic to achieve some specific task.

App.js 🡪 Parent component

ToggleProvider -🡪

Toggle 🡪Data display information

ToggleConsumer 🡪 consumer consume the data.

create-react-app compound-component-design-app

atomic design pattern : In React JS Automatic design pattern are use atoms. Atoms are the smallest component that can use on their own such as TextField, PasswordField, Button, Search Bar, Menu bar etc.

Input Atoms

Button Atoms

Atomic design pattern

1. Atoms : the smallest building block of UI component like button, input field or all html form tags.
2. Molecules : group of more than one atoms provide to create the simple or complex forms.
3. Organism : using molecules we can create specific component like Header component, Footer component, Form (Login Form, Application Form, etc).
4. Template : Wireframes or layout that showcase the final design.
5. Pages : specific the instance of template where real content is display.

Using JSX or Using template page.

Controlled and Un Controlled Component design pattern

Controlled component control the property using useState in function style or this.state in class style component. If variable is consider as state variable those variable part Virtual DOM elements.

Here we can use useState() in functional components.

UnControlled component. The property of UnControlled component controlled by Actual DOM. useRef() hook. To get the value of variable controlled by actual dom we need to use useRef hook in function component.

create-react-app controlled-and-uncontrolled-components