Types of components

Component describe the part of user interface

They are the reusable can be nested inside other component

Stateless functional component

This function is a javascript function which return Html

import React from 'react'

export default function Test() {

  return (

    <div>Test</div>

  )

}

Arrow function

const Test = () => <h1>Hello world</h1>

export default Test;

&

export const Test = () => <h1>Hello world</h1>

but it only can import within {Test}

Statefull Class component

Class component is es6 class that extend component class

In this must have render method for returning a html

import React, { Component } from 'react'

export default class Test extends Component {

  render() {

    return (

      <div>Test</div>

    )

  }

}

Different between class component functional component

|  |  |
| --- | --- |
| Functional  Simple function receiving prop and returning declaration  Use functional component as much as possible  Absence of ‘this’ keyword  Solution without using state  Mainly responsible for the ui  That is also called as stateless component dumb component presentational component | Class  More feature rich  Maintaine their own private data – state  They can contain complex ui logic  They provide lifecycle hooks  This component also known as  Statefull smart and container component |

Jsx

Java script xml(jsx) extension to the javascript language syntax

Write xml like code for element and component

Jsx tags have tag name attributes and children

Jsx is not necessity to write React application

Jsx makes react code simpler and elegant

Jsx ultimately transpile to pure javascript which understood by the browser

With jsx

import React from 'react'

export default function Test() {

  return (

    <div><h1>Test</h1></div>

  )

}

Without jsx

import React from 'react'

export default function Test() {

  return react.createElement('div',

  {id: 'hello', className: 'dummy class'},

  react.createElement('h1',null,'hello world'))

}

Props

App.js file

function App() {

  return (

    <>

   <div>

     <Test name="mangal" heroname="punjabiman"> this is fake </ Test>

     <Test name="rakesh" heroname="delhiman" />

     <Test name="dola" heroname="machoman" />

   </div>

    </>

  );

}

Test.js

export default function Test(props) {

  return (

    <>

    <div>name is: {props.name} aka {props.heroname}</div>

    {props.children}

    </>

  )

}

In class Component

E xport default class Test extends Component {

  render() {

    return (

      <>

      <div>Welcom: {this.props.name} aka {this.props.heroname}</div>

      {this.props.children}

      </>

    )

  }

}

Props vs state

|  |  |
| --- | --- |
| Props  Props get passed to the component  Function parameter  Props are immutable  Props in functional component  This.props in class component | State  State is manage within component  Variable declare in the function body  State can be change  Usestate hook – functional component  This.state – class component |

States

State in class component

export default class Test extends Component {

constructor(){

  super()

  this.state = {

    Message: 'welcome Visitors'

  }

}

changemessage(){

  this.setState({

    Message: 'thank you for subscribint'

  })

}

  render() {

    return (

      <>

      <h1>

        {this.state.Message}

      </h1>

      <button onClick={() => this.changemessage()}>Subscribe</button>

      </>

    )

  }

}

Increment

import React, { Component } from 'react'

export default class Test extends Component {

constructor(){

  super()

  this.state = {

    Message: 'welcome Visitors',

    Count: 0

  }

}

changemessage(){

  this.setState({

    Message: 'thank you for subscribint'

  })

  this.setState({

    Count: this.state.Count+1

  },() => {console.log('call back', this.state.Count)})

  console.log('old call back',this.state.Count)

}

  render() {

    return (

      <>

      <h1>

        {this.state.Message}

      </h1>

      <button onClick={() => this.changemessage()}>Subscribe {this.state.Count}</button>

      </>

    )

  }

}

Prevstate  
  
export default class Test extends Component {

constructor(){

  super()

  this.state = {

    Count: 0

  }

}

increment(){

 this.setState((prevState) => ({

   Count: prevState.Count +1

 }))

}

incrementfive(){

  this.increment()

  this.increment()

  this.increment()

  this.increment()

  this.increment()

 }

  render() {

    return (

      <>

      <h1>

      </h1>

      <button onClick={() => this.incrementfive()}>Subscribe {this.state.Count}</button>

      </>

    )

  }

}

prevState with props

export default class Test extends Component {

constructor(){

  super()

  this.state = {

    Count: 0

  }

}

increment(){

 this.setState((prevState, props) => ({

   Count: prevState.Count + this.props.number

 },() => {console.log('call back ', this.state.Count)}

))

}

 render() {

    return (

      <>

      <h1>

      </h1>

      <button onClick={() => this.increment()}>Subscribe {this.state.Count}</button>

      </>

    )

  }

}

always make make use of setState and never modify the state directly

code has to be excuted after the state has been updated place that code in call back function which is the second argument to the setstate method

when you have to update state based on the previous state value pass in a function as an argument instead of the regular object

destructing props and state

destructure in functional component   
  
1stway   
import React from 'react'

function Test({name,heroname}) {

  return (

    <div>{name} {heroname}</div>

  )

}

export default Test

2nd way

function Test(props) {

  const {name ,heroname } = props

  return (

    <div>{name} {heroname}</div>

  )

}

export default Test

Destructure in class component

export default class Test extends Component {

  render() {

    const {name, heroname} = this.props

    //const {state1, state2} = this.state

    return (

      <div>{name},{heroname}</div>

    )

  }

}

Event handeling

function Test() {

  function clickhandeler(){

    console.log("this is mangal");

  }

  return (

    <button onClick={clickhandeler}>click</button>

  )

}

export default Test

with class component

export default class clickme extends Component {

  clickme(){

    console.log("class component ")

  }

  render() {

    return (

      <button onClick={this.clickme}>clickme</button>

    )

  }

}

EventBind

|  |  |
| --- | --- |
| 1st  Declare in render  export default class clickme extends Component {    constructor(props) {      super(props)        this.state = {         Message: "hello"      }    }    clickme(){      this.setState({        Message: "goodbye"      })      }    render() {      return (        <>      <h1>{this.state.Message}</h1>   <button onClick={this.clickme.bind(this)}>clickme</button>        </>        )    }  } | 2nd  Declare in render  export default class clickme extends Component {    constructor(props) {      super(props)        this.state = {         Message: "hello"      }    }    clickme(){      this.setState({        Message: "goodbye"      })      }    render() {      return (        <>      <h1>{this.state.Message}</h1>  <button onClick={() => this.clickme()}>clickme</button>        </>        )    }  } |
| 3rd method  Declare in constructor  import React, { Component } from 'react'  export default class clickme extends Component {    constructor(props) {      super(props)        this.state = {         Message: "hello"      }      this.clickme = this.clickme.bind(this)    }    clickme(){      this.setState({        Message: "goodbye"      })      }    render() {      return (        <>      <h1>{this.state.Message}</h1>          <button onClick={this.clickme}>clickme</button>        </>        )    }  } | 4th method  Arrow function  export default class clickme extends Component {    constructor(props) {      super(props)        this.state = {         Message: "hello"      }    }  clickme = () =>{    this.setState({      Message: "goodbye"    })  }    render() {      return (        <>      <h1>{this.state.Message}</h1>          <button onClick={this.clickme}>clickme</button>        </>        )    }  } |

Method as props

Create a parent.js

Create a child.js

Import the parent in App.js

|  |  |
| --- | --- |
| Child  import React, { Component } from 'react'  export default class Child extends Component {    render() {      return (        <>        <button onClick={this.props.greet} >clickme</button>        </>      )    }  } | Parent  import React, { Component } from 'react'  import Child from './Child'  export default class Parent extends Component {      constructor(props) {        super(props)          this.state = {           ParentName: 'parent'        }      }      greetparent = () =>{          alert(`hell ${this.state.ParentName}`)      }    render() {      return (          <>              <Child greet={this.greetparent} />          </>        )    }  } |

|  |  |
| --- | --- |
| Parameterize  export default class Child extends Component {    render() {      return (        <>        <button onClick={() => this.props.greet("from child")} >clickme</button>        </>      )    }  } | parameterizes  export default class Parent extends Component {      constructor(props) {        super(props)          this.state = {           ParentName: 'parent'        }      }      greetparent = (Childname) =>{          alert(`hell ${this.state.ParentName}${Childname}`)      }    render() {      return (          <>              <Child greet={this.greetparent} />          </>      )    }  } |

Conditional render

|  |  |
| --- | --- |
| 1st  If else  export default class Parent extends Component {  constructor(props) {    super(props)    this.state = {       isLoggedIn:true    }  }    render() {      if(this.state.isLoggedIn){          return (              <h1>hellloo mangal</h1>          )      }      else{          return (              <h1>hello guest</h1>          )      }      }  } | 2nd  variable  export default class Parent extends Component {  constructor(props) {    super(props)    this.state = {       isLoggedIn:true    }  }    render() {      let message      if(this.state.isLoggedIn){          message = <h1>hello mangal</h1>        }      else{          message = <h1>helloo guest</h1>      }     return message;    }  } |
| Ternary conditional operator can work with jsx  export default class Parent extends Component {  constructor(props) {    super(props)    this.state = {       isLoggedIn:true    }  }    render() {  return this.state.isLoggedIn ? (      <h1>hello mangal</h1>  ): (      <h1>Hellooo guest</h1>  )    }  } | Short circuit operator  This method gives you something or nothing  export default class Parent extends Component {  constructor(props) {    super(props)    this.state = {       isLoggedIn:true    }  }    render() {  return this.state.isLoggedIn && <h1>helllo mangal</h1>    }  } |

List rendering

|  |  |
| --- | --- |
| 1st  export default class Parent extends Component {  render() {      const names = ['mangal','rahul', 'raju']  return(      <>      <h1>{names[0]}</h1>      <h1>{names[1]}</h1>      <h1>{names[2]}</h1>      </>  )    }  } | 2nd Map method  export default class Parent extends Component {  render() {      const names = ['mangal','rahul', 'raju']  return(      <>      {names.map(name => <h1>{name}</h1>)}      <h1>{names[0]}</h1>        </>  )    }  } |

|  |  |
| --- | --- |
| 3rd  export default class Parent extends Component {  render() {      const names = ['mangal','rahul', 'raju']      const namelist =  names.map(name => <h1>{name}</h1>)  return(      <>      {namelist}      </>  )    }  } | function Child(){    const person = [{      id: 1,      name: "mangal",      age: 21,      skill:'react'    },    {      id: 2,      name: "rahul",      age: 23,      skill:'angular'    },    {      id: 3,      name: "rakes",      age: 20,      skill:'.net'    },    {      id: 4,      name: "vishnu",      age: 25,      skill:'vue'    }    ]      const personlist = person.map(person => (  <h1>{person.name}{person.age}{person.skill}</h1>      ))      return (        <>         {personlist}        </>      )    }  export default Child  } |

Create a other file for person data

And import as props to person

Then import a person file to app.js

|  |  |
| --- | --- |
| Person file  function Person(){    const persons = [{      id: 1,      name: "mangal",      age: 21,      skill:'react'    },    {      id: 2,      name: "rahul",      age: 23,      skill:'angular'    },    {      id: 3,      name: "rakes",      age: 20,      skill:'.net'    },    {      id: 4,      name: "vishnu",      age: 25,      skill:'vue'    }    ]  const personlist = persons.map(person => (        <List person = {person} />      ))      return (        <>         {personlist}        </>      )    }  export default Person | List file  import React from 'react'  function List({person}) {    return (      <div>{person.name}{person.age}{person.skill}</div>    )  }  export default List |

Unique key

|  |  |
| --- | --- |
| import React, { Component } from 'react'  import List from './List'  function Person(){    const persons = [{      id: 1,      name: "mangal",      age: 21,      skill:'react'    },    {      id: 2,      name: "rahul",      age: 23,      skill:'angular'    },    {      id: 3,      name: "rakes",      age: 20,      skill:'.net'    },    {      id: 4,      name: "vishnu",      age: 25,      skill:'vue'    }    ]  const personlist = persons.map(person => (        <List key={person.id} person = {person} />      ))      return (        <>         {personlist}        </>      )    }  export default Person | You can use a array index as key as well  function Person(){    const name = ['mangal','singh','rahul']    const personlist = name.map((name,index) => (      <h1>        {index}        {name}      </h1>    ))      return (        <>         {personlist}        </>      )    }  export default Person |

Style react component

Css stylesheet

Inline styling

Css modules

Css in js liabrary

|  |  |
| --- | --- |
| Create myfile.css  And import it on stylesheet.js  Conditional styling  Props <Person primary = {true} />  function Person(props) {    let className = props.primary ? 'primary': 'person'    return (       <h1 className={`${className} font-xl`}>Person</h1>    )  export default Person | Inline styling  const heading = {  fontSize: '200px',  color: 'red'  }  function Person(props) {    let className = props.primary ? 'primary': 'person'    return (      <h1 style={heading}>Person</h1>    )  }  export default Person |

Module css

Create a file with name style.module.css

import React from 'react'

import style from'./myfile.module.css' this is module.css is imported

function Person(props) {

  return (

    <>

      <h1 className={style.person}>Person</h1> how its used

    <h1 className={style.primary}>primary</h1>

    <h1 className={style.font\_xl}>font-xl</h1>

    </>

  )

}

Basic form handling

Controlled component

|  |  |
| --- | --- |
| import React, { Component } from 'react'  export default class Person extends Component {    constructor(props) {      super(props)        this.state = {         Username: ''      }    }    changehandel = (event) => {      this.setState({        Username: event.target.value      },() => {console.log(this.state.Username)})    }    render() {      return (        <form>          <label>username</label>          <input type='text' value={this.state.Username} onChange={this.changehandel}/>        </form>      )    }  } | import React, { Component } from 'react'  export default class Person extends Component {    constructor(props) {      super(props)        this.state = {         Username: '',         comment: '',         skill: ''      }    }    changehandel = (event) => {      this.setState({        Username: event.target.value,          },() => {console.log(`username:${this.state.Username}`)})    }    changecomment = event =>{      this.setState({        comment:event.target.value      },() => {console.log(this.state.comment)})    }    changeskill = event =>{      this.setState({        skill:event.target.value      },() => {console.log(this.state.skill)})    }  submitchanges = e => {    alert(`${this.state.Username}and${this.state.comment}and ${this.state.skill}`)    e.preventDefault()  }    render() {      const {Username,comment,skill} = this.state      return (        <form onSubmit={this.submitchanges}>          <label>username</label>          <input type='text' value={Username} onChange={this.changehandel}/>          <br />          <textarea value={comment} onChange={this.changecomment}></textarea>          <br />          <select value={skill} onChange={this.changeskill}>            <option value="react">react</option>            <option value="angular">angular</option>            <option value="vue">vue</option>          </select>          <button type='submit'>submit</button>        </form>      )    }  } |

LifeCycle Method

Life cycle method have 4 faces

Mounting - when an instance of a component is being create and inserted on dom

Their have 4 methods constructor static getderivedFromprops render and component did mount

Updating – when component is being re-render as a result of changes to their its props

Their have 5 methods static getderivedfromprops, shouldcomponentupdate,render,getsnapshotbeforeupdate,and componentdidupdate

Unmounting – when a component is being removed from a dom

There is only one method componentwillunmount

Error handling – when there is an error during rendering , in a lifecycle method, or in the constructor of any child component

Their have 2 methods

Staticgetderivedstatefromerror and componentdidcatch

|  |  |
| --- | --- |
| Mounting lifecycle method  CONSTRUCTOR  A special function that will get call whenever a new component is create  Initial state binding the event handeler  Do not cause the side effect http  Super(props)  Directly overwrite this.state |  |

Pure Component

Create Three file

Parentcomponent.js

RegularCom.js

PureCom.js

Import a parent component in App.js

|  |  |  |
| --- | --- | --- |
| Parent  import React, { Component } from 'react'  import Pure from './Pure'  import Reg from './Reg'  export default class Parent extends Component {    constructor(props) {      super(props)        this.state = {         name: "mangal"      }    }    componentDidMount(){      setInterval(() =>{        this.setState({          name: "rahul==="        })      },2000)    }    render() {      console.log("parent Component render \*\*\*\*\*\*\*\*\*\*\*")      return (        <div>          Parent Compnent      <Reg name = {this.state.name} />      <Pure name={this.state.name} />        </div>      )    }  } | Pure  import React, { PureComponent } from 'react'  export default class Pure extends PureComponent {    render() {     console.log("Pure copoent \*\*\*\*\*\*\*")      return (        <div>Pure          {this.props.name}</div>      )    }  } | Regular  import React, { Component } from 'react'  export default class Reg extends Component {    render() {       console.log("reg component \*\*\*\*\*\*\*\*\*\*\*\*\*")      return (        <div>Reg {this.props.name}</div>      )    }  } |

We can create a component by extending the pure Component class

Pure component implement the shouldComponentUpdate lifecycle method by performing a shallow comparison on the props and state of the component

If there is no difference the component is not re-render – performance boost

It is good idea to ensure that all the children component are also to avoid unexpected behavior

Pure component never mutates the state . always return a new object that reflect the new state

Memo Comp

|  |  |
| --- | --- |
| Memo.js  import react from 'react'  function Memo({name}) {      console.log("\*\*\*\*\*\*\*using memo\*\*\*\*\*\*\*\*\*\*\*")    return (      <div>{name}</div>    )  }  export default react.memo(Memo)  similar to pure component  but it’s a functional component | Parent.js  import React, { Component } from 'react'  import Memo from './Memo'  import Pure from './Pure'  import Reg from './Reg'  export default class Parent extends Component {    constructor(props) {      super(props)        this.state = {         name: "mangal"      }    }    componentDidMount(){      setInterval(() =>{        this.setState({          name: "mangal"        })      },2000)    }    render() {      console.log("parent Component render \*\*\*\*\*\*\*\*\*\*\*")      return (        <div>          Parent Compnent          <Memo name = {this.state.name} />        </div>      )    }  } |

Ref dom element

Ref is mainly use for focus on input field

|  |  |
| --- | --- |
| ref  import react from 'react'  import React, { Component } from 'react'  export default class Parent extends Component {    constructor(props) {      super(props)      this.state ={        name: ''      }    this.inputref = React.createRef()    }      componentDidMount(){      this.inputref.current.focus();      console.log(this.inputref);    }    clickEvent = () =>{      this.setState({        name: this.inputref.current.value      })    }    render() {      return (        <div>          <input type='text' ref={this.inputref}/>          <div>{this.state.name}</div>          <button onClick={this.clickEvent}>submit</button>        </div>      )    }  } | Call back ref  import react from 'react'  import React, { Component } from 'react'  export default class Parent extends Component {    constructor(props) {      super(props)      this.state ={        name: ''      }  this.callbackRef = null;  this.setcallbackref = Element =>{    this.callbackRef = Element;  }    }      componentDidMount(){      if(this.callbackRef){        this.callbackRef.focus();      }    }    clickEvent = () =>{      this.setState({        name: this.callbackRef.value      })    }    render() {      return (        <div>          <input type='text' ref={this.setcallbackref}/>          <div>{this.state.name}</div>          <button onClick={this.clickEvent}>submit</button>        </div>      )    }  } |

Ref with class focus with button click

Create two component

|  |  |
| --- | --- |
| Parent.js  import React, { Component } from 'react'  import Memo from './Memo'  export default class Parent extends Component {    constructor(props) {      super(props)        this.compoentRef = React.createRef();      }  clickHandler =() =>{    this.compoentRef.current.focusInput();  }    render() {      return (        <div>          <Memo ref={this.compoentRef} />          <button onClick={this.clickHandler}>focus input</button>        </div>      )    }  } | Input.js  import React, { Component } from 'react'  export default class Input extends Component {    constructor(props) {      super(props)    this.inputref = React.createRef()    }    focusInput(){      this.inputref.current.focus();    }    render() {      return (        <div>        <input type='text' ref = {this.inputref} />        </div>      )    }  } |

Forwarding refs

Forwarding refs is technique automatically passing a ref through a component to one of its children

|  |  |
| --- | --- |
| Parent  import React, { PureComponent } from 'react'  import Memo from './Memo';  export default class Parent extends PureComponent {    constructor(props) {      super(props)    this.inputref = React.createRef();      }    clickHandler = () =>{      this.inputref.current.focus()    }    render() {      return (        <div>          <Memo ref={this.inputref} />          <button onClick={this.clickHandler}>submit</button>        </div>      )    }  } | Child  import React from 'react'  const memo = React.forwardRef((props , ref) =>{    return (      <div>        <input type='text' ref={ref} />      </div>    )  })  export default memo |

Portals in react

Error boundary || Error handling

A class that implement either one or both of the lifecycle methods getderivedfromerror or componentdidcatch becomes an error boundary

|  |  |  |
| --- | --- | --- |
| App.js  import React from "react"  import ErrorBoundry from "./Component/ErrorBoundry";  import Hero from "./Component/Hero";  function App() {    return (      <>     <div>       <ErrorBoundry>    <Hero heroname='batman'/>    </ ErrorBoundry>    <ErrorBoundry>    <Hero heroname='superman'/>    </ ErrorBoundry>    <ErrorBoundry>    <Hero heroname='joker'/>    </ ErrorBoundry>         </div>      </>    );  }  export default App; | Errorboundary.js  import React, { Component } from 'react'  export default class ErrorBoundry extends Component {      constructor(props) {        super(props)          this.state = {           hasError: false        }      }      static getDerivedStateFromError(error){          return{              hasError: true          }      }  componentDidCatch(error,info){          console.log(error)          console.log(info)      }    render() {      if(this.state.hasError){          return <h1>Something Went Wrong</h1>      }      return this.props.children      }  } | Hero.js  import React from 'react'  function Hero({heroname}) {      if(heroname === 'joker'){          throw new Error('not a hero');      }    return (      <div>{heroname}</div>    )  }  export default Hero |

Error boundaries are react Coponent that catch javascript error in their child component tree, log those error and display a fall-back ui

A class component becomes an error boundry by defining either or both of getderivedstatefromerror and componentdidcatch lifecycle method

Provide a way to Graceful handle error in application code

Higher order Component HOC

A pattern where function takes a component as an argument and return a new component

ClickCounter.js and HoverCounter.js import in app.js

|  |  |  |
| --- | --- | --- |
| With Counter  import React from 'react'  const  UpdatedComponent =(OriginalComponent) => {      class NewComponent extends React.Component{          render(){              return(                  <>                  <OriginalComponent name='mangal'/>                  </>              )          }      }      return NewComponent    }  export default UpdatedComponent | ClickCounter.js  import React, { Component } from 'react'  import UpdatedComponent from './WithCounter'   class ClickCounter extends Component {      constructor(props) {        super(props)          this.state = {           count: 0        }      }      incrementCount = ()=>{          this.setState(prevState => {          return {count:prevState.count+1}          })      }    render() {        const {count} = this.state      return (        <button onClick={this.incrementCount}>{this.props.name}click{count}</button>      )    }  }  export default UpdatedComponent(ClickCounter) | HoverCounter.js  import React, { Component } from 'react'  import UpdatedComponent from './WithCounter'   class HoverCounter extends Component {      constructor(props) {        super(props)          this.state = {           count: 0        }      }      incrementCount = () =>{          this.setState(prevState =>{              return {count: prevState.count+1}          })      }    render() {        const{count} = this.state      return (        <div onMouseOver={this.incrementCount}>{this.props.name}HoverCounter {count}</div>      )    }  }  export default UpdatedComponent(HoverCounter) |

2nd way

|  |  |  |
| --- | --- | --- |
| import React from 'react'  const  UpdatedComponent =(OriginalComponent) => {      class NewComponent extends React.Component{          constructor(props) {              super(props)                this.state = {                 count: 0              }            }            incrementCount = ()=>{                this.setState(prevState => {                return {count:prevState.count+1}                })            }          render(){              return(                  <>                  <OriginalComponent count={this.state.count} increment={this.incrementCount}/>                  </>              )          }      }      return NewComponent    }  export default UpdatedComponent | import React, { Component } from 'react'  import UpdatedComponent from './WithCounter'   class ClickCounter extends Component {      render() {        const {count,increment} = this.props      return (        <button onClick={increment}>click{count}</button>      )    }  }  export default UpdatedComponent(ClickCounter) | import React, { Component } from 'react'  import UpdatedComponent from './WithCounter'   class HoverCounter extends Component {      render() {        const{count,increment} = this.props      return (        <div onMouseOver={increment}>HoverCounter {count}</div>      )    }  }  export default UpdatedComponent(HoverCounter) |

The props passed on hoc not in component

<ClickCounter name='mangal' />

<HoverCounter sname='singh' />

|  |  |  |
| --- | --- | --- |
| import React from 'react'  const  UpdatedComponent =(OriginalComponent,numbers) => {      class NewComponent extends React.Component{          constructor(props) {              super(props)                this.state = {                 count: 0              }            }            incrementCount = ()=>{                this.setState(prevState => {                return {count:prevState.count+numbers}                })            }          render(){                return(                  <>                  <OriginalComponent count={this.state.count} increment={this.incrementCount}{...this.props}/>                  </>              )          }      }      return NewComponent    }  export default UpdatedComponent | import React, { Component } from 'react'  import UpdatedComponent from './WithCounter'   class ClickCounter extends Component {      render() {        const {count,increment} = this.props      return (        <button onClick={increment}>{this.props.name}click{count}</button>      )    }  }  export default UpdatedComponent(ClickCounter,5) | import React, { Component } from 'react'  import UpdatedComponent from './WithCounter'   class HoverCounter extends Component {      render() {        const{count,increment} = this.props      return (        <div onMouseOver={increment}>{this.props.sname}HoverCounter {count}</div>      )    }  }  export default UpdatedComponent(HoverCounter,10) |

Render Props pattern

The term render prop refers to technique for sharing code

Between React Component using a prop whose value is function

|  |  |
| --- | --- |
| App.js  function App() {    return (      <>     <div>    <Counter render={(count,incrementcount) =>{      return(        <>       <ClickCountertwo count={count} incrementCount={incrementcount} />        </>)}} />    <Counter render={(count,incrementcount) =>{      return(        <>       <HoverCountertwo count={count} incrementCount={incrementcount} />  </>)    }} />     </div>      </>    );  } | Counter.js  import React, { Component } from 'react'   class Counter extends Component {       constructor(props) {         super(props)           this.state = {            count: 0         }       }       incrementcount = () =>{           this.setState(preState =>{             return{count: preState.count+1}           })       }    render() {      return (        <div>{this.props.render(this.state.count, this.incrementcount)}</div>      )    }  }  export default Counter |
| Clickcounter.js  import React, { Component } from 'react'  export default class ClickCountertwo extends Component {    render() {        const{count, incrementCount} = this.props      return (        <div>            <button onClick={incrementCount}>click{count}</button>        </div>      )    }  } | HoverCounter.js  import React, { Component } from 'react'  export default class HoverCountertwo extends Component {    render() {      const{count, incrementCount} = this.props      return (        <div onMouseOver={incrementCount}>HoverCountertwo{count}</div>      )    }  } |

Context

Context provide a way to pass data through the component tree without having to pass props down manually at every level

|  |  |
| --- | --- |
| UserContext.js  import React from "react";  const UserContext = React.createContext()  const UserProvider = UserContext.Provider  const UserConsumer = UserContext.Consumer  export {UserContext , UserProvider , UserConsumer} | App.js  import React from "react"  import CompC from "./Component/CompC";  import { UserProvider } from "./Component/UserContext";  function App() {    return (      <>     <div>       <UserProvider value="mangal">         <CompC />       </UserProvider>     </div>      </>    );  }  export default App; |
| CompA.js  import React, { Component } from 'react'  import CompB from './CompB'   class CompA extends Component {    render() {      return (        <div><CompB /></div>      )    }  }  export default CompA | ComB.js  import React, { Component } from 'react'  import CompC from './CompC'   class CompB extends Component {    static contextType = UserContext;    render() {      return (        <div><CompC /></div>      )    }  }  //CompB.contextType = UserContext;  export default CompB |
| CompC.js  import React, { Component } from 'react'  import { UserConsumer } from './UserContext'   class CompC extends Component {    render() {      return (        <UserConsumer>            {(username) =>{                return(                    <>                    <h1>{username}</h1>                    </>                )            }}        </UserConsumer>      )    }  }  export default CompC |  |

Http And React

|  |  |
| --- | --- |
| getdata  import axios from 'axios'  import React, { Component } from 'react'  export default class Crud extends Component {      constructor(props) {        super(props)          this.state = {           Post:[]        }      }      async componentDidMount(){         await axios.get ('/api/Employees/')          .then(Response =>{              console.log(Response)              this.setState({Post: Response.data})          })          .catch(error =>{              console.log(error)          })      }        render() {        const {Post} = this.state      return (        <div>{          Post.map(post =>{                return(                    <>                   <h1 key={post.id}>Name:{post.Name} and age:{post.age}</h1>                    </>                )            })            }</div>      )    }  } | import React, { Component } from 'react'  import Axios from 'axios'   class Post extends Component {       constructor(props) {         super(props)        this.state = {            Name:'',            age:''         }       }       Handlechange = (e) =>{           this.setState({[e.target.name]:e.target.value})       }       submithandle =(e) =>{  e.preventDefault();  console.log(this.state)  Axios.post('https://localhost:44344/api/Home',{      Name:this.state.Name,      age: parseInt(this.state.age)  }).then(Response =>{      console.log(Response.data)  }).catch(error => {      console.log(error)  })       }    render() {        const {Name, age} = this.state      return (        <div>            <form onSubmit={this.submithandle}>                <div>                    <input type='text' placeholder='name' name='Name' value={Name} onChange={this.Handlechange} />                </div>                <div>                    <input type='number' placeholder='name' name='age' value={age} onChange={this.Handlechange}/>                </div>                <button type='submit' >Submit</button>            </form>        </div>      )    }  }  export default Post |

Hooks

Hooks allow you to use react feature without having to write a class

Avoid whole confusion with this

Allow you to reuse stateful logic

State use in function component

|  |  |
| --- | --- |
| Counter  import React, { useState } from 'react'  function Post() {      const [count,setCount] = useState(0)    return (      <div onClick={() => setCount(count+1)} >this{count}</div>    )  }  export default Post | Prevcount  import React, { useState } from 'react'  function Post() {      const initial = 0      const [count,setCount] = useState(initial)      const incrementby5 =()=>{          for(let i = 0; i<5;i++){              setCount(pre => pre+1)          }        }       return (      <>      <h1>Count:{count}</h1>      <button onClick={() => setCount(count+1)}>Increment</button>      <button onClick={() =>setCount(count-1)}>Decrement</button>      <button onClick={() => setCount(initial)}>Reset</button>      <button onClick={incrementby5}>increment5</button>      </>    )  }  export default Post |

|  |  |
| --- | --- |
| Use state with object  import React, { useState } from 'react'  function Post() {      const [name,setName] = useState({          firstname:'',          lastname:''      })    return (        <>        <input type='text' value={name.firstname} onChange={e => setName({...name,firstname: e.target.value})} />      <input type='text' value={name.lastname} onChange={e => setName({...name,lastname: e.target.value})} />      <h1>your first name is {name.firstname}</h1>      <h1>your lastname name is {name.lastname}</h1>      <h1>{JSON.stringify(name)}</h1>        </>      )  }  export default Post | Use state with array  import React, { useState } from 'react'  function Post() {      const [item, setItem] = useState([]);      const addItem =() =>{          setItem([...item,{              id: item.length,              value: Math.floor(Math.random()\*10)+1          }            ])      }    return (      <div>          <button onClick={addItem}>add</button>          <ul>              {item.map(items =>(                  <li key={items.id}>{items.value}</li>              ))}          </ul>      </div>    )  }  export default Post |

UseEffect Hook

It is a close replacement for componentdidmount and componentdidupdate

And componentwillunmount

|  |  |
| --- | --- |
| Useeffect hooks after render  import React, { useEffect, useState } from 'react'  function Post() {      const [count, setCount] = useState(0);      useEffect(() =>{          document.title = `total count${count}`      })    return (      <div onClick={() => setCount(count+1)}>Count:{count}</div>    )  }  export default Post | Fetch data  import axios from 'axios';  import React, { useEffect, useState } from 'react'  function Post() {      const [posts, setPost] = useState([]);      useEffect(() =>{          axios.get("https://localhost:44344/api/Home/1").then(res =>{              console.log(res.data)              setPost([res.data])          }).catch(error =>{              console.log(error)          })      },[])      return (      <div><ul>          {posts.map(po => (          <li key={po.id}>{po.name} and {po.age}</li>      ))}      </ul>        </div>    )  }  export default Post |

|  |  |
| --- | --- |
| import React, { useEffect, useState } from 'react'  function Post() {      const [name, setName] = useState('post')      useEffect(() =>{          console.log("component did mount")      },[])      useEffect(() =>{          console.log("component did update")          return () =>{              console.log("unmount")          }        },[name])    return (      <div>          <button onClick={() => setName("user")}>user</button>          <button onClick={() => setName("employees")}>employees</button>          <button onClick={() => setName("customer")}>customer</button>          <h1>{name}</h1>        </div>    )  }  export default Post |  |

Usecontext

|  |  |
| --- | --- |
| import Post from "./Component/Post";  import React, { createContext } from 'react';  const name = createContext()  function App() {      return (      <div className="App">        <name.Provider value="mangal">        <Post />        </name.Provider>        </div>    );  }  export default App;  export {name} | import React, {useContext } from 'react'  import{ name } from '../App'  function Post() {   const Name = useContext(name)    return (      <div>{Name}</div>    )  }  export default Post |

Usereducer

import React, { useReducer } from 'react'

const initial = {count:1}

const reducer = (state,action) =>{

console.log(state,action)

switch(action.type){

    case "in":

        return{count:state.count+1}

        case "dec":

            return {count:state.count-1}

            default:

                return  state

}

}

function Post() {

    const [state, dispatch] = useReducer(reducer, initial)

  return (

    <div>

        <button onClick={() => dispatch({type:"in"}) }>+</button>

        {state.count}

        <button onClick={() => dispatch({type:"dec"}) }>-</button>

    </div>

  )

}

export default Post

Firebase OTP

Firebase.js

// Import the functions you need from the SDKs you need

import { initializeApp } from "firebase/app";

import { getAuth } from "firebase/auth"

// TODO: Add SDKs for Firebase products that you want to use

// https://firebase.google.com/docs/web/setup#available-libraries

// Your web app's Firebase configuration

const firebaseConfig = {

  apiKey: "AIzaSyBSoAx7pCkWenjBUQmllQ6NPc4S1KTGtnM",

  authDomain: "otp-8dc45.firebaseapp.com",

  projectId: "otp-8dc45",

  storageBucket: "otp-8dc45.appspot.com",

  messagingSenderId: "333558127881",

  appId: "1:333558127881:web:1b49e01d03507808e247d8"

};

// Initialize Firebase

const app = initializeApp(firebaseConfig);

export const authentication = getAuth(app);

Form.js

import React,{useState} from 'react'

import {authentication } from "./Firebaseotp"

import { RecaptchaVerifier, signInWithPhoneNumber } from "firebase/auth";

function Counter() {

  const code = "+91"

  const [number, setNumber] = useState(code)

  const [otp, setOtp] = useState()

  const generecaptcha = () =>{

    window.recaptchaVerifier = new RecaptchaVerifier('sign-in-button', {

      'size': 'invisible',

      'callback': (response) => {

        // reCAPTCHA solved, allow signInWithPhoneNumber.

      }

    }, authentication);

  }

  const requestotp = (e) =>{

e.preventDefault();

if(number.length >= 10){

  generecaptcha();

  let appverify = window.recaptchaVerifier

  signInWithPhoneNumber(authentication,"+91"+number,appverify)

.then(confirmationResult =>{

  window.confirmationResult = confirmationResult;

}).catch((error) =>{

  console.log(error)

})

}

  }

  const verifyOtp = (e) =>{

    let otp = e.target.value

    setOtp(otp);

    console.log(otp)

    if(otp.length === 6){

      console.log(otp)

      let confirmationResult = window.confirmationResult

      // const code = getCodeFromUserInput();

      confirmationResult.confirm(otp).then((result) => {

  // User signed in successfully.

  const user = result.user;

  // ...

}).catch((error) => {

  // User couldn't sign in (bad verification code?)

  // ...

});

    }

  }

  return (

    <>

    <form onSubmit={requestotp}>

      <input type='number' name="number" placeholder="numbe" value={number} onChange={(e) => setNumber(e.target.value)}/>

      <button type='submit'>submit</button>

    </form>

    <form onSubmit={requestotp}>

      <input type='number' name="otp" placeholder="otp" value={otp} onChange={verifyOtp}/>

      <button type='submit'>submit</button>

    </form>

    <div id='sign-in-button'></div>

    </>

  )

}

export default Counter

Styled Component

Install vscode-styled-component

And

Npm install styled-component

|  |  |
| --- | --- |
| import React from 'react'  import styled from 'styled-components'  const S\_button = styled.button`  background-color: green;  color: white;  text-align: center;  cursor: pointer;  `  function Post() {    return (      <div>          <button>Submit</button>        <S\_button>Submit</S\_button>      </div>    )  }  export default Post |  |

|  |  |
| --- | --- |
| import React from 'react'  import {S\_button} from './Button'  function Post() {    return (      <div>          <button>Submit</button>        <S\_button>Submit</S\_button>      </div>    )  }  export default Post | import styled from 'styled-components'  const S\_button =styled.button`  background-color: green;  color: white;  text-align: center;  cursor: pointer;  `  export {S\_button} |

Styled component using props

|  |  |
| --- | --- |
| import React from 'react'  import {S\_button} from './Button'  function Post() {    return (      <div>          <button>Submit</button>        <S\_button varient="outline">Submit</S\_button>      </div>    )  }  export default Post | import styled from 'styled-components'  const S\_button =styled.button`  background-color: ${(props) => props.varient === 'outline' ? 'green': 'black' };  color: white;  text-align: center;  cursor: pointer;  `  export {S\_button} |

Extendting styles

import styled from 'styled-components'

export const S\_button =styled.button`

background-color: ${(props) => props.varient === 'outline' ? 'green': 'black' };

color: white;

text-align: center;

cursor: pointer;

`

export const Fancybutton = styled(S\_button)` //extended

padding-left: 10px;

padding-right: 10px;

margin: 20px;

height: 20px;

width: 100px;

position: relative;

   <Fancybutton as="a">submit</Fancybutton> // for ancher tag

Pseudo classes

export const Fancybutton = styled(S\_button)`

padding-left: 10px;

padding-right: 10px;

margin: 20px;

height: 20px;

width: 100px;

position: absolute;

&:hover{

    margin: 40px;

    color: red;

}

`

Passed props and attributes

export const InputStyle = styled.input.attrs({

    type:"text",

    name:"MobileNumber",

    placeholder:'number'

})`

background-color: red;

text-align: center;

&:focus{

    background-color: yellow;

    color: red;

}

Adding props

export const InputStyle = styled.input.attrs((props) => ({

    type:"text",

    name:"MobileNumber",

    placeholder:'number'

}))`

background-color: red;

text-align: center;

&:focus{

    background-color: yellow;

    color: red;

}

`

Animation

export const rotate = keyframes`

from{

    transform: rotate(0deg);

}

to{

    transform: rotate(360deg);

}

`

export const StyleDiv = styled.div`

position: relative;

height: 20px;

width: 20px;

background-color: blue;

animation: ${rotate} infinite 2s linear;

`

Global styled

` import styled,{keyframes,createGlobalStyle} from 'styled-components'

export const Global = createGlobalStyle`

button{

    font-family: 'Roboto';

}

MaterialUI

Button

import React, { useState } from 'react'

import {Typography,Button} from '@material-ui/core'

import {Delete} from '@material-ui/icons'

import { isDisabled } from '@testing-library/user-event/dist/utils'

const Post = () => {

  const [btnDisable, setBtnDisable] = useState(false)

   const conchangebtn = () =>{

  setBtnDisable(true)

  }

  return (

    <div>

      <Typography>hii</Typography> endIcon={<Delete />

      <Button variant='contained' color='primary' startIcon={<Delete /> } disabled={btnDisable}

      onClick={conchangebtn}

      >submit</Button>

    </div>

  )

}

export default Post

Button Group

 <ButtonGroup color='primary' variant='outlined'>

      <Button  color='primary' startIcon={<Delete /> } disabled={btnDisable}

      onClick={conchangebtn}

      >submit</Button>

      <Button variant='contained' color='primary' startIcon={<Delete /> } disabled={btnDisable}

      onClick={conchangebtn}

      >submit</Button>

      <Button variant='contained' color='primary' startIcon={<Delete /> } disabled={btnDisable}

      onClick={conchangebtn}

      >submit</Button>

      </ButtonGroup>

Checkbox

import React, { useState } from 'react'

import {Checkbox} from '@material-ui/core'

import {Favorite,FavoriteBorder,ViewAgenda,RemoveRedEye,RemoveRedEyeOutlined } from '@material-ui/icons'

function Post() {

  const name = []

  function getValue(e){

var data = name;

data.push(e.target.value)

console.log(data)

data.splice(data,0)

  }

  return (

    <div>

      <Checkbox  value='mangal' name='mangal'  onClick={(e) =>{getValue(e)}}></Checkbox>

      <Checkbox value='rakesh' name='mangal'  onChange={(e) =>{getValue(e)}}></Checkbox>

      <Checkbox value='rahul' name='mangal' onChange={(e) => {getValue(e)}} icon={<RemoveRedEye />}

      checkedIcon={<RemoveRedEyeOutlined/>}

      ></Checkbox>

      <Checkbox value='rahul' onChange={(e) => {getValue(e)}}

      icon={<Favorite/>}

      checkedIcon={<FavoriteBorder/>}

      ></Checkbox>

    </div>

  )

}

export default Post

radiobutton

import React, { useState } from 'react'

import {Radio} from '@material-ui/core'

function Post() {

  const [gender, setGender] = useState("male")

  const testFunction = (e) =>{

console.log(e.target.value)

setGender(e.target.value)

  }

  return (

    <div>

      <div>

      <Radio

      color='primary'

      value='male'

      checked={gender === "male"}

      onChange={ testFunction}

      />

    <span>Male</span>

      </div>

      <div>

      <Radio

      color='secondary'

      value='Female'

      checked={gender === "Female"}

      onChange={ testFunction}

      />

    <span>Female</span>

      </div>

      <div>

      <Radio

      color='primary'

      value='Other'

      checked={gender === "Other"}

      onChange={ testFunction}

      />

    <span>Other</span>

      </div>

    </div>

  )

}

export default Post

Slider

import React from 'react'

import {Slider} from '@material-ui/core'

import { getValue } from '@testing-library/user-event/dist/utils'

function Post() {

  const marks=[

    {

      value:0,

      label:"start"

    },

    {

      value:50,

      label:"Middle"

    },

    {

      value:100,

      label:"End"

    }

  ]

const getValue = (e,val)=>{

console.log(val)

}

  return (

    <div>

      <div style={{height:300,margin:"auto",marginTop:100}}>

      <Slider

      color='secondary'

      defaultValue={100}

      valueLabelDisplay='auto'

      step={10}

      max={200}

      marks={marks}

      onChange={getValue}

      orientation="vertical"

      />

      </div>

    </div>

  )

}

export default Post

2nd method

import React, { useState } from 'react'

import {Slider} from '@material-ui/core'

function Post() {

  const [val, setval] = useState([40,100])

const updateValue = (e,val)=>{

setval(val)

console.log({val})

}

  return (

    <div>

      <div style={{height:300,margin:"auto",marginTop:100}}>

      <Slider

      color='secondary'

      value={val}

      valueLabelDisplay='auto'

      step={10}

      max={100}

      onChange={updateValue}

      orientation="vertical"

      />

      </div>

    </div>

  )

}

export default Post

Select

import React, { useState } from 'react'

import {Select,MenuItem} from '@material-ui/core'

function Post() {

  const [course, setcourse] = useState(3)

  const updatecourse=(e,val)=>{

    console.log(e.target.value)

setcourse(e.target.value)

  }

  return (

    <div>

      <Select

       value={course}

       displayEmpty

       onChange={updatecourse}

       >

        <MenuItem value="">Select Course</MenuItem>

        <MenuItem value={1} >Java</MenuItem>

        <MenuItem value={2} >.Net</MenuItem>

        <MenuItem value={3} >PHP</MenuItem>

        <MenuItem value={4}>NODE</MenuItem>

      </Select>

    </div>

  )

}

export default Post

TextField

import React from 'react'

import {TextField} from '@material-ui/core'

function Post() {

  const getValue =(e)=>{

console.log(e.target.value)

  }

  return (

    <div>

      <TextField

      label="Enter Your Name"

      color='secondary'

      variant='standard'

      onChange={getValue}

      />

    </div>

  )

}

export default Post

Switch

import React from 'react'

import {Switch} from '@material-ui/core'

function Post() {

  const getvalue=(e,val)=>{

console.log(val)

  }

  return (

    <div>

      <Switch

      color='primary'

      size='small'

      onChange={getvalue}

      />

    </div>

  )

}

export default Post

Box

import React from 'react'

import {Box,Button} from '@material-ui/core'

function Post() {

  return (

    <div>

<Box component="div" style={{color:'red'}} clone m={20} p={10}>

<Button>hello</Button>

</Box>

    </div>

  )

}

export default Post

React Router

import React, { createContext } from 'react';

import {Routes, Route} from 'react-router-dom'

import Home from "./Component/Home";

import About from "./Component/About";

import Post from './Component/Post'

function App() {

  return (

    <div className="App">

     <Routes>

       <Route path="/" element={<Home />}></Route>

       <Route path="About" element={<About /> }></Route>

       <Route path="Post" element={<Post />}></Route>

     </Routes>

    </div>

  );

}

export default App;

Link

import React from 'react'

import {Link} from 'react-router-dom'

function Navbar() {

  return (

    <div>

        <nav>

            <Link to='/'>Home</Link>

            <Link to='/About'>About</Link>

            <Link to='/Post'>post</Link>

        </nav>

    </div>

  )

}

export default Navbar

ActiveLink

import React from 'react'

import {NavLink} from 'react-router-dom'

function Navbar() {

  return (

    <div>

        <nav>

            <NavLink to='/'>Home</NavLink>

            <NavLink to='/About'>About</NavLink>

            <NavLink to='/Post'>post</NavLink>

        </nav>

    </div>

  )

}

export default Navbar

nav a.active{

    color: red;

    text-decoration: none;

    font-size: 50px;

}

Navigating programmatically

|  |  |
| --- | --- |
| Home.js  import React from 'react'  import {useNavigate} from 'react-router-dom'  import {Button} from '@material-ui/core'  function Home() {      const navigate = useNavigate();      const navigation =()=>{      navigate('Post')  optional// navigate('Post',{replace: true})      }    return (      <div>            <Button color='secondary' variant='outlined' onClick={navigation}  >Submit</Button>      </div>    )  }  export default Home | Post.js  import React from 'react'  import {useNavigate} from 'react-router-dom'  import {Button} from '@material-ui/core'  function Post() {    const navigate = useNavigate();    const navigation =()=>{        navigate(-1)    }    return (      <div>        <Button color='secondary' variant='outlined' onClick={navigation}  >Goback</Button>      </div>    )  }  export default Post |

No Match Route

|  |  |
| --- | --- |
| App.js  import React, { createContext } from 'react';  import {Routes, Route} from 'react-router-dom'  import Home from "./Component/Home";  import About from "./Component/About";  import Post from './Component/Post'  import Navbar from './Component/Navbar';  import './app.css'  import NoMatch from './Component/NoMatch';  function App() {      return (      <div className="App">       <Routes>         <Route path="/" element={<Home />}></Route>         <Route path="About" element={<About /> }></Route>         <Route path="Post" element={<Post />}></Route>         <Route path='\*' element={<NoMatch />} />       </Routes>       <Navbar />            </div>    );  }  export default App; | NoMatch.js  import React from 'react'  function NoMatch() {    return (      <div>Page Not Found</div>    )  }  export default NoMatch |

Nested router

|  |  |
| --- | --- |
| App.js  import React, { createContext } from 'react';  import {Routes, Route} from 'react-router-dom'  import Home from "./Component/Home";  import About from "./Component/About";  import Post from './Component/Post'  import Navbar from './Component/Navbar';  import './app.css'  import NoMatch from './Component/NoMatch';  import Products from './Component/Products';  import Featured from './Component/Featured';  import New from './Component/New';  function App() {      return (      <div className="App">       <Routes>         <Route path="/" element={<Home />}></Route>         <Route path="About" element={<About /> }></Route>         <Route path="Post" element={<Post />}></Route>         <Route path='\*' element={<NoMatch />} />         <Route path='Products' element={<Products />}>          <Route path='Featured' element={<Featured />} />          <Route path='New' element={<New />} />         </Route>       </Routes>       <Navbar />            </div>    );  }  export default App; | Product.js  import React from 'react'  import {Link, Outlet} from 'react-router-dom'  function Products() {    return (      <div>          <div>          <input type='search' placeholder='search' />          </div>          <nav>              <Link to='featured'>featured</Link>              <Link to='new'>new</Link>          </nav>  <Outlet />      </div>    )  }  export default Products |
| Navbar.js  import React from 'react'  import {NavLink} from 'react-router-dom'  function Navbar() {    return (      <div>          <nav>              <NavLink to='/'>Home</NavLink>              <NavLink to='/About'>About</NavLink>              <NavLink to='/Post'>post</NavLink>              <NavLink to='/Products'>Products</NavLink>          </nav>        </div>    )  }  export default Navbar | Featured.js  import React from 'react'  function Featured() {    return (      <div>Featured</div>    )  }  export default Featured  New.js  import React from 'react'  function New() {    return (      <div>New</div>    )  }  export default New |

Index route

|  |  |
| --- | --- |
| import React, { createContext } from 'react';  import {Routes, Route} from 'react-router-dom'  import Home from "./Component/Home";  import About from "./Component/About";  import Post from './Component/Post'  import Navbar from './Component/Navbar';  import './app.css'  import NoMatch from './Component/NoMatch';  import Products from './Component/Products';  import Featured from './Component/Featured';  import New from './Component/New';  import Indexs from './Component/Indexs';  function App() {      return (      <div className="App">       <Routes>         <Route path="/" element={<Home />}></Route>         <Route path="About" element={<About /> }></Route>         <Route path="Post" element={<Post />}></Route>         <Route path='\*' element={<NoMatch />} />         <Route path='Products' element={<Products />}>           <Route index element={<Featured />}/> //Index route is here          <Route path='Featured' element={<Featured />} />          <Route path='New' element={<New />} />         </Route>       </Routes>       <Navbar />            </div>    );  }  export default App; |  |

Dynamic Routes

 <Route path='Indexs/:userid' element={<UserDetails />}>

       </Route>

       <Route path='Indexs/admin' element={<Admin />}></Route>

It can be nested as well

Url params

import React from 'react'

import {useParams} from 'react-router-dom'

function UserDetails() {

    const params = useParams();//    const {userid} = useParams();

and you can destructured it

    const userid = params.userid // don’t have to write this code

  return (

    <div>UserDetails  {userid}</div>

  )

}

export default UserDetails

something new Array

 {

          Array(rating).fill().map((\_) =>(

            <p>\*</p>

          ))}

Client ID

481625763417-a5ucls8rqt2vtlskp1cs6luaou0sm5gm.apps.googleusercontent.com

Client secret

GOCSPX-V1hTj0KUgjmM\_SdfB8SH7qF85YiH

Auth code 4/0AX4XfWiPrJWIp-46XOo\_VRniRWXPBukWedvdkPu7u0fCKTVF-eG\_q87PqJPPAD-A7f-EZg

Ref 1//04kia9NK0ffymCgYIARAAGAQSNwF-L9IrOKRjFUjLtW7ovMKL8Ec13cuvwufXxsGB1EDuz2udBPe3xQJbKXN81i1AKt6k9aANxTk

Acc ya29.A0ARrdaM-ZgtZ4QN3RomI3DFV\_F4eRQJ0LzXeuVWlofwUgj-FoAUUUTjRKk6MUztE8ZZgJ8XeGYEbNFmBlCSLRQu2Wx2sdZIjo\_tvh\_nSzSgkjoyMRcjaCCWe8Uvt0cDuWDklcEDXsZyL1BUizaB7mjlsF5MsQ