**==========================================================**

[**https://shorturl.at/sMS17**](https://shorturl.at/sMS17)

**SPA CRUD Task**

**React With Redux**

**Thunk Middleware**

**Deploying React Application**

**==========================================================**

==========================================================

SPA CRUD Application

==========================================================

<>

src

Components

- IndexComponent.js Routing

- CreateComponent.js Insert record -> axios post

- ReadComponent.js Fetch records -> axios get

- UpdateComponent.js Update record -> axios post

- DeleteComponent.js Delete record -> axios post

- url.js -> API url

Download axios (API calls)

>yarn add axios --save

Download react-router-dom (SPA)

>yarn add react-router-dom --save

--------OR--------

Combine both

>yarn add axios react-router-dom --save

\*\*\*ReadComponent.js\*\*\*

import React from 'react'

export default class ReadComponent extends React.Component {

constructor() {

super()

this.state = {

products: []

}

}

render() {

return (

<div className='container mt-2'>

<div className='text-warning h1'>I am from Read Component</div>

</div>

)

}

}

Note:- create all components skeleton and check for routing then go with api calls

\*\*\*IndexComponent.js\*\*\*

import React from "react";

import { NavLink, Route, BrowserRouter as Router, Routes } from 'react-router-dom'

import CreateComponent from "./CreateComponent";

import ReadComponent from "./ReadComponent";

import UpdateComponent from "./UpdateComponent";

import DeleteComponent from "./DeleteComponent";

export default class IndexComponent extends React.Component {

render() {

return (

<div>

<div className="nav nav-pills">

<Router>

<div className="nav-item"> <NavLink to="/create" className='nav-link'>Create</NavLink></div>

<div className="nav-item"> <NavLink to="/read" className='nav-link'>Read</NavLink></div>

<div className="nav-item"> <NavLink to="/update" className='nav-link'>Update</NavLink></div>

<div className="nav-item"> <NavLink to="/delete" className='nav-link'>Delete</NavLink></div>

<br /><br />

<Routes>

<Route path="/create" element={<CreateComponent />}></Route>

<Route path="/read" element={<ReadComponent />}></Route>

<Route path="/update" element={<UpdateComponent />}></Route>

<Route path="/delete" element={<DeleteComponent />}></Route>

</Routes>

</Router>

</div>

</div>

)

}

}

\*\*\*index.js\*\*\*

. . .

const root = ReactDOM.createRoot(document.getElementById('root'));

root.render(

<IndexComponent/>

);

. . .

\*\*\*index.html\*\*\*

in head section -> Bootstrap CDN

<link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/4.5.0/css/bootstrap.min.css">

<link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/font-awesome/4.7.0/css/font-awesome.min.css">

<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>

<script src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.16.0/umd/popper.min.js"></script>

<script src="https://maxcdn.bootstrapcdn.com/bootstrap/4.5.0/js/bootstrap.min.js"></script>

—-----------------------------------------------------------------------------------------------------

create url.js

<>

src

- url.js

let url = `https://bewildered-puce-wear.cyclic.app`

export default url

\*\*\*ReadComponent.js\*\*\*

import React from 'react'

import axios from 'axios'

import url from './url'

export default class ReadComponent extends React.Component {

constructor() {

super()

this.state = {

products: []

}

}

componentDidMount() {

this.setState({

status: 'Loading'

})

axios.get(url + '/fetch').then((posRes) => {

this.setState({

products: posRes.data,

status: ''

})

}, (errRes) => {

this.setState({

status: errRes.message

})

console.log(errRes)

})

}

render() {

return (

<div className='container mt-2'>

<div className='text-warning h1'>I am from Read Component</div>

<h1 className='text-danger'>{this.state.status} </h1>

<table className='table table-borderd table-dark table-striped table-hover w-50 mx-auto'>

<thead>

<tr>

<th>Sr No </th>

<th>P\_id</th>

<th>P\_name</th>

<th>P\_cost</th>

</tr>

</thead>

<tbody>

{this.state.products.map((element, index) => (

<tr>

<td>{index + 1} </td>

<td>{element.p\_id}</td>

<td>{element.p\_name}</td>

<td>{element.p\_cost}</td>

</tr>

))}

</tbody>

</table>

</div>

)

}

}

\*\*\*CreateComponent.js\*\*\*

import React from 'react'

import axios from 'axios'

import url from './url'

export default class CreateComponent extends React.Component {

constructor() {

super()

this.state = {

products: []

}

}

render() {

return (

<div className='container mt-2'>

<div className='text-info h1'>I am from Create Component</div>

<div className='w-50 '>

<form onSubmit={this.insert}>

<input type='number'

placeholder='p\_id'

name='p\_id'

className='form-control my-2'></input>

<input type='text'

placeholder='p\_name'

name='p\_name'

className='form-control my-2'></input>

<input type='number'

placeholder='p\_cost'

name='p\_cost'

className='form-control my-2'></input>

<input type='submit' value='Create' className='btn btn-outline-success'></input>

</form>

<h1 className='text-primary'>{this.state.status} </h1>

</div>

</div>

)

}

insert = (e) => {

e.preventDefault()

this.setState({

status: 'Loading'

})

let obj = {

"p\_id": parseInt(e.target.p\_id.value),

"p\_name": e.target.p\_name.value,

"p\_cost": parseInt(e.target.p\_cost.value)

}

axios.post(url + '/insert', obj).then((posRes) => {

console.log(posRes)

this.setState({

status: posRes.data.insert

})

}, (errRes) => {

console.log(errRes)

})

}

}

\*\*\*UpdateComponent.js\*\*\*

import React from 'react'

import axios from 'axios'

import url from './url'

export default class UpdateComponent extends React.Component {

constructor() {

super()

this.state = {

products: []

}

}

render() {

return (

<div className='container mt-2'>

<div className='text-primary h1'>I am from Update Component</div>

<div className='w-50 '>

<form onSubmit={this.update}>

<input type='number'

placeholder='p\_id'

name='p\_id'

className='form-control my-2'></input>

<input type='text'

placeholder='p\_name'

name='p\_name'

className='form-control my-2'></input>

<input type='number'

placeholder='p\_cost'

name='p\_cost'

className='form-control my-2'></input>

<input type='submit' value='Update' className='btn btn-outline-success'></input>

</form>

<h1 className='text-primary'>{this.state.status} </h1>

</div>

</div>

)

}

update = (e) => {

e.preventDefault()

this.setState({

status: 'Loading'

})

let obj = {

"p\_id": parseInt(e.target.p\_id.value),

"p\_name": e.target.p\_name.value,

"p\_cost": parseInt(e.target.p\_cost.value)

}

axios.post(url + '/update', obj)

.then((posRes) => {

console.log(posRes)

this.setState({

status: posRes.data.update

})

}, (errRes) => {

console.log(errRes)

this.setState({

status: JSON.stringify(errRes)

})

})

}

}

\*\*\*DeleteComponent.js\*\*\*

import axios from 'axios'

import React from 'react'

import url from './url'

export default class DeleteComponent extends React.Component {

constructor() {

super()

this.state = {

products: []

}

}

render() {

return (

<div className='container mt-2'>

<div className='text-warning h1'>I am from Delete Component</div>

<div className='w-50 '>

<form onSubmit={this.delete}>

<input type='number'

placeholder='p\_id'

name='p\_id'

className='form-control my-2'></input>

<input type='submit' value='Delete' className='btn btn-outline-success'></input>

</form>

<h1 className='text-primary'>{this.state.status} </h1>

</div>

</div>

)

}

delete = (e) => {

e.preventDefault()

this.setState({

status: 'Loading'

})

let obj = {

p\_id: parseInt(e.target.p\_id.value)

}

axios.post(url + '/delete', obj)

.then((posRes) => {

console.log(posRes)

this.setState({

status: posRes.data.delete

})

}, (errRes) => {

console.log(errRes)

this.setState({

status: errRes.message

})

})

}

}

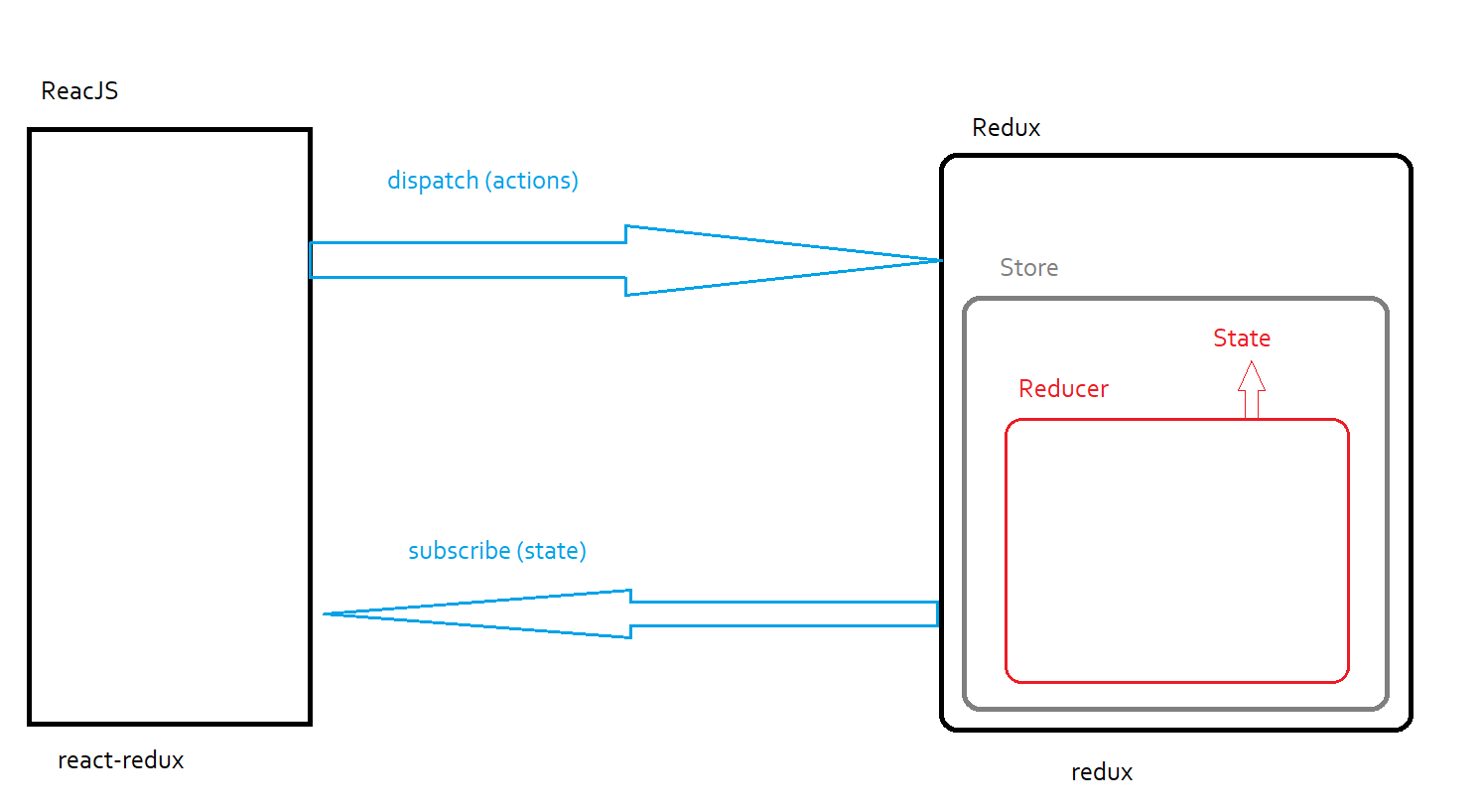
===================================

Redux

===================================

- Redux is used for global state management.

- Redux is working at any layer of UI.



Redux Architecture

- Create a store using the 'redux' library.

- global business logic written in reducer.

- Output of the reducer is state.

- integrate this architecture with any front-end technology, eg ReactJS.

- 'react-redux' library is used to integrate react with redux.

- A request sent by reactjs is called as dispatch.

- dispatch contains various actions.

Eg FETCH,

WITHDRAW,

UPDATE,

DELETE,

DEPOSIT,

...

- response received by reactjs is as subscribe

- subscribe contains state, implies received response is state.

Download libraries

redux

react-redux

>yarn add redux react-redux --save

Directory structure

<>

src

reduxeg

reducer

- reducer.js

- myComponent.js

- index.js

create reducer

\*\*\*reducer.js\*\*\*

const initialState = {

products: []

}

const reducer = (state = initialState, actions) => {

switch (actions.type) {

case 'PRODUCTS':

return {

...state,

products: [

{ "p\_id": 111, "p\_name": "P\_one", "p\_cost": 10000 },

{ "p\_id": 222, "p\_name": "P\_two", "p\_cost": 20000 },

{ "p\_id": 333, "p\_name": "P\_three", "p\_cost": 30000 },

{ "p\_id": 444, "p\_name": "P\_four", "p\_cost": 40000 },

{ "p\_id": 555, "p\_name": "P\_five", "p\_cost": 50000 }

]

}

}

return state

}

export default reducer

\*\*\*myComponent.js\*\*\*

import React from 'react'

import { connect } from 'react-redux'

class MyComponent extends React.Component {

render() {

return (

<div>

<button onClick={this.props.getProducts}>Products</button>

<br /><br />

<h4>{JSON.stringify(this.props.products)} </h4>

</div>

)

}

}

const receive = (state) => {

return {

products: state.products

}

}

const send = (dispatch) => {

return {

getProducts: () => {

dispatch({ type: 'PRODUCTS' })

}

}

}

export default connect(receive, send)(MyComponent)

\*\*\*index.js\*\*\*

//import reducer

import reducer from './02 Reduxeg/reducer/reducer';

//import provider

import { Provider } from 'react-redux';

//import createStore

import { legacy\_createStore as createStore } from 'redux';

import MyComponent from './02 Reduxeg/myComponent';

//create store

const store = createStore(reducer)

const root = ReactDOM.createRoot(document.getElementById('root'));

root.render(

<Provider store={store}>

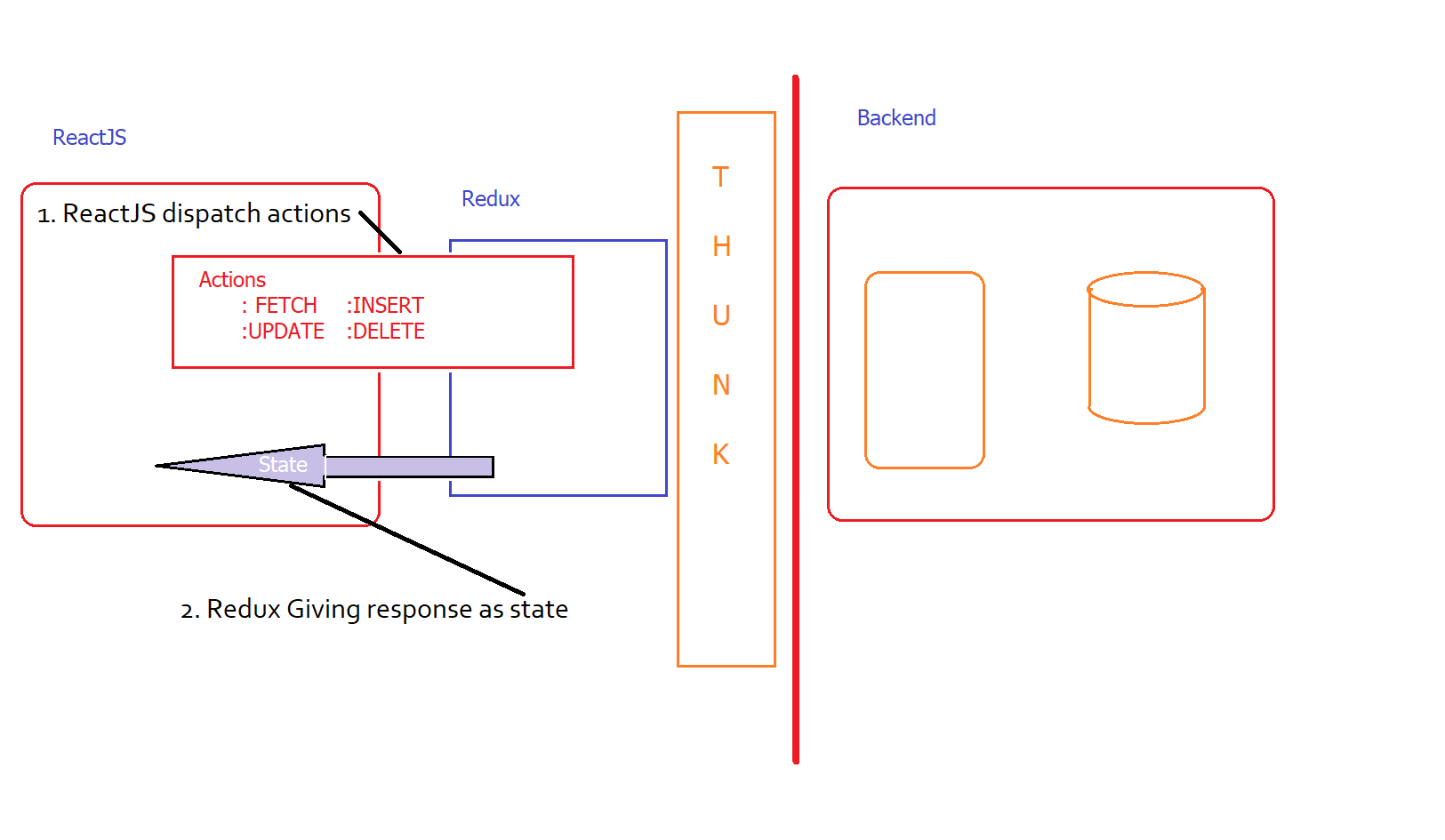
<MyComponent />

</Provider>

);

Thunk Middleware

CRUD Application



url

https://bewildered-puce-wear.cyclic.app/

FETCH https://bewildered-puce-wear.cyclic.app/fetch GET

INSERT https://bewildered-puce-wear.cyclic.app/insert POST

UPDATE https://bewildered-puce-wear.cyclic.app/update POST

DELETE https://bewildered-puce-wear.cyclic.app/delete POST

Terminologies

- Actions

: Used to monitor following actions

: FETCH

: INSERT

: UPDATE

: DELETE

- reducer (Redux)

: used to maintain global states

Download following libraries

- For API calls -> axios

- For redux -> redux

- For react with redux -> react-redux

- Redux with thunk -> redux-thunk

- Bootstrap styling -> bootstrap, react-bootstrap

> yarn add axios redux react-redux redux-thunk bootstrap react-bootstrap --save

Create actions

<>

src

actions

- actions.js

- url.js

\*\*\*url.js\*\*\*

let url = `https://bewildered-puce-wear.cyclic.app`

export default url

\*\*\*actions.js\*\*\*

import axios from 'axios'

import url from '../url'

export const getProducts = () => {

return (dispatch) => {

return axios.get(url + '/fetch')

.then((posRes) => {

dispatch(readAction(posRes.data))

}, (errRes) => {

console.log(errRes)

})

}

}

const readAction = (records) => {

return {

type: 'FETCH', value: records

}

}

Create reducer

<>

src

reducer

- reducer.js

\*\*\*reducer.js\*\*\*

const intialState = {

data: []

}

const reducer = (state = intialState, actions) => {

switch (actions.type) {

case 'FETCH':

state.data = []

return {

...state,

data: state.data.concat(actions.value)

}

}

return state

}

export default reducer

\*\*\*myComponent.js\*\*\*

import React from 'react'

import { connect } from 'react-redux'

import \* as actions from './actions/actions'

class MyComponent extends React.Component {

componentDidMount() {

this.props.getProducts()

}

render() {

return (

<div>

data : {JSON.stringify(this.props.data)}

</div>

)

}

}

const receive = (state) => {

return {

data: state.data

}

}

const send = (dispatch) => {

return {

getProducts: () => {

dispatch(actions.getProducts())

}

}

}

export default connect(receive, send)(MyComponent)

6. Configure Thunk Middleware

\*\*\*index.js\*\*\*

//import reducer

import reducer from './03 ThunkApp/reducer/reducer';

//import provider

import { Provider } from 'react-redux';

//import thunk middleware

import thunk from 'redux-thunk'

//import createStore

import { applyMiddleware, legacy\_createStore as createStore } from 'redux';

import MyComponent from './03 ThunkApp/myComponent';

//create store

const store = createStore(reducer, applyMiddleware(thunk))

const root = ReactDOM.createRoot(document.getElementById('root'));

root.render(

<Provider store={store}>

<MyComponent />

</Provider>

);

1. componentDidMount() method makes action getProducts

- getProducts: () => {

dispatch(actions.getProducts())

}

2. In actions

getProducts() -> fetch data -> records

return { type : 'FETCH', value : records }

3. In reducer

records will concatenated to global state

data : state.data.concat(actions.value)

reducer return state

4. myComponent.js receive state

///////////////////

TEST APPLICATION AT THIS STAGE

AFTER THAT PROCEED

///////////////////

\*\*\*actions.js\*\*\*

import axios from 'axios'

import url from '../url'

export const getProducts = () => {

return (dispatch) => {

return axios.get(url + '/fetch')

.then((posRes) => {

dispatch(readAction(posRes.data))

}, (errRes) => {

console.log(errRes)

})

}

}

const readAction = (records) => {

return {

type: 'FETCH', value: records

}

}

export const insertProduct = (record) => {

return (dispatch) => {

return axios.post(url + '/insert', record)

.then((posRes) => {

dispatch(insertAction(posRes.data))

}, (errRes) => {

console.log(errRes)

})

}

}

const insertAction = (result) => {

return {

type: 'INSERT', value: result

}

}

export const updateProduct = (record) => {

return (dispatch) => {

return axios.post(url + '/update', record)

.then((posRes) => {

dispatch(updateAction(posRes.data))

}, (errRes) => {

console.log(errRes)

})

}

}

const updateAction = (result) => {

return {

type: 'UPDATE', value: result

}

}

export const deleteProduct = (record) => {

return (dispatch) => {

axios.post(url + '/delete', record)

.then((posRes) => {

dispatch(deleteAction(posRes.data))

}, (errRes) => {

console.log(errRes)

})

}

}

const deleteAction = (result) => {

return {

type: 'DELETE', value: result

}

}

\*\*\*reducer.js\*\*\*

const intialState = {

data: []

}

const reducer = (state = intialState, actions) => {

switch (actions.type) {

case 'FETCH':

state.data = []

return {

...state,

data: state.data.concat(actions.value)

}

case 'INSERT':

case 'UPDATE':

case 'DELETE':

return {

...state,

status: actions.value

}

}

return state

}

export default reducer

\*\*\*myComponent.js\*\*\*

import React from 'react'

import { connect } from 'react-redux'

import \* as actions from './actions/actions'

import { Modal, Table } from 'react-bootstrap'

import 'bootstrap/dist/css/bootstrap.css'

let arr = []

class MyComponent extends React.Component {

constructor() {

super()

this.state = {

loading: false,

status: false,

insertPopup: false,

updatePopup: false

}

}

componentDidMount() {

this.props.getProducts()

}

showPopup = (msg) => {

if (msg === `addRec`) {

this.setState({

status: true,

insertPopup: true,

updatePopup: false

})

}

else {

this.setState({

status: true,

insertPopup: false,

updatePopup: true

})

}

}

closePopup = () => {

this.setState({

status: false

})

}

save = (e) => {

e.preventDefault()

if (this.state.insertPopup)

this.insert(e)

else

this.update(e)

this.closePopup()

}

insert = (e) => {

let obj = {

"p\_id": e.target.p\_id.value,

"p\_name": e.target.p\_name.value,

"p\_cost": e.target.p\_cost.value

}

this.props.insertProduct(obj)

this.setState({

result: "Insert Success"

})

arr.push(obj)

}

update = (e) => {

let obj = {

"p\_id": e.target.p\_id.value,

"p\_name": e.target.p\_name.value,

"p\_cost": e.target.p\_cost.value

}

this.props.updateProduct(obj)

this.setState({

result: "Update Success"

})

arr.forEach((e, i) => {

if (e.p\_id === obj.p\_id) {

e.p\_name = obj.p\_name

e.p\_cost = obj.p\_cost

}

})

}

delette = (\_id) => {

this.props.deleteProduct(\_id)

this.setState({

result: "Delete Success"

})

arr.splice(arr.findIndex((e, i) => {

return e.p\_id == \_id

}), 1)

}

render() {

arr = this.props.data

return (

<div className='container mt-5'>

<button className='btn btn-outline-primary mb-2 mr-auto'

onClick={(() => { this.showPopup('addRec') })}> Add +</button>

{/\* ----- modal code start------- \*/}

<Modal show={this.state.status}

onHide={this.closePopup}

size='sm'

centered>

<div className='modal-header'>

<div className='modal-title'>Add / Update</div>

</div>

<div className='modal-body'>

<form onSubmit={this.save}>

<div className='form-group'>

<label>P\_ID</label>

<input type='number'

className='form-control my-2'

placeholder='Enter P\_ID'

name='p\_id'></input>

</div>

<div className='form-group'>

<label>P\_NAME</label>

<input type='text'

className='form-control my-2'

placeholder='Enter P\_NAME'

name='p\_name'></input>

</div>

<div className='form-group'>

<label>P\_COST</label>

<input type='number'

className='form-control my-2'

placeholder='Enter P\_COST'

name='p\_cost'></input>

</div>

<input type='submit' value='Add / Update' className='btn btn-success m-3'></input>

<button className='btn btn-danger m-3' onClick={this.closePopup}>Close</button>

</form>

</div>

</Modal>

{/\* ----- table code start------- \*/}

<Table bordered

variant='secondary'

size='sm'

hover

striped

className='text-center'>

<thead>

<tr>

<th>Srno</th>

<th>ID</th>

<th>NAME</th>

<th>COST</th>

<th>EDIT</th>

<th>DELETE</th>

</tr>

</thead>

<tbody>

{arr.map((element, index) => (

<tr key={index}>

<td>{index + 1} </td>

<td>{element.p\_id} </td>

<td>{element.p\_name} </td>

<td>{element.p\_cost} </td>

<td><button className='btn btn-success' onClick={() => { this.showPopup("update") }}>U</button></td>

<td><button className='btn btn-danger' onClick={() => { this.delette(element.p\_id) }}>D</button> </td>

</tr>

))}

</tbody>

</Table>

</div>

)

}

}

const receive = (state) => {

return {

data: state.data

}

}

const send = (dispatch) => {

return {

getProducts: () => { dispatch(actions.getProducts()) },

insertProduct: (record) => { dispatch(actions.insertProduct(record)) },

updateProduct: (record) => { dispatch(actions.updateProduct(record)) },

deleteProduct: (id) => { dispatch(actions.deleteProduct({ "p\_id": id })) }

}

}

export default connect(receive, send)(MyComponent)

Deploying react application to (Firebase)

\* build ReactJS application

>npm run build

1. https://console.firebase.google.com/

2. Create new project

3. Continue 2 times

4. Configure Google Analytics -> default account

5. Click on create project wait till finish setup

click on continue

6. Click on web (</>)

register app

add firebase sdk

left side panel under build select hosting

7. click on get started

8. install firebase tools

>npm install -g firebase-tools

9. after installing click on next

10. Initialise your project

Sign in to google

>firebase login

11. initialise project

>firebase init

- y

- select hosting Configure files for Firebase Hosting and (optionally) set up GitHub Action deploys

- hit spacebar to select and hit enter

- use existing project -> select projectname

- public directory 'build'

- configure single page application -> y

- setup auto deploy -> no

- DO NOT OVERWRITE index.html

12. Click on next

13. Firebase deploy

>firebase deploy

14. Click on continue to console