6.REDUX with class component.

1.Устанавливаем Redux:

Npm install redux.

2.Создаем папку redux и store.js.

3.Устанавливаем папку react-redux:

Npm install react-redux;

4.В index.js импортируем Provider из react-redux:

import React from 'react';

import ReactDOM from 'react-dom';

import './scss/app.scss';

import App from './App';

import reportWebVitals from './reportWebVitals';

import {BrowserRouter} from "react-router-dom";

import {Provider} from "react-redux"

import store from './redux/store';

ReactDOM.render(

<Provider store={store}>

<BrowserRouter>

   <App />

</BrowserRouter>

</Provider>

 ,

  document.getElementById('root')

);

reportWebVitals();

5.Немного ранее мы создали store и закомбайнили редусеры(несуществующие):

import { combineReducers, createStore } from "redux";

let reducers=combineReducers({})

 let store=createStore(reducers)

 window.store=store

 export default store

6.Создаем filters\_Reducer.js:

const initialState={

sortBy:"popular",

category:0

}

const filters\_Reducer=(state=initialState,action)=>{

  if(action.type==="SET\_SORT\_BY"){

return {

...state,sortBy:action.payload

}

  }

    return state

}

export default filters\_Reducer

Начальное состояние.Будем менять.

7.Изменяем немного Home.Делаем из массива строк varies –массив обьектов:

  <SortPopupWithHooks varies={[{name:"популярности",type:"popular"},

           {name:"цене",type:"price"},{name:"алфавиту",type:"alphabet"}]}/>

8.Меняем SortPopupWithHooks:

import React, { useEffect, useRef, useState } from "react";

const SortPopupWithHooks=(props)=>{

/\*  1  \*/const[activeItem,setActiveItem]=useState(/\* был 0+\*/0)

const[visiblePopup,setVisiblePopup]=useState(false)

const sortRef=useRef(null)

const func=(event)=>{

  !event.path.some(elem=>elem===sortRef.current) &&  setVisiblePopup(false)

}

useEffect((e)=>{

document.body.addEventListener("click",func)},

[]

)

const toggleVisiblePopup=()=>{

  setVisiblePopup(!visiblePopup)

}

/\*2  +\*/const chooseParam=(index)=>{

    setActiveItem( index)

    setVisiblePopup(false)

}

return <div ref ={sortRef} className="sort">

<div className="sort\_\_label">

  <svg className={visiblePopup?"rotated":""}

    width="10"

    height="6"

    viewBox="0 0 10 6"

    fill="none"

    xmlns="http://www.w3.org/2000/svg"

  >

    <path

      d="M10 5C10 5.16927 9.93815 5.31576 9.81445 5.43945C9.69075 5.56315 9.54427 5.625 9.375 5.625H0.625C0.455729 5.625 0.309245 5.56315 0.185547 5.43945C0.061849 5.31576 0 5.16927 0 5C0 4.83073 0.061849 4.68424 0.185547 4.56055L4.56055 0.185547C4.68424 0.061849 4.83073 0 5 0C5.16927 0 5.31576 0.061849 5.43945 0.185547L9.81445 4.56055C9.93815 4.68424 10 4.83073 10 5Z"

      fill="#2C2C2C"

    />

  </svg>

  <b>Сортировка по:</b>

{/\*5 +\*/} <span onClick={toggleVisiblePopup}>{props.varies[activeItem].name/\* был activeItem\*/ }</span>

  {window.visiblePopup=visiblePopup }

</div>

{visiblePopup && <div className="sort\_\_popup">

  <ul>

     {props.varies.map((elem,index)=>{

 /\*3+\*/    return  <li key={`${elem}\_${index}`} onClick={()=>chooseParam(index )}

 /\*4+   \*/    className={activeItem===index ? "active":""}>{elem.name}</li>

      })

    }

    </ul>

</div>}

</div>

}

export default SortPopupWithHooks

Промежуточный вид.

9.Идем в редусер –filters\_Reducer.js:

Там все хорошо .

10.Создаем setSortByActionCreator.js:

export const setSortBy=(name)=>({

type:»SET\_SORT\_BY»,

payload:name

})

11. Создаем setCtegoryActionCreator.js:

export const setCategory=(catIndex)=>({

type:"SET\_CATEGORY",

payload:catIndex

})

12.Создаем pizzas\_Reducer:

const initialState={

    pizzas:[]

    }

    const pizzas\_Reducer=(state=initialState,action)=>{

      if(action.type==="SET\_PIZZAS"){

    return {

    ...state,pizzas:action.payload

    }

      }

        return state

      }

    13.Создаем setPizzasActionCreator:

export const setPizzas=(pizzas)=>({

type:"SET\_PIZZAS",payload:pizzas

})

14.Добавили условие для SET\_CATEGORY типа в редусер:

export const setCategory=(catIndex)=>({

type:"SET\_CATEGORY",

payload:catIndex

})

15.При помощи store.dispatch мы можем напрямую делать изменения в редусере ,изменяя обьект Action.Например

Store.dispatch({type:SET\_SORT\_BY,payload:»price»}).

16.Пишем классовый компонент App:

class App extends React.Component{

componentDidMount(){

  fetch("http://localhost:3000/db.json")

  .then((response)=>response.json()).then(json=>{

    store.dispatch(setPizzaActionCreator(json.pizzas))

  })

}

render(){

return  (

  <div>

   <div className="wrapper">

      <Header/>

      <div className="content">

        <Route exact path="/" render={()=><Home pizzas= {this.pizzas}/>}/>

        <Route  path="/cart" component={Cart}/>

      </div>

    </div>

    </div>

  )

}

}

Начальное состояние.

17.Оборачиваем App.js коннектом:

class App extends React.Component{

componentDidMount(){

  fetch("http://localhost:3000/db.json")

  .then((response)=>response.json()).then(json=>{

    store.dispatch(setPizzasActionCreator(json.pizzas))

  console.log(json.pizzas)

  })

}

render(){

return  (

  <div>

   <div className="wrapper">

      <Header/>

      <div className="content">

        <Route exact path="/" render={()=><Home pizzas= {this.props.pizzas}/>}/>

        <Route  path="/cart" component={Cart}/>

      </div>

    </div>

    </div>

  )

}

}

const mapStateToProps=(state)=>{

return {

pizzas:state.pizzas.pizzas

}

}

export default connect (mapStateToProps) (App);

18.Добавляем mapDispatchToProps в connect.Это нужно для того ,чтобы ыефеу и экшены прокинулись в пропсы и ре нужно было писать store.dispatch в компоненте:

const mapDispatchToProps=(dispatch)=>{

return {

setPizza:(pizzas)=>dispatch(setPizzasActionCreator(pizzas))

}

}

19.Вид классовой компоненты:

class App extends React.Component{

componentDidMount(){

  fetch("http://localhost:3000/db.json")

  .then((response)=>response.json()).then(json=>{

    this.props.setPizza(json.pizzas)

  console.log(json.pizzas)

  })

}

render(){

return  (

  <div>

   <div className="wrapper">

      <Header/>

      <div className="content">

        <Route exact path="/" render={()=><Home pizzas= {this.props.pizzas}/>}/>

        <Route  path="/cart" component={Cart}/>

      </div>

    </div>

    </div>

  )

}

}

const mapStateToProps=(state)=>{

return {

pizzas:state.pizzas.pizzas

}

};

const mapDispatchToProps=(dispatch)=>{

return {

setPizza:(pizzas)=>dispatch(setPizzasActionCreator(pizzas))

}

}

export default connect (mapStateToProps,mapDispatchToProps) (App);