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

const **Screen03** = (props) => {

const { navigate } = props.navigation;

const [data, setData] = useState([]);

const [filter, setFilter] = useState("");

const [newName, setNewName] = useState("");

const [newDescription, setNewDescription] = useState("");

const **getData** = async () => { await fetch("https://654efe92358230d8f0ccf0fb.mockapi.io/Login/filter")

.then((response) => response.json()) .then((json) => setData(json)) .catch((error) => console.error(error)); };

useEffect(() => { **getData**(); }, [data]);

const **deleteItem** = async (id) => { await fetch( `https://654efe92358230d8f0ccf0fb.mockapi.io/Login/filter/${id}`,

{ method: "DELETE", } ); };

const **updateItem** = async (id, name, description) => {

await fetch( `https://654efe92358230d8f0ccf0fb.mockapi.io/Login/filter/${id}`,

{ method: "PUT", headers: { "Content-Type": "application/json", },

body: JSON.stringify({ name: name, description: description, }), } ); };

return ( <View style={{ flex: 1, alignItems: "center", justifyContent: "center",

<TextInput onChangeText={(text) => setFilter(text)} placeholder="Find by name" multiline={true}

<FlatList

data={data.filter((item) => { if (filter == "") return true; return item.name.includes(filter); })}

renderItem={ ({ item }) => (

// item.id < 5 ? (

<View> <View > <TextInput onChangeText={(text) => setNewName(text)} defaultValue={item.name} editable={true} >…

<TextInput onChangeText={(text) => setNewDescription(text)} defaultValue={item.description} editable={true}>…<>

<TouchableOpacity

onPress={() => {**deleteItem**(item.id); **getData**(); }} style={{ }} > <Text >Xóa</Text> </TouchableOpacity>

<TouchableOpacity

onPress={() => {**updateItem**( item.id, newName == "" ? item.name : newName, newDescription == "" ? item.description : newDescription ); **getData**(); }} style={{ }} > <Text>Sửa</Text> </TouchableOpacity>

</View>

)

// ) : ( <View></View> )

}

keyExtractor={(item) => item.id} />

<TouchableOpacity onPress={() => { navigate("Screen02"); }} > <Text > Add </Text> </TouchableOpacity>

const **Screen02** = (props) => {

const { navigation, route } = props;

const { navigate, goBack } = navigation;

const [name, setName] = React.useState("");

const [description, setDescription] = React.useState("");

const addNew = async () => { const newC = { name, description };

await fetch("https://654efe92358230d8f0ccf0fb.mockapi.io/Login/filter", {

method: "POST", headers: { "Content-Type": "application/json", }, body: JSON.stringify(newC), })

.then((response) => response.json()) .then((responseJson) => { console.log("responseJson", responseJson); })

.catch((error) => { console.log("error", error); }); };

return ( <TextInput onChangeText={(text) => setName(text)} placeholder="Enter name"

<TextInput onChangeText={(text) => setDescription(text)} placeholder="Enter"

<TouchableOpacity onPress={() => { addNew(), navigate("Screen03");}} > <Text > Submit </Text> </TouchableOpacity>

import { Provider } from "react-redux"; import store from "./rdtk/store";import Test from "./Test";

export default function **App**() { return ( <Provider store={store}> <Test /> </Provider> );}

function **Reducer**(state = { result: 0 }, action) {switch (action.type) {

case "ACTION1": return { result: state.result + action.payload, };

default: return state; }}

export default Reducer;

import { createStore } from "redux";

import Reducer from "./Reducer";

const store = createStore(Reducer);

export default store;

import { createSlice } from "@reduxjs/toolkit";

export const **Reducer** = createSlice({name: "abc", initialState: { result: 0, },

reducers: { **cong**: (state, action) => {state.result += action.payload; },**tru**: ,…},

});

export const { cong, tru, nhan, chia } = Reducer.actions;

export default Reducer.reducer;

import { configureStore } from "@reduxjs/toolkit";

import Reducer from "./Reducer";

const store = configureStore({

reducer: { theme: Reducer, },});

export default store;

import { Provider, connect } from "react-redux";

import { useState } from "react";

const **Test** = (props) => { const { cong, tru, nhan, chia, result } = props;

const [number, setNumber] = useState(0);

return (

<View style={styles.container}>

<TextInput onChangeText={(text) => { setNumber(parseInt(text)); }} ></TextInput>

<TouchableOpacity onPress={() => **cong**(number)}> <Text>+</Text> </TouchableOpacity>

<TouchableOpacity onPress={() => **tru**(number)} > <Text>-</Text> </TouchableOpacity>

<Text> {**result**} </Text> </View> );};

const mapStateToProps = (state) => { return { result: state.result, }; };

const mapDispatchToProps = (dispatch) => { return {

cong: (payload) => dispatch({ type: "ACTION1", payload: payload }), };};

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

import React, { useState } from "react";

import { useDispatch, useSelector } from "react-redux";

import { chia, cong, nhan, tru } from "./rdtk/Reducer";

const **TestRT** = () => { const [number, setNumber] = useState(0);

const result = useSelector((state) => state.theme.result); const dispatch = useDispatch();

return ( <View> <TextInput onChangeText={(text) => { setNumber(parseInt(text)); }}></TextInput>

<TouchableOpacity onPress={() => **dispatch(cong(number))**} > <Text>+</Text> </TouchableOpacity>

<Text > {result} </Text> </View> );};

export default TestRT;

"@expo/webpack-config": "^19.0.0",

"@react-navigation/native-stack": "^6.9.14",

"@reduxjs/toolkit": "^1.9.7",

"expo": "~49.0.13",

"expo-image": "^1.5.1",

"expo-status-bar": "~1.6.0",

"react": "18.2.0",

"react-dom": "18.2.0",

"react-native": "0.72.5",

"react-native-web": "~0.19.6",

"react-redux": "^8.1.3",

"redux": "^4.2.1"

git config --global user.name "John Doe"

git config --global user.email johndoe@example.com

git config --list

git init

git add .

git commit -m "ghi gi thi gi"