Express

1. Basic server

1.1. สร้าง folder สำหรับ project

1.2. npm init -y เพื่อบอกว่าโปรเจคนี้ใช้ npm

1.3. สร้างไฟล์ index.js

Source code

**var** express = ***require***(**'express'**);

**var** app = express();

app.get(**'/'**, **function**(req, res){

res.send(**'Hello world'**)

});

app.listen(8000);

1.4. npm i -s express

1.5. node index

ดูผลลัพธ์ได้ใน Google Chrome พิม localhost:8000

1. Rest API

CRUD

เปลี่ยนโค๊ด ไฟล์ index.js

let express = require('express');

let bodyParser = require('body-parser');

let cors = require('cors');

let router = express.Router();

let app = express();

app.use(cors());

// all of our routes will be prefixed with /api

app.use('/api', bodyParser.json(), router); //[use json]

app.use('/api', bodyParser.urlencoded({ extended: false }), router);

let bearIndex = 2;

let bears = [

{ 'id': 0, 'name': 'pooh', 'weight': 211 },

{ 'id': 1, 'name': 'vinnie', 'weight': 111 }

];

router.route('/bears')

.get((req, res) => res.json(bears))

.post((req, res) => {

var bear = {};

bear.id = bearIndex++;

bear.name = req.body.name;

bear.weight = req.body.weight

bears.push(bear);

res.json(bears)

})

router.route('/bears/:bear\_id')

.get((req, res) => res.json(bears[req.params.bear\_id])) // get a bear

.put((req, res) => { // Update a bear

var id = req.params.bear\_id

bears[id].name = req.body.name;

bears[id].weight = req.body.weight;

res.json(bears);

})

.delete((req, res) => { // Delete a bear

delete bears[req.params.bear\_id]

res.json(bears);

})

app.use("\*", (req, res) => res.status(404).send('404 Not found'));

app.listen(8000, () => console.log("Server is running"));

npm i -s express body-parser cors

node index

ดูผลลัพธ์ได้ใน Google Chrome พิม localhost

let express = require('express');

let bodyParser = require('body-parser');

let cors = require('cors');

let router = express.Router();

let app = express();

app.use(cors());

// all of our routes will be prefixed with /api

app.use('/api', bodyParser.json(), router); //[use json]

app.use('/api', bodyParser.urlencoded({ extended: false }), router);

let studentIndex = 1;

let students = [

{

'id': '5835512048',

'name': 'Tanakorn',

'surname': 'Karode',

'major': 'COE',

'gpa': 3.00

},

];

router.route('/students')

.get((req, res) => res.json(students))

.post((req, res) => {

var student = {};

student.id = req.body.id;

student.name = req.body.name;

student.surname = req.body.surname;

student.major = req.body.major;

student.gpa = req.body.gpa;

students.push(student);

res.json(students)

})

router.route('/students/:student\_id')

.get((req, res) => {

res.json(students[req.params.student\_id])

}) // get a bear

.put((req, res) => { // Update a bear

var id = req.params.student\_id

students[id].id = req.body.id;

students[id].name = req.body.name;

students[id].surname = req.body.surname;

students[id].major = req.body.major;

students[id].gpa = req.body.gpa;

res.json(students);

})

.delete((req, res) => { // Delete a bear

delete students[req.params.student\_id]

res.json(students);

})

app.use("\*", (req, res) => res.status(404).send('404 Not found'));

app.listen(8000, () => console.log("Server is running"));

React

1. npx create-react-app ชื่อโปรเจค(ภาษาอังกฤษ นะครับ)
2. cd ชื่อโปรเจค
3. npm i
4. npm start
5. ดูใน Browser พิม url ว่า localhost:3000

แนะนำ

ลบไฟล์ในโฟลเดอร์ src ให้เหลือแค่ index.js App.js serviceWorker.js

(อย่าลืมลบที่ import ใน index.js และ App.js)

ใน App.js แบบ Basic สุดๆ

import React from 'react';

class App extends React.Component {

render() {

return (

<div>

Hello World

</div>

)

}

}

export default App;

**Redux**

1. npm i -s redux react-redux redux-thunk axios
2. สร้าง folder redux ใน src
3. สร้าง store.js

import { combineReducers, createStore, applyMiddleware } from 'redux';

import thunk from 'redux-thunk';

import StudentReducer from './student/reducer';

const reducers = combineReducers({

Student: StudentReducer

})

export default createStore(reducers, applyMiddleware(thunk));

1. สร้าง actions/reducer
   1. สร้าง folder ชื่อ student ใน redux
   2. สร้างไฟล์ actions.js

import axios from 'axios';

const url = 'http://localhost:8000/api/students'

const types = {

FETCH\_STUDENTS: 'FETCH\_STUDENTS'

}

const dispatcher = {

fetchStudents: () => async (dispatch, getState) => {

const result = await axios.get(url);

dispatch({ type: types.FETCH\_STUDENTS, students: result.data })

}

}

export default {

types,

dispatcher

}

* 1. สร้างไฟล์ reducer.js

import StudentAction from './actions';

const types = StudentAction.types;

const initialState = {

students: []

}

export default function StudentReducer(state = initialState, action) {

switch(action.type) {

case types.FETCH\_STUDENTS:

return {

...state,

students: action.students

}

default :

return state;

}

}

1. เพิ่ม Provider ใน App.js

import React from 'react';

import { Provider } from 'react-redux';

import store from './redux/store';

import Student from './Student';

class App extends React.Component {

render() {

return (

<Provider store={store}>

<Student/>

</Provider>

)

}

}

export default App;

6. ตัวอย่างการใช้ Redux ในไฟล์ Student.js

import React from 'react';

import { connect } from 'react-redux';

import StudentAction from './redux/student/actions';

const dispatcher = StudentAction.dispatcher;

class Student extends React.Component {

render() {

return (

<div>

<button onClick={this.handleClick}>Fetch</button>

{

this.props.Student.students.map((student) => (

<div>

{student.id} <br/>

{student.name} <br/>

{student.surname} <br/>

{student.major} <br/>

{student.gpa} <br/>

</div>

))

}

</div>

)

}

handleClick = () => {

this.props.fetchStudents();

}

}

export default connect(state => state, dispatcher)(Student);

Action //ฟังก์ชันในการจัดการ state กลาง (ทำอะไรกับข้อมูล)

Reducer //Pure function ที่เก็บข้อมูล (ข้อมูล Update ยังไง)

Store //ตัวเก็บข้อมูลกลาง (รวม Reducer หลายตัว เป็น store อันเดียว)

Source code:

React : <https://github.com/tanakorn0314/react-redux-thunk>

Express : <https://github.com/tanakorn0314/express-simple-rest>

อย่าลืม npm i ก่อนรันนะครับผม

โจทย์

1. ทำ Server สำหรับเก็บข้อมูลสินค้า product (id, name, price) สามารถทำ CRUD ได้
2. ทำหน้าเว็บด้วย React + Redeux
   1. แสดงข้อมูลสินค้า
   2. เพิ่มสินค้าได้
   3. อัพเดทสินค้า
   4. ลบสินค้า

ProductList.js

import React from 'react';

import { connect } from 'react-redux';

import ProductAction from './redux/product/actions';

const productDispatcher = ProductAction.dispatcher;

class ProductList extends React.Component {

render() {

return (

<div>

{

this.props.Product.products.map((product, index) => (

<div>

<div>id = {product.id}</div>

<div>name = {product.name}</div>

<div>price = {product.price}</div>

<br/>

</div>

))

}

<button onClick={this.handleClick}>Test</button>

</div>

)

}

componentDidMount() {

this.props.fetchAllProducts();

}

handleClick = () => {

console.log(this.props.Product.products)

}

}

export default connect(

state => state,

productDispatcher

)(ProductList);