Creamos la carpeta clienteReactJs. Abrimos terminal y ejecutamos:

npm install -g create-react-app. //not necessary if it’s already installed

npx create-react-app my-client —typescript

Habremos generado toda la estructura de carpetas que hemos explicado anteriormente.

En la carpeta src copiamos el archivo client.ts que programamos en apartados anteriores. Entramos en la carpeta my-client y ejecutamos:

npm install xhr2 --save //for client.ts -> allow running on browser and server

Creamos el archivo environment.json de igual forma que el anterior proyecto.

Instalamos loas siguientes librerías para la interfaz gráfica:

npm install @types/jquery

npm install @types/bootstrap

Y el router para gestionar las URLs:

npm install @types/react-router-dom

Nota importante, al igual que la otra vez, vamos a ejecutar el servidor a la misma vez el cliente. Es decir, ejecutamos el servidor NodeJS para ServerRestFul en el puerto 8080 y el cliente en el servidor de simulación 3000, por defecto. Esto provoca problemas con las directivas CORS, como vimos en el proyecto anterior. Para resolverlo devemos instalar un proxy:

npm i --save http-proxy-middleware

Ahora en src creamos un archivo denominado setupProxy.js con el código:

const proxy = require('http-proxy-middleware');

module.exports = function(app) {

app.use(proxy('/items',

{ target: 'http://localhost:8080/' }

));

}

#### Código fuente

index.tsx

import React from "react";

import ReactDOM from "react-dom";

import "./index.css";

import App from "./App";

import About from "./About";

import \* as serviceWorker from "./serviceWorker";

import { Route, BrowserRouter, Switch } from "react-router-dom";

ReactDOM.render(

<BrowserRouter>

<Switch>

<Route exact path="/" component={App} />

<Route path="/about" component={About} />

</Switch>

</BrowserRouter>,

document.getElementById("root")

);

// If you want your app to work offline and load faster, you can change

// unregister() to register() below. Note this comes with some pitfalls.

// Learn more about service workers: https://bit.ly/CRA-PWA

serviceWorker.unregister();

App.tsx

import React, { Component } from "react";

import "./App.css";

import "bootstrap/dist/css/bootstrap.min.css";

import EditItem from "./EditItem";

import { clientRestful } from "./client";

import { item } from "./items";

import \* as enviroment from "./enviroment.json";

import MyModal from "./MyComponents/MyModal";

class App extends Component {

myService: clientRestful;

state = {

selectedItem: {

id: 0,

title: "",

description: ""

},

myitems: [],

editItem: false,

newItem: false

};

constructor(props: any) {

super(props);

this.onSubmit = this.onSubmit.bind(this); //child can access to this of this class

this.onUpdateItem = this.onUpdateItem.bind(this);

this.onCreateItem = this.onCreateItem.bind(this);

this.onCancelEdit = this.onCancelEdit.bind(this);

this.myService = new clientRestful(enviroment.endPoint);

}

componentDidMount() {

this.getItems();

}

getItems() {

this.myService.getItems().subscribe(items => {

this.setState({ myitems: items });

});

}

render() {

const items = this.state.myitems;

const editItem = this.state.editItem;

if (!items) return null;

const listItems = items.map((newitem: item) => (

<tr key={newitem.id}>

<td>{newitem.id}</td>

<td>{newitem.title}</td>

<td>{newitem.description}</td>

<td>

<button

className="btn btn-info btn-edit"

onClick={() => this.onEditItem(newitem)}

>

E

</button>

<button

className="btn btn-danger btn-remove"

onClick={() => this.onRemove(newitem.id)}

>

D

</button>

</td>

</tr>

));

return (

<div className="App">

<nav className="navbar navbar-light bg-light">

<a className="navbar-brand" href="about">

<img

src="https://raw.githubusercontent.com/Developodo/home/master/developodomini.png"

width="30"

height="30"

className="d-inline-block align-top"

alt=""

/>

Typescript(React) + Bootstrap (by Developodo)

</a>

</nav>

<MyModal

classButton="float-right"

callback={this.onCreateItem}

message={this.getForm()}

title={"Add new Item"}

openMessage={"Add"}

/>

<button className="btn btn-success" onClick={() => this.onAddItem()}>

Add

</button>

<table id="data" className="table table-striped">

<thead>

<tr>

<th scope="col">#</th>

<th scope="col">Title</th>

<th scope="col">Descripion</th>

<th scope="col">Actions</th>

</tr>

</thead>

<tbody>{listItems}</tbody>

</table>

<hr />

{editItem && (

<EditItem

onSubmit={this.onSubmit}

onCancel={this.onCancelEdit}

item={this.state.selectedItem}

addItem={this.state.newItem}

/>

)}

<MyModal

classButton="float-right margened "

callback={this.onCreateItem}

message={this.getForm()}

title={"Add new Item (2)"}

openMessage={"Another Add"}

/>

</div>

);

}

onEditItem(it: item) {

this.setState({

editItem: true,

newItem: false,

selectedItem: it

});

}

onAddItem() {

this.setState({

editItem: true,

newItem: true,

selectedItem: {

id: 0,

title: "",

description: ""

}

});

}

onRemove(id: string | number) {

this.clearState();

if (window.confirm("Remove it?")) {

this.myService.removeItem(id).subscribe(() => {

this.getItems();

});

}

}

onCancelEdit() {

this.setState({

editItem: false,

newItem: false

});

}

onSubmit(i: item) {

if (this.state.newItem) {

this.onCreateItem(i);

} else {

this.onUpdateItem(i);

}

}

onUpdateItem(uitem: item) {

console.log(uitem);

this.myService

.updateItem(uitem.id, uitem.title, uitem.description)

.subscribe(i => {

this.clearState();

this.getItems();

});

}

onCreateItem(uitem: item) {

console.log(uitem);

this.myService.createItem(uitem.title, uitem.description).subscribe(i => {

this.clearState();

this.getItems();

});

//this.clearState();

}

clearState() {

this.setState({

editItem: false,

newItem: null

});

}

getForm(): any {

let html = (

<div>

<div className="form-group">

<label>Title: </label>

<input

className="form-control"

defaultValue=""

name="title"

required

placeholder="item title"

/>

</div>

<div className="form-group form-row">

<label>Description: </label>

<input

className="form-control"

defaultValue=""

name="description"

required

placeholder="item description"

/>

</div>

</div>

);

return html;

}

}

export default App;

EditItem.tsx

import React, { Component, ChangeEvent } from "react";

import { item } from "./items";

import "./App.css";

interface myprops {

onCancel: any;

onSubmit: any;

item: item;

addItem: boolean;

}

class EditItem extends Component<myprops, {}, any> {

handleFormChange(event: ChangeEvent<HTMLInputElement>) {

this.setState({

change: true

}); //forze render

this.dataItem[event.target.name] = event.target.value;

event.preventDefault();

}

dataItem: any = null;

constructor(props: myprops) {

super(props);

//this.onCancel = this.onCancel.bind(this);

//this.onSubmit = this.onSubmit.bind(this);

this.handleFormChange = this.handleFormChange.bind(this);

this.dataItem = this.props.item;

if (this.props.addItem) {

this.dataItem = {

id: 0,

title: "",

description: ""

};

}

}

//update without closing component

componentWillUpdate(prevProps: any, prevState: any, snapshot: any) {

this.dataItem = prevProps.item;

}

render() {

let textButton: string;

if (this.props.addItem) {

textButton = "Add";

} else {

textButton = "Update";

}

return (

<div>

{this.props.addItem ? (

<div className="alert alert-success">Data for new Item</div>

) : (

<div className="alert alert-primary">

<span>Edit item:</span> <span>{this.dataItem.id}</span>

</div>

)}

<div className="form-group">

<label className="control-label">Title: </label>

<input

className="form-control"

value={this.dataItem.title}

name="title"

required

placeholder="item title"

onChange={this.handleFormChange}

/>

</div>

<div className="form-group">

<label className="control-label">Description: </label>

<input

className="form-control"

value={this.dataItem.description}

name="description"

required

placeholder="item description"

onChange={this.handleFormChange}

/>

</div>

<button

className="btn btn-danger"

onClick={() => this.props.onCancel()}

>

Cancel

</button>

<button

className="btn btn-primary float-right"

onClick={() => {

this.props.onSubmit(this.dataItem);

}}

>

{textButton}

</button>

</div>

);

}

}

export default EditItem;

item.ts

export interface item {

id: number;

title: string;

description: string;

}

MyComponents/MyModal.tsx

import React, { Component } from "react";

import $ from "jquery";

import "popper.js";

import "bootstrap/dist/js/bootstrap.bundle";

import { item } from "../items";

interface myprops {

openMessage: string;

title: string;

message: string;

callback: any;

classButton: string;

}

class MyModal extends Component<myprops> {

state = {

open: false

};

constructor(props: myprops) {

super(props);

this.state.open = false;

this.Open = this.Open.bind(this);

this.Action = this.Action.bind(this);

}

Open() {

console.log("jjj");

this.setState(

{

open: true

},

() => {

$("#addModal").modal("show");

}

);

}

Action() {

let t = String($('[name="title"]').val());

let d = String($('[name="description"]').val());

let newitem: item = {

id: 0,

title: t,

description: d

};

this.props.callback(newitem);

$("#addModal").modal("hide");

this.setState({

open: false

}); //async, need callback, show modal must be after changed

}

render() {

let classButton = "btn btn-primary ";

classButton = classButton + this.props.classButton;

return (

<span>

<button id="myaddbutton" className={classButton} onClick={this.Open}>

{this.props.openMessage}

</button>

{!this.state.open ? (

<span />

) : (

<div

className="modal fade"

id="addModal"

role="dialog"

aria-labelledby="addModalLabel"

aria-hidden="true"

>

<div className="modal-dialog" role="document">

<div className="modal-content">

<div className="modal-header">

<h5 className="modal-title">{this.props.title}</h5>

<button

className="close"

data-dismiss="modal"

aria-label="Close"

>

<span aria-hidden="true">&times;</span>

</button>

</div>

<div className="modal-body">

<div id="addModalContent">{this.props.message}</div>

</div>

<div className="modal-footer">

<button className="btn btn-secondary" data-dismiss="modal">

Close

</button>

<button className="btn btn-primary" onClick={this.Action}>

{this.props.openMessage}

</button>

</div>

</div>

</div>

</div>

)}

</span>

);

}

}

export default MyModal;

About.tsx

import React, { Component } from "react";

class About extends Component {

constructor(props: any) {

super(props);

this.setState({}); //useless

}

render() {

return (

<div>

Developed by Developodo, what else?<br />

<a href="/">Go back!</a>

</div>

);

}

}

export default About;