Date : 23/05/2018

How to combine the application of (React and Express JS)

To set up this application first thing we need that create an Express application

----------------------------------------------------------

TO CREATE AN EXPRESS APP

Run the command “npm install –g express-generator”

Express “name of project” like

Cd “app directory” “Npm install”

To start the application “npm start” by default your app will listen the port not 3000.

You can assign different port no.

--------------------------------------------------------

To set the different port no dynamically for the express js run this command

“set PORT=3001”

Now start your application using “npm start”

And see the output on the browser <http://localhost/3001>

You will get the output “your first express app”

Now create a React application inside your project folder

Make a new folder and name as “client”

Another way is that from command promt you can do all the process, so for that please follow my below instructions-

Go to your command prompt.

Navigate your express project folder.

Run the command (create-react-app “your app name” );

So write your app name “client”

Now navigate the client folder in your command prompt

Now you are ready to go with your react application so you need to change something.

Open your react “package.json” file and edit the code –

Your code will look like –

{

Name:client,

Version:

Dependencies:{

………………

……………..

}, // end of dependencies file

Scripts:{

Start:

Build:

Test:

Eject:

}, // end of script file

// add new one proxy key to listen the express server

“proxy”:<http://localhost:3001> // the local server of express application

}// end of package.json file

Now you are ready to go with your express first application

Run this command in your browser –<http://localhost:3001/>

The output you will get –

**Express**

Welcome to Express

Fetch the data from backend and show that data through the component –

Edit the file “myapp”>routes>users.js

var express = require('express');

var router = express.Router();

/\* GET users listing. \*/

router.get('/', function(req, res, next) {

res.json([

{id:0, name:"wishvanath"},

{id:1, name:"wishva"},

{id:2, name:"wish"}

]); // end of json response

});

module.exports = router;

run the express server on the port no -3001

now you are able to fetch the data from express backend

so study the below code –

“app.js” file –

import React, { Component } from 'react';

import './App.css';

class App extends Component {

state ={

users:[] // empty state of the userlist

}

componentDidMount(){

// fetch the userlist from the express backend

fetch('/users')

.then(res => res.json())

.then(users =>this.setState({users}));

}

render() {

return (

<div className="App">

<h1>User List</h1>

<li>

{this.state.users.map(user =>

<li key = {user.id}>{user.name}</li>

)}

</li>

</div>

);

}

}

export default App;

Define the Function event

import React, { Component } from 'react';

import './App.css';

class App extends Component {

state ={

list:[] // empty state of the userlist

}

componentDidMount(){

// fetch the userlist from the express backend

fetch('/users')

.then(res => res.json())

.then(list =>this.setState({list},console.log(list)));

};

render() {

// define the function in render section user the function keyword

function wish(e){

console.log("you have clicked me ");

}

return (

<div className="App">

<h1>User List</h1>

<li>

{this.state.list.map(user =>

<li key = {user.id}>{user.name}<button onClick = {wish}>click me</button> </li>

)}

</li>

</div>

);

}

}

export default App;

Another method –

To declare the function outside the render method

import React, { Component } from 'react';

import './App.css';

class App extends Component {

state ={

list:[] // empty state of the userlist

}

componentDidMount(){

// fetch the userlist from the express backend

fetch('/users')

.then(res => res.json())

.then(list =>this.setState({list},console.log(list)));

};

wish(e){

console.log("you have clicked me ");

}

render() {

// define the function in render section user the function keyword

return (

<div className="App">

<h1>User List</h1>

<li>

{this.state.list.map(user =>

<li key = {user.id}>{user.name}<button onClick = {this.wish}>click me</button> </li>

)}

</li>

</div>

);

}

}

export default App;

Self declarative Function –

import React, { Component } from 'react';

import './App.css';

class App extends Component {

state ={

list:[] // empty state of the userlist

}

componentDidMount(){

// fetch the userlist from the express backend

fetch('/users')

.then(res => res.json())

.then(list =>this.setState({list},console.log(list)));

};

clickme = () =>{

console.log("you have clicked me");

}

render() {

// define the function in render section user the function keyword

return (

<div className="App">

<h1>User List</h1>

<li>

{this.state.list.map(user =>

<li key = {user.id}>{user.name}<button onClick = {this.clickme}>click me</button> </li>

)}

</li>

</div>

);

}

}

export default App;

To show the data from the state component

import React, { Component } from 'react';

import './App.css';

class App extends Component {

state = {

userName:"wishvanath",

email:"anandwishvanath@gmail.com"

}

render() {

return (

<div className="App">

<h1>User List</h1>

User Name ={this.state.userName},

Email id = {this.state.email}

</div>

);

}

}

export default App;

Another Way to define the state –

import React, { Component } from 'react';

import './App.css';

class App extends Component {

state = {

userName:"wishvanath",

email:"anandwishvanath"

}

render() {

return (

<div className="App">

<h1>User List</h1>

User Name ={this.state.userName},

Email id = {this.state.email}

</div>

);

}

}

export default App;

To pass the data from button event

import React, { Component } from 'react';

import './App.css';

class App extends Component {

state ={

list:[] // empty state of the userlist

}

componentDidMount(){

// fetch the userlist from the express backend

fetch('/users')

.then(res => res.json())

.then(list =>this.setState({list},console.log(list)));

};

clickme(user){

console.log("you have clicked me");

console.log(user.name)

console.log(user.id)

}

render() {

// define the function in render section user the function keyword

return (

<div className="App">

<h1>User List</h1>

<ul>

{this.state.list.map(user =>

<li key = {user.id}>{user.name}<button onClick = {this.clickme.bind(this, user)} >click me</button> </li>

)}

</ul>

</div>

);

}

}

export default App;

Show the state data in table format

import React from 'react';

class Todolist extends React.Component {

constructor(props) {

super(props);

this.state = {

items:[

{id:1, name:"Java"},

{id:2, name:"DBMS"},

{id:3, name:"React"}

]

};

}

render() {

return (

<div>

{/\* show the data of state \*/}

{/\* to show the data into the table \*/}

<table border = "2px">

<tr>

{this.state.items.map(each\_itm =>

<td key = {each\_itm.id}>${each\_itm.name}</td>

)}

</tr>

</table>

<ul>

{this.state.items.map(each\_item =><li key ={each\_item.id} >

{each\_item.name}

</li>)}

</ul>

</div>

);

}

}

Todolist.propTypes = {};

export default Todolist;

How to integrate the express and react in same module-

Till now we learnt that how to integrate the express and react module in one. But both application is running on different port no and we have to start two node server one for the express application and another for the react application.

So in this section we will learn how to run both application in one server or how to reduce the time to make it sufficient …

Create both application react and express

So first you have to create express application and then in client folder you have to create the react application.

After that in your express project (main project ) folder install the nodemon dependencies using the command –

“npm install nodemon --save-dev” or

“npm install nodemon –save”

Now you need to change some code in your package.json file of express module –

Which is as –

In script tag edit this code –

{

"name": "myapp",

"version": "0.0.0",

"private": true,

"scripts": {

"start": "nodemon ./bin/www"

},

"dependencies": {

"cookie-parser": "~1.4.3",

"debug": "~2.6.9",

"express": "~4.16.0",

"http-errors": "~1.6.2",

"jade": "~1.11.0",

"morgan": "~1.9.0"

},

"devDependencies": {

"nodemon": "^1.17.5"

}

}

Run the Application –

“npm start”

Note:

This script will work when you have created your express application with express cli.

So another method if you have created your exp application using “npm init”

Then edit code something like this –

"start": "node server.js",

"server": "nodemon server.js"

// now create a simple express server.js fle in your main prorject

Run this Application –

“npm run server”