1. Install nodejs and npm, show 2503 and 2502 error.

Solution:

Go to C:\Windows\temp, right click the folder. Select security, click edit, select users(….), click full control.

A screenshot of a social media post

Description automatically generated

1. Fix a vulnerable npm package in package-lock.json:

npm audit fix, npm will install latest version of package automatically.

1. SetState is not a function: 原因是没有bind function. 比如，原来是onClick={handleLogout}, 变成{this.handleLogout}即可解决
2. Indent: Ctrl k + ctrl f
3. ComponentDidMount is after render. The order is constructor, render,componentDidMount.
4. Difference between Component and pureComponent: component doesn’t implement *shouldComponentUpdate()* by default. On the other hand, purecomponent does implement it and perform a shallow comparison on react state and props values.

It increases app performance. Shallow comparison is comparing scala values and references when comparing object.

1. Name Is missing in props validation:

class Greeting extends React.PureComponent {

  render() {

    return <h1>Hi there, my name is {this.props.name}!</h1>;

  }

}

Add following code:

Greeting.propTypes = {

  name: PropTypes.string

};

Greeting後面的是小寫p開頭,name後面的是大寫P開頭

1. Sass: you can use variables and other sass function
2. Promise: A good rule of thumb is to always either return or terminate promise chains, and as soon as you get a new promise, return it immediately, to flatten things.

From <https://developer.mozilla.org/en-US/docs/Web/JavaScript/Guide/Using_promises>

JS is single threaded, meaning that two bits of script cannot run at the same time. They have to run one after another.

<https://developers.google.com/web/fundamentals/primers/promises>

1. React-scroll: 随着scroll down, navigation menu的标题随着变色
2. React-parallax-tilt: when mouse move on the image, animation spears.
3. State an props: state is equivalent to local variables in a function. Props, on the other hand, is equivalent to function parameters.

Class SampleFunc extends React.Component{

Render(){

Return <div>Hello {this.props.name}</div>

}

}

<SampleFunc name = “Joni ” />

1. Async function always return a promise, whether you use await or not. That promise resolves with whatever the async function returns, or rejects with whatever the async function throws.

<https://developers.google.com/web/fundamentals/primers/async-functions>

fetch(url).then… equals to await fetch(url).