每次開機都要做一次:

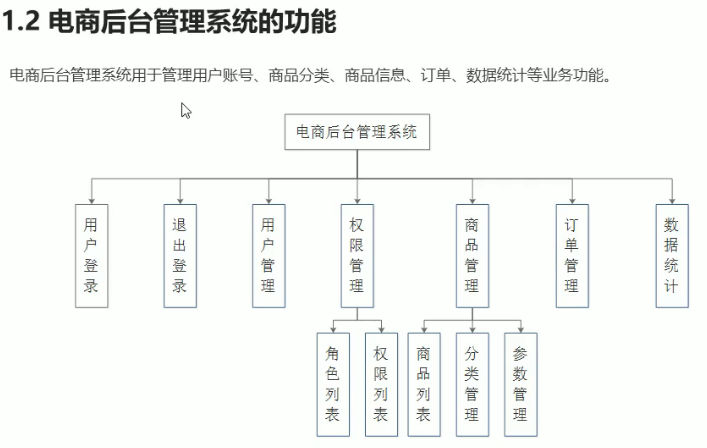
1打開php study 開mysql

2在api\_server路徑下node app.js

3直接開vue ui 命令

4 在ui中開server用

後台管理系統功能



前端SPA（單頁面應用）



前端項目技術棧

Vue

Vue-router

Element-UI

Axios

Echarts 圖表用

後端項目技術棧

Node.js

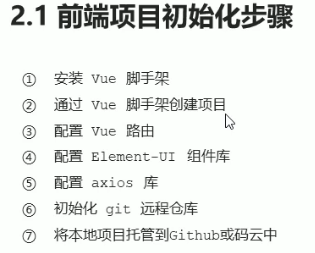
Express

Jwt 狀態保持工具

Mysql 後台數據庫

Sequelize 數據庫框架

前端項目初始化



npm install

全局安裝完最新的vue-cli後

直接 在cmd 輸入vue ui (需要vue 3.0以上)

就可以利用可視化形式創建項目。

我們項目要用vue-cli 3.x/4.x 和vue 2.x 和cli service

vue-cli3.x的安装命令是npm install -g @vue/cli

vue-cli2.x的安装命令是npm install -g vue-cli

npm i vue-router 安裝3.0 vue-router對應vue 2.x

npm install -g @vue/cli

npm i @vue/cli-service

npm i vue-template-compiler

vue 2.x 安裝npm i vue

less-loader也不可以用太高version

npm install less [less-loader@4.1.0](mailto:less-loader@4.1.0) --save

less和less-loader都@4.1.0

Git 倉庫 記得輸入 init project。

之後手動配置, 我們用babel router linter/formatter 和 使用設定檔案。



成功創建項目後,去安裝插件

我們這次用的插件是

vue-cli-plugin-element (element-UI)



之後點依賴 安裝axios 安裝到執行依賴

初始化git遠程倉庫 和 託管到github

教學:

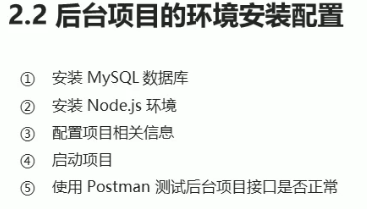
<https://www.jianshu.com/p/31cbbbc5f9fa>

https://www.jianshu.com/p/5a1d203acfd3

生成ssh key。

ssh-keygen -t rsa -C "catcat1997@windowslive.com"

後台安裝



phpstudy的mysql導出路徑為:

D:\phpstudy\_pro\Extensions\MySQL5.7.26\data\mybd

我們的後台

D:\html+css\Vue Learn\vue\_shop\_project\vue\_api\_server

接口位置 <http://127.0.0.1:8888/api/private/v1/>

D:\html+css\Vue Learn\vue\_shop\_project\vue\_api\_server\routes/api/private/v1

mysql -u root password '000000'

登入mysql的命令 –u是user 。

mysql命令要在mysql的bin下使用

D:\phpstudy\_pro\Extensions\MySQL5.7.26\bin

在vue\_api\_server/config下有個default.json

可以設一下自動入的password 我們用000000 。

在D:\html+css\Vue Learn\vue\_shop\_project\vue\_api\_server下

npm install

再跑 node ./app.js

phpstudy的mysql有root和mydb mydb不要寫錯名!!!!。

因為app.js內setting了名字要對應到mydb。

跑node app.js的cmd要長開, 使我們的api接口可以使用。

正常我們用cookie / session 保持登陸狀態

但如果有跨域問題, 我們會用token

token原理:



本地git 倉庫

(以下所有命令(本地/線上上傳),都是在同一路徑(上傳文件夾根目錄))

<https://www.jianshu.com/p/e79e03034ba2>

自己建立一個新的 Repository git init

git add filename 這裡的filename是(dot) .

git add . #添加当前目录下的所有文件

git commit –m “add files” 本地提交 (更新一下改變了的file到本地)

git branch 可以查看分支

git checkout –b login 開創名為login 的分支

上傳git

<https://www.bilibili.com/video/BV1EE411B7SU?p=7>

git remote –v 查看現在地址

新加remote地址:

git remote add origin <https://github.com/catcat1997/vue_shop_test.git>

git remote add origin [git@github.com:catcat1997/vue\_shop\_test.git](mailto:git@github.com:catcat1997/vue_shop_test.git)

上傳文件到github:

git push –u origin master

git push origin master

git remote rm origin 可以刪除remote地址

git status 看查路徑的根目錄是否clean (沒有更變)

如果有更變(不clean) 就可以先上傳一下(本地+線上)

,利用 git add . 和 git commit –m “add files”

vue 2.x 和 vue 3.x 會有語法不同。

vue3.x

import { createApp } from 'vue'

import App from './App.vue'

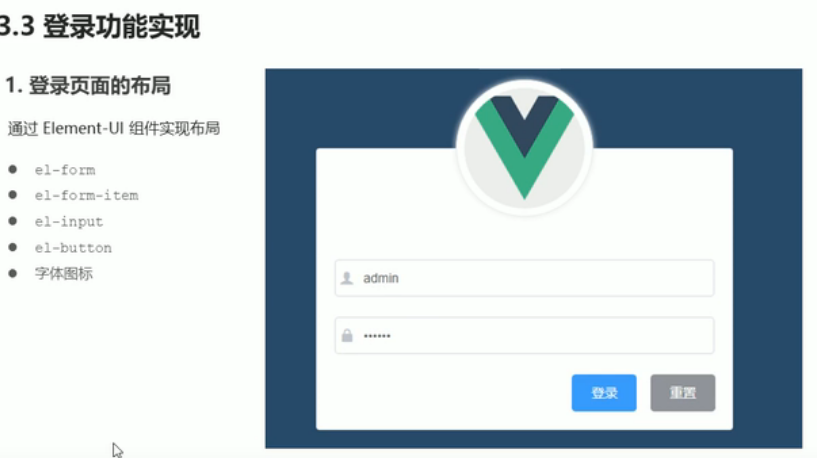
export const eventBus = createApp(App)

createApp(App).mount('#app')

登錄/退出功能

會用到的Element-UI組件 實現布局

el-form el-form-item el-input el-button



利用element組件時記得去查看

標簽樣式和

srcipt的 export default內容。

比如data內寫什麼, data內的數據rule如何寫, 寫完用prop傳值到什麼標簽。

我們的表單rules 驗證規則對象 loginFormRules 內

分別有username和password規則

先在loginForm內 v-bind:rules=”loginFormRules”

之後在username的input內再用prop=”username”傳的。

　　ref 被用來給元素或子組件注冊引用信息，引用信息將會注冊在父組件的 $refs 對象上。

resetFields()

是element 給我們寫好的,我們直接用。

ref和$refs其實就是用來獲取/操作DOM元素的；類似於jquey中的$(".xxx")；

ref 是表單的引用對象

調用時可以this.$refs.<表單引用對象>

比如:

<el-form ref="loginFormRef">

調用時:

this.$refs.loginFormRef.resetFields();

es6 語法, 數據解構

{data: result} result是要解構的對象

解構我們想要的是 data。

編程式導航

this.$router.push('/home');

路由導航守衛控制訪問權限

如果用戶沒有登錄(sessionStorage中沒有token), 但是直接用url訪問特定頁面,就會重新導航到login頁面。 redirect

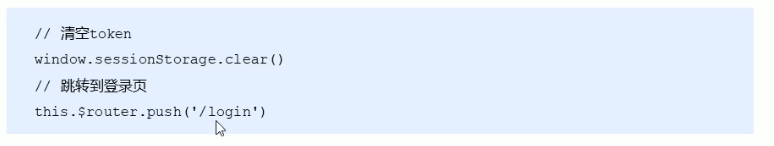
利用beforeEach(to,from,next) 導航守衛



退出功能實現原理

基於token的方式實現退出比較簡單, 只需要銷毀本地的token即可。

這樣, 後續的請求就不會携帶token, 必須重新登錄生成一個新的token之後才可以訪問頁面。



第三方庫的組件使用:

1在main.js引入第三方庫

import './plugins/element.js'

2在第三方庫js內引入需要用的組件:

比如:

import { Button } from 'element-ui'

import { Form, FormItem } from 'element-ui'

Vue.use(Button)

Vue.use(Form)

Vue.use(FormItem)

特殊的掛載

Vue.prototype.$message = Message

3使用:

<el-form-item class="btns">

      <el-button type="primary" @click="login">Login</el-button>

      <el-button type="info" @click="resetLoginForm">Reset</el-button>

</el-form-item>

完成登錄功能

上傳到github

可以在vscode Ctrl+` 開啟terminal

git status 查看更改了的file

git add . 上傳到本地倉

git commit –m “add file” 也可以git commit –m “完成了登錄功能”

我們現在有login 和 master的倉庫branch分支

git checkout master 切換到master分支 (checkout 切換分支)

在master分支下:

git merge login 合并merge login分支下的檔案。

最後可以git push 上傳到github。

之前我們用 git push –u origin master。(推送到github的master分支)

但我們現在就在master分支下, 可以直接push。

如果我們想推送login分支, 也可以:

先轉git checkout login 轉到login分支。

再git push –u origin login。 推送到github的login分支。

主頁布局 /home

CSS

display:flex;

jutify-content: space-between;

align-items:center;

<https://w3c.hexschool.com/flexbox/87d66dc4>

align-items 和 justify-content 相反，主要以交錯軸線來做排版。

aside菜單分為二級。

el-menu

el-submenu

el-menu-item

element網上的是三級菜單, 我們要自己刪走最深的一級,

因為我們只要用到二級。

通過接口獲取菜單數據 axios.interceptors.request.use()

需要授權的API, 必須在請求頭中使用 Authorization字段提供 token令牌。

通過axios請求攔截器添加token, 保證擁有獲取數據的權限。



左側菜單權限 利用get



get 返回的是promise 為了簡化保密操作,我們用await和async。

每一個v-for 都盡量提供一個v-bind:key 的值

https://www.zhihu.com/question/61064119

:unique-opened="true"

要加v-bind: 不然 true的值就會是string, 會報錯。

加了v-bind就成了boolean值。

letter-spacing: 0.2em;

css的, 字符之間開放空間0.2em。

collapse-transition 和 collapse 都是Element-UI內navmanu寫好的屬性。

自定義屬性給值時記得加v-bind。 如果沒有v-bind 屬性值true/false就成了字符串。

<el-aside :width="isCollapse ? '60px' : '200px'">

根據isCollapse 的true/false 決定抑疊側邊菜單大小。

{

        path: '/home',

        component: Home,

        redirect: '/welcome',

        children: [

            { path: '/welcome', component: Welcome, }

        ]

    }

Home頁面主體右邊的Main內的router-view 都要寫在/home的children下, 才能渲染到 home的router-view內。

Menu Attribute

router 是否使用 vue-router 的模式，啟用該模式會在激活導航時以 index 作為 path 進行路由跳轉 boolean — false。

    <el-menu …………… :router="true">

我們改造, 以index作為path進行路由跳轉, 因為我們的MySQL的data中有寫好了path數據,所以把path數據放到二級菜單的index內。

<el-menu-item :index="'/' + subItem.path" v-for="subItem in item.children" :key="subItem.id">

利用element 的layout布局 <el-row>和<el-col span=”份數”> element-ui的布局一行24格(份)。

Row 組件 提供 gutter 屬性來指定每一欄之間的間隔，默認間隔為 0。

*// //全局導入*

import '../../node\_modules/element-ui/lib/theme-chalk/index.css'

import elementUI from 'element-ui'

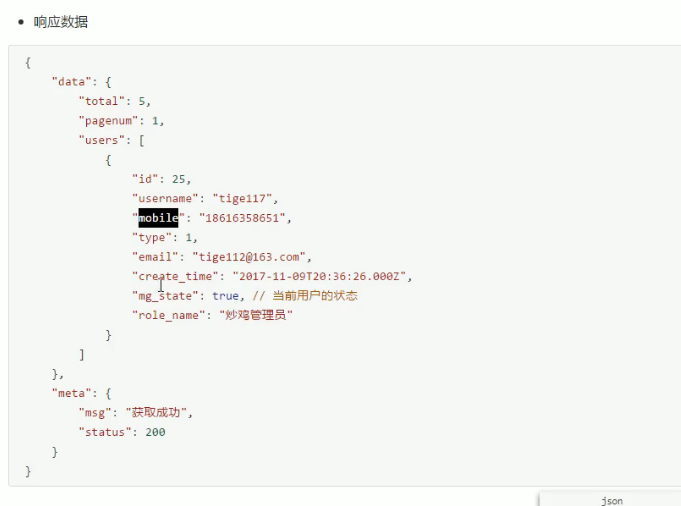
Vue.use(elementUI)

全局導入element-ui 要同時導入它的css , 記得Vue.use(elementUI)

用戶數據列表



響應數據



*<!-- 用戶列表渲染區 -->*

        <el-table :data="userList" border stripe>

            <el-table-column type="index" label="#"></el-table-column>

            <el-table-column label="姓名" prop="username"></el-table-column>

            <el-table-column label="郵箱" prop="email"></el-table-column>

            <el-table-column label="電話" prop="mobile"></el-table-column>

            <el-table-column label="角色" prop="role\_name"></el-table-column>

            <el-table-column label="狀態" prop="mg\_state"></el-table-column>

            <el-table-column label="操作"></el-table-column>

        </el-table>

因為我們在data(){}中有userList接收了數據庫中的數據, 所以我們

在table中:data=”userList” 取數據, 再在子標簽用prop傳值。vue用法。

v-slot 作用域插槽

\*\*\*\*\*

因為elementUI組件是封裝好了的,要改變里面的內容必須用作用域插槽。

Slot 是一種內容分發(content distribution)的 API，中文翻譯為插槽，適合用在結構比較複雜，元件內容可以重複使用的地方。簡單說就是在 component 中可以預留空間，在父層再把內容放進去。

<el-table-column label="狀態" prop="mg\_state">

                <template v-slot:default="scope">

                    {{scope.row}}

                </template>

</el-table-column>

Switch開關下有event用

change switch 状态发生变化时的回调函数 新状态的值

@change事件

const {data:res} =await this.$http.put(`users/${*userinfo*.id}/state/${*userinfo*.mg\_state}`)

{data:res}

解構數據data 重命名為res。

Input Attributes

clearable 是否可清空 boolean — false

### Input Events

clear 在点击由 clearable 属性生成的清空按钮时触发

slot

父元件內的宿主內容會被放入子元件的<slot>中,只有當父元件內的宿主內容為空時,才會顯示子元件中<slot>的備用內容。

https://cythilya.github.io/2017/10/11/vue-component-slot/



**要注意v-slot只能添加在<template>上**

讓父層在指定 slot 內容的時候可以指定需要的 component 的 data。





添加用戶

<el-form :model="addForm" :rules="addFormRules" ref="addFormRef" label-width="70px" class="demo-ruleForm">

<el-form-item label="用戶名" prop="username">

          <el-input v-model="addForm.username"></el-input>

    </el-form-item>

</el-form>

Form 組件提供了表單驗證的功能，只需要通過 rules 屬性傳入約定的驗證規則，並將 Form-Item 的 prop 屬性設置為需校驗的字段名即可。

:model=”addForm” 是綁定的數據 (數據綁定對象)

:rules 是提示message的規則

驗證email和手機號用 正則表達式

var checkMobile = (*rule*, *value*, *callback*) => {

*//驗證手機號的正則表達式*

            const regMobile = /^(0|86|17951)?(13[0-9]|15[012356789]|17[678]|18[0-9]|14[57])[0-9]{8}$/

            if (regMobile.test(*value*)) {

                return callback()

            }

            callback(new Error('請輸入合法的手機號碼'))

        }

mobile: [{

                    required: true,

                    message: '請輸入郵箱',

                    trigger: 'blur'

                }, {

                    validator: checkMobile,

                    trigger: 'blur'

                }]

v-slot=“scope”

作用域插槽中定义一个对象(这里对象被定义为scope)来存储插槽上绑定的数据的用法

scope.row

使用ElementUI表格模板渲染数据时使用

总体上说明：

当前行数据的获取也会用到插槽,scope相当于一行的数据， scope.row相当于当前行的数据对象

————————————————

上傳到github

git checkout –b user 開創user分支

git add .

git commit –m “完成用戶列表功能”

git push –u origin user

git checkout master 換到master分支

git merge user 合并user分支的內容到master分支

git push 推送合并了的master分支到github。

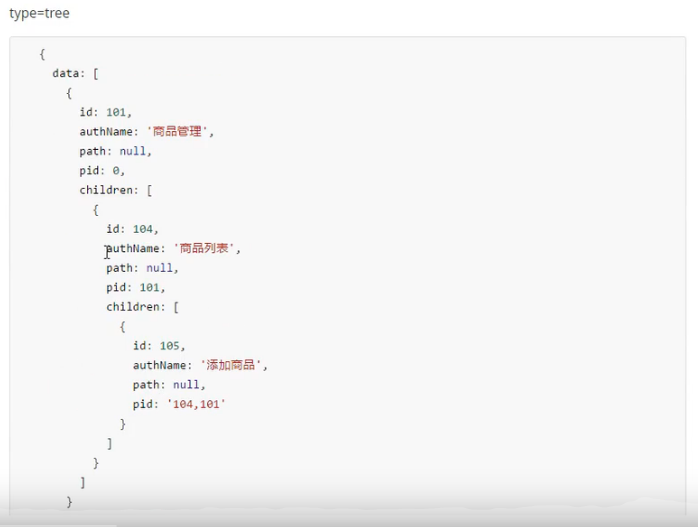
git checkout –u rights 開創rights分支

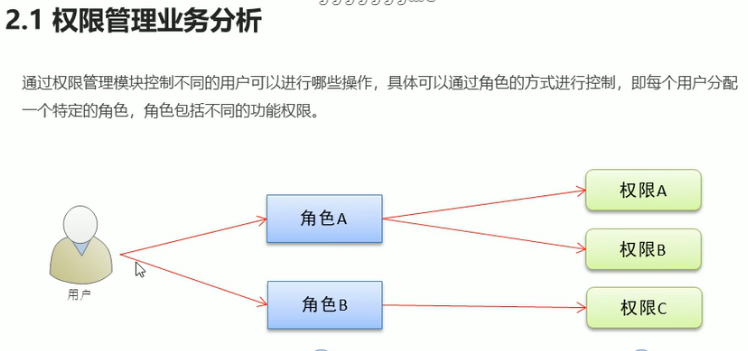
權限列表

權限列表api









<el-table-column type="expand"></el-table-column>

type=”expand” 是一個可展开的按钮

分配權限btn的dialog利用了element ui 的tree和tree api接口。

<el-tree :data="rightsList" :props="treeProps">

tree 控件的綁定對象

defaultProps: {

children: 'children',

label: 'label'

}

tree控件下有好多有用的attribute屬性

比如

default-expand-all

show-checkbox

node-key=”id”

\*\* :default-checked-key=”defKeys” 打開時自動checked已存在的key,。

<el-tree :data="rightsList" :props="treeProps" show-checkbox node-key="id" default-expand-all :default-checked-keys="defKeys"></el-tree>

\*\*\*\*\*

遞歸函數

*// 通過遞歸的形式, 獲取角色下所有三級權限的id, 保存到defKeys中。*

        getLeafKeys(*node*, *arr*) {

*// 如果當前 node節點不包含children屬性 就是三級節點*

            if(!*node*.children) {

                return *arr*.push(*node*.id);

            }

*// 遞歸*

*node*.children.forEach(*item* => this.getLeafKeys(*item*, *arr*))

        },

分配權限的提交

利用tree組件下的attribute

getHalfCheckedKeys和getCheckedKeys 全選和半選的checkedkeys

分類管理