**Vue-next-admin后台管理系统**

# 概述

* PandaUi

本项目来源基于开源PandaUi进行二次开发，PandaUi是纯前端项目

[https://gitee.com/PandaAdmin/PandaUi](https://gitee.com/PandaAdmin/PandaUi#%E5%B9%B3%E5%8F%B0%E5%AE%98%E7%BD%91)

* [vue-next-admin](https://lyt-top.gitee.io/vue-next-admin-doc-preview/)

而PandaUi是基于[vue-next-admin](https://lyt-top.gitee.io/vue-next-admin-doc-preview/)修改的：<https://gitee.com/lyt-top/vue-next-admin>

[vue-next-admin](https://lyt-top.gitee.io/vue-next-admin-doc-preview/)也是纯前端框架。

* PandaX

Panda还有个完整的系统PandaX，前端用的是PandaUi，后端用的是go语言开发。

* 本项目

本项目后台没有用go，数据库sql脚本复制来自Pandax

另外，本项目还集成了vxe-grid（vxe-table）表格，包括相应的后台访问插件XEAjax等

# 学习文档

1.Vue-next-admin在线文档：

<https://lyt-top.gitee.io/vue-next-admin-doc-preview/>

2.PandaUi在线文档：

https://pandaadmin.gitee.io/pandax-docs/guide/

3.其他相关项目开发文档

* [vue3 文档地址：https://v3.cn.vuejs.org/](https://gitee.com/link?target=https://v3.cn.vuejs.org/)
* [vue3 composition-Api 文档地址：https://composition-api.vuejs.org/zh/](https://gitee.com/link?target=https://composition-api.vuejs.org/zh/)
* [vue2 文档地址：https://cn.vuejs.org/](https://gitee.com/link?target=https://cn.vuejs.org/)
* [element-plus 文档地址：https://element-plus.gitee.io/zh-CN/#/zh-CN](https://element-plus.gitee.io/zh-CN/" \l "/zh-CN)
* [element 文档地址：https://element.eleme.cn/#/zh-CN](https://gitee.com/link?target=https://element.eleme.cn/%23/zh-CN)
* [vue-router-next 文档地址：https://next.router.vuejs.org/](https://gitee.com/link?target=https://next.router.vuejs.org/)
* [router 文档地址：https://router.vuejs.org/zh/](https://gitee.com/link?target=https://router.vuejs.org/zh/)
* [next.vuex 文档地址：https://next.vuex.vuejs.org/](https://gitee.com/link?target=https://next.vuex.vuejs.org/)
* [vuex 文档地址：https://vuex.vuejs.org/zh/](https://gitee.com/link?target=https://vuex.vuejs.org/zh/)
* [vue cli 文档地址：https://cli.vuejs.org/zh/](https://gitee.com/link?target=https://cli.vuejs.org/zh/)
* [vite 文档地址：https://vitejs.cn/](https://gitee.com/link?target=https://vitejs.cn/)
* [typescript 文档地址：https://www.tslang.cn/docs/home.html](https://gitee.com/link?target=https://www.tslang.cn/docs/home.html)
* [jsplumb 文档地址：https://wdd.js.org/jsplumb-chinese-tutorial/#/](https://gitee.com/link?target=https://wdd.js.org/jsplumb-chinese-tutorial/%23/)

二、文档编写

* [vitepress 文档地址：https://vitepress.vuejs.org/](https://gitee.com/link?target=https://vitepress.vuejs.org/)
* [vuepress 文档地址：https://vuepress.vuejs.org/zh/](https://gitee.com/link?target=https://vuepress.vuejs.org/zh/)

# 下载安装

## 下载

地址：<https://gitee.com/lyt-top/vue-next-admin>

## 安装运行

1. 过程：



上面是说明

下面是步骤：

1. 安装cnpm和yarn（这个我以前已经安装过了）

npm install -g cnpm --registry=https://registry.npm.taobao.org

npm install -g yarn

1. 下载安装软件

# 克隆项目

git clone https://gitee.com/lyt-top/vue-next-admin.git

然后会有新文件夹vue-next-admin，里面就是代码

# 进入项目

cd vue-next-admin

再安装依赖

# 安装依赖

cnpm install

（3）运行或者打包

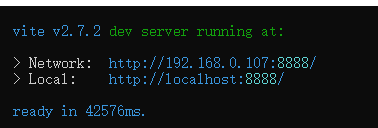
# 运行项目

cnpm run dev

#打包发布

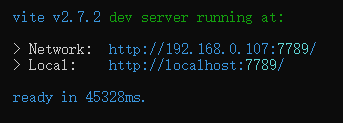
cnpm run build

如果cnpm run dev 运行的话：界面会出现：



浏览器运行上面的地址即可

另外一个项目：git clone <https://github.com/PandaGoAdmin/PandaUi.git>



## 配置后台完整登录运行

上面运行起来，界面上的验证码不会正确显示，通过浏览器调试可以知道，调用的后台是127.0.0.1:7788的，由于本地没有后台运行，因此出错。

这里先配置网上案例的后台地址，即本地运行前端，后台借用网上。

修改2个地方：（注意，网上地址：http://103.145.39.46:8080）

1.修改VITE\_API\_URL地址，在文件：.env.development中

VITE\_API\_URL = 'http://103.145.39.46:8080'

2.修改代理proxy：vite.config.ts

server: {

        host: '0.0.0.0',

        port: VITE\_PORT,

        open: VITE\_OPEN,

        proxy: {

            '/api': {

                target: 'http://103.145.39.46:8080/',

                ws: true,

                changeOrigin: true,

                rewrite: (path) => path.replace(/^\/api/, '/'),

            },

        },

        //old one

        /\* proxy: {

            '/api': {

                target: 'http://127.0.0.1:7788',

                ws: true,

                changeOrigin: true,

                rewrite: (path) => path.replace(/^\/api/, '/'),

            },

        }, \*/

    },

# 二次开发

## 开发环境

下载安装vscode

在vscode里：File-->Open Folder... --->选择项目文件夹

## 公共模块

### 环境

1 通用环境：.env

这里设置端口、打包路径、运行的时候是否打开浏览器：

# port 端口号,网站端口

VITE\_PORT = 7789

# open 运行 npm run dev 时自动打开浏览器

VITE\_OPEN = true

# public path 配置线上环境路径（打包）、本地通过 http-server 访问时，请置空即可

VITE\_PUBLIC\_PATH = ./

2.开发环境（本地调试发布）：.env.development

# 本地环境

ENV = 'development'

# 本地环境接口地址

# VITE\_API\_URL = 'http://localhost:7789'

VITE\_API\_URL = 'http://103.145.39.46:8080'

说明：VITE\_API\_URL就表示后台的baseURL的路径

3.部署环境：.env.production

# 线上环境

ENV = 'production'

# 线上环境接口地址

VITE\_API\_URL = 'http://103.145.39.46:8080'

说明：VITE\_API\_URL就表示后台的baseURL的路径

特别注意：修改了配置的话，需要重新运行网站才能起效果：

在命令行里先Ctrl C 停止原来的运行，在执行下面命令行

# 运行项目

cnpm run dev

特别注意：复制了一个xxx.vue在同一个文件夹里，那么需要把这个文件放入另外不相干的文件夹里，可能因为里面的内容相似导致不能正确显示，移走复制的那个就可以了。

### 后台地址baseURL

src\utils\request.ts

const service = axios.create({

    //baseURL: 'http://localhost:53605/api/',

    baseURL: import.meta.env.VITE\_API\_URL as any,

    timeout: 50000,

    headers: { 'Content-Type': 'application/json' },

});

注意系统的api路径（后台路径）就是由基本路径baseURL+接口的后缀名称，例如：

[www.abc.com/user/signIn，](http://www.abc.com/user/signIn，)那么baseURL为www.abc.com，api后缀为/user/signIn

### 跨域配置

#### 跨域问题

即使本地运行前后台，也会有跨域问题，因为前端的端口和后台的端口不能一样（同一台电脑，不同电脑域名或者ip肯定不一样），所以会出现访问不了后台的错误，浏览器调试提示：

No 'Access-Control-Allow-Origin' header is present on the requested resource等

* 跨域其实可以客户端处理，就是下面的代理配置：[使用vue的代理配置](#_使用vue的代理配置)
* 也可以服务器端处理，比如用nigx，iis上的响应标头配置等

#### 使用vue的代理配置

vite.config.ts

1.下面的配置是把你本地运行的前端连接到发布者网上的后台。你在自己没有后台的情况下可以这么做。

server: {

        host: '0.0.0.0',

        port: VITE\_PORT,

        open: VITE\_OPEN,

        proxy: {

            '/api': {

                target: 'http://103.145.39.46:8080/',

                ws: true,

                changeOrigin: true,

                rewrite: (path) => path.replace(/^\/api/, '/'),

            },

        },

        //old one

        /\* proxy: {

            '/api': {

                target: 'http://127.0.0.1:7788',

                ws: true,

                changeOrigin: true,

                rewrite: (path) => path.replace(/^\/api/, '/'),

            },

        }, \*/

    },

2.配置你自己的本地后台：例如我的是这样的：

server: {

        host: '0.0.0.0',

        port: VITE\_PORT,

        open: VITE\_OPEN,

        proxy: {

            '/api': {

                target: 'http://localhost:4498',//我本地后台

                ws: true,

                changeOrigin: true,

                //

                rewrite: (path) => path.replace(/^\/api/, '/'),

            },

        },

特别注意：.env.development下面需要修改VITE\_API\_URL

# 本地环境

ENV = 'development'

# 本地环境接口地址

#old one

# VITE\_API\_URL = 'http://127.0.0.1:7788'

# 注意：1.http://localhost:7789/api这个地址是访问后台的，

#2.这里要带个'/api'，因为vite.config.ts里面配置的proxy用这个进行匹配，然后根据rewrite去掉api，就用代理的地址来访问后台了

VITE\_API\_URL = 'http://localhost:7789/api'

设置完毕需要重启网站（到发布的命令行里ctrl c停止，再运行cnpm run dev）

#### 跨域其他参考

跨域其他参考：

<https://blog.csdn.net/qq_34450741/article/details/107444815>

### 本地资源目录

Assets文件夹

### 全局配置

src\store\modules\themeConfig.ts

包括网站title，副标题，全局语言，水印，布局配置等

去掉或者修改水印（默认有）：

isWartermark: false,

wartermarkText: 'PandaX',

这些改过以后需要重启服务或者刷新页面才能看到

### 消息弹出框

有ElMessage和ElMessageBox

ElMessage.error('网络超时')

有warning，info，success等

ElMessageBox.confirm等

## 路由模块

### 路由元数据

/\*\*

 \* 路由meta对象参数说明

 \* meta: {

 \*      title:          菜单栏及 tagsView 栏、菜单搜索名称（国际化）

 \*      isLink：        是否超链接菜单，开启外链条件，`1、isLink:true 2、链接地址不为空`

 \*      isHide：        是否隐藏此路由

 \*      isKeepAlive：   是否缓存组件状态

 \*      isAffix：       是否固定在 tagsView 栏上

 \*      isIframe：      是否内嵌窗口，，开启条件，`1、isIframe:true 2、链接地址不为空`

 \*      auth：          当前路由权限标识（多个请用逗号隔开），最后转成数组格式，用于与当前用户权限进行对比，控制路由显示、隐藏

 \*      icon：          菜单、tagsView 图标，阿里：加 `iconfont xxx`，fontawesome：加 `fa xxx`

 \* }

 \*/

菜单的结构定义：

   children: [

            {

                path: '/fun',

                name: 'funIndex',

                component: () => import('/@/layout/routerView/parent.vue'),

                redirect: '/fun/tagsView',

                meta: {

                    title: 'message.router.funIndex',

                    isLink: '',

                    isHide: false,

                    isKeepAlive: true,

                    isAffix: false,

                    isIframe: false,

                    auth: ['base'],

                    icon: 'iconfont icon-crew\_feature',

                },

### 静态路由

staticPageRoutes，静态路由就是写死的菜单

### 动态路由

动态路由就是从后台读取的该用户的菜单json然后赋值给：

覆盖 dynamicRoutes 第一个顶级 children 的内容

## 登录模块

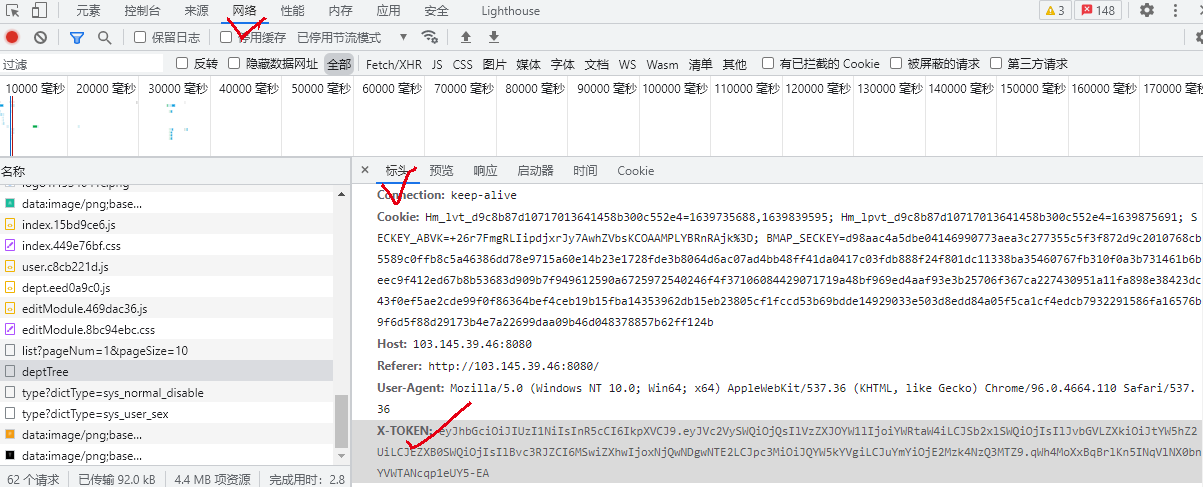
### Token

流程：用户在前端请求登录——>在后台验证通过后根据用户ID生成token——>请求返回时将token带给前端并存在前端——>以后前端再次请求时需要带上token【在请求头里】

——>后台接受到token并验证，如果验证通过，返回true，否则false（实际情况下就是验证通过，这次请求成功，否则这次请求无效）

1.登录的时候服务器返回token内容到客户端（一次登录产生一次），后续请求只要带上token，无需再用户名和密码，为了减轻服务器压力

通过浏览器调试查看后续请求：



这里前台设置的key：X-TOKEN

我后来在前台改为AUTH-TOKEN了

**2.JWT：**Json web token (JWT), 是为了在网络应用环境间传递声明而执行的一种基于JSON的开放标准（[(RFC 7519](https://link.jianshu.com/?t=https://tools.ietf.org/html/rfc7519" \t "https://www.cnblogs.com/ckfuture/p/_blank)).该token被设计为紧凑且安全的，特别适用于分布式站点的单点登录（SSO）场景

### 页面

src\views\login\index.vue

登录背景由网上改为本地图片

### 登录显示信息配置

简体中文是zh-ch.ts: src\i18n\pages\login\zh-cn.ts

这里可以定制登录界面，比如去掉手机登录等功能

### 账号密码登录

#### 页面文件

Account.vue

#### 验证码

项目中验证码是死的，不会生成，有2种方法：1.纯前端（不安全）2.后端生成验证码和图片。

本项目我添加的是后端生成验证码及图片，并且后台进行校验。

##### 存前端

这里采用最简单的随机生成4位数的方式

工作：

1. 添加生成验证码函数：RandomNum() 在src\utils\commonFunction.ts里面
2. 添加import：

import commonFunction from '/@/utils/commonFunction';

以及：setup()里面：

const { RandomNum } = commonFunction();

1. 修改原来的验证码显示：

<span class="login-content-code-img">1234</span>

改为：

<el-button id=validCodeBtn class="login-button-code"

        @click="onChangeCode"             >

                    <div >

                        <span>{{initCode }}</span>

                        </div>

                    </el-button>

加button的原因是因为点击可以改变验证码

1. 添加initCode 对应的代码：

（1）setup()

//初始验证码

        const initCode = RandomNum();

（2）返回的地方

return {

            currentTime,

            onSignIn,

            onChangeCode,

            initCode,

            ...toRefs(state),

        };

1. 添加onChangeCode函数（上面return以及添加了onChangeCode）：

const onChangeCode = (value: any = 0) => {

            var code = RandomNum(); //生成验证码

            var codeBtn = document.getElementById("validCodeBtn") as HTMLElement;

            if(codeBtn != null)

                codeBtn.innerText = code;

            return 0;

        };

1. 验证验证码是否正确输入：

var codeBtn = document.getElementById("validCodeBtn") as HTMLElement;

            var validCode = codeBtn.innerText;

            if(state.ruleForm.code != validCode)

            {

                alert("验证码错误");

                return;

            }

##### 后台生成

具体看代码。略

#### 登录后台交互

## 表格页面

典型的，一个表格页面由以下部分组成，下面以岗位管理（post单词表示岗位）为例说明

### 页面界面

1.表格页面：

src\views\system\post\index.vue

<template></template>是元素

<script></script>是脚本，具体解释：（略）

2.编辑页面：

src\views\system\post\component\editModule.vue

### 后台接口

src\api\system\post.ts

有5个方法：

查询表格：listPost

查询详情：getPost

添加：addPost

修改：updatePost

删除：delPost

## Vxe表格使用

安装内容：

1.命令行进入项目文件夹，例如我的F:\MyProject\PandUi

2.运行：（建议下面统一都用cnpm（在有npm的地方））

cnpm install xe-ajax

//cnpm install xe-utils

//cnpm install vxe-table@next

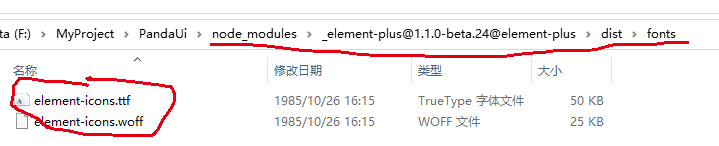
或者：//npm install xe-utils vxe-table@next

这里一次性：

npm install xe-utils vxe-table@next vxe-table-plugin-element@next element-plus

npm install font-awesome

用于表格的一些图标，但是上面安装了后，运行，浏览器调试发现，少了



这2个文件，应该是安装的时候同时自动删除了一些其他东西，解决办法是在根目录上搜索这2个文件，复制到这个目录下（带路径）

网上有人说这样安装（我没试过）

npm install less less-loader css-loader style-loader file-loader font-awesome --save

npm install --save @fortawesome/fontawesome-svg-core

npm install --save @fortawesome/vue-fontawesome

npm install --save @fortawesome/free-solid-svg-icons

【特别注意，如果cnpm install vxe-table那么安装完毕会有告警，说版本只支持vue2.6.0，而我目前安装的vue3.2.26，建议你安装最新版本，加上@next就不会有告警了。】

在main.ts中：

1.import

import 'font-awesome/scss/font-awesome.scss' //一些图标

import 'xe-utils'

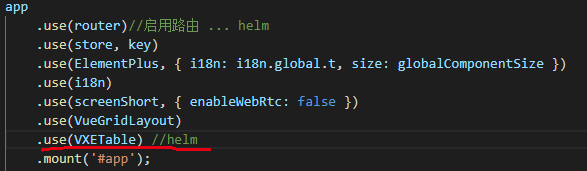
import VXETable from 'vxe-table'

import 'vxe-table/lib/style.css'

//import VXETablePluginElement from 'vxe-table-plugin-element'

//import 'vxe-table-plugin-element/dist/style.css'

2.use



还没有用，也没axios 安装：npm install axios -S

如果发生类似下面的错误：（浏览器调试提示的错误）

GET http://localhost:7789/@vite/client net::ERR\_ABORTED 404 (Not Found)

可能vite被破坏了，那么重新安装：

npm install -g create-vite-app

然后重启服务：cnpm run dev

服务器端跨域配置：

注意，这里使用

return XEAjax.get(baseUrl + `/JiaoBanZhuang\_Device/GetVueTableData?rows=${page.pageSize}&page=${page.currentPage}`, queryParams)

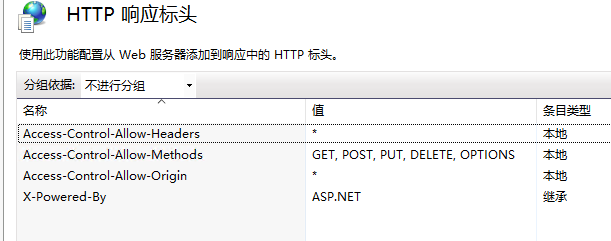
其中

const baseUrl= "https://www.zhjtypt.com/wldevice/";

也会产生跨域问题（浏览器提示：cors错误），需要在服务器端（我这里是IIS中设置）：



我的是放在Default Web Site（这个不需要设置）下的应用程序wldevice，只需要设置wldevice下http响应标头：



特别注意：

属性：

height: 700,

只要是固定值，会导致页面挣脱了外层框架

所以不设置就好了

其实没最后好，还有其他错误，其他错误排除后，还是会挣脱外层框架！

后来查到了，是vxe-grid的最大化了（全屏了），去掉代码中的：

this.$refs.xGrid.maximize()

就可以了。默认情况下是“还原”状态，不是全屏。（折腾很久）

## 权限管理

这是admin账户登录的时候返回的后台数据：【菜单部分可以参考路由元数据：[路由元数据](#_路由元数据)】

{"code":200,"msg":"success","data":{"expire":1640172580,"menus":[{"name":"/system","path":"/system","redirect":"","component":"Layout","meta":{"title":"系统设置","isLink":"","isHide":false,"isKeepAlive":true,"isAffix":false,"isFrame":false,"auth":[],"icon":"el-icon-s-tools"},"children":[{"name":"/system/role","path":"/system/role","redirect":"","component":"/system/role/index","meta":{"title":"角色管理","isLink":"","isHide":false,"isKeepAlive":false,"isAffix":false,"isFrame":false,"auth":["system:role:list"],"icon":"el-icon-user-solid"},"children":[]},{"name":"/system/user","path":"/system/user","redirect":"","component":"/system/user/index","meta":{"title":"用户管理","isLink":"","isHide":false,"isKeepAlive":false,"isAffix":false,"isFrame":false,"auth":["system:user:list"],"icon":"el-icon-user"},"children":[]},{"name":"/system/api","path":"/system/api","redirect":"","component":"/system/api/index","meta":{"title":"API管理","isLink":"","isHide":false,"isKeepAlive":false,"isAffix":false,"isFrame":false,"auth":["system:api:list"],"icon":"iconfont icon-siweidaotu"},"children":[]},{"name":"/system/menu","path":"/system/menu","redirect":"","component":"/system/menu/index","meta":{"title":"菜单管理","isLink":"","isHide":false,"isKeepAlive":false,"isAffix":false,"isFrame":false,"auth":["system:menu:list"],"icon":"el-icon-menu"},"children":[]},{"name":"/system/dept","path":"/system/dept","redirect":"","component":"/system/dept/index","meta":{"title":"部门管理","isLink":"","isHide":false,"isKeepAlive":false,"isAffix":false,"isFrame":false,"auth":["system:dept:list"],"icon":"el-icon-set-up"},"children":[]},{"name":"/system/post","path":"/system/post","redirect":"","component":"/system/post/index","meta":{"title":"岗位管理","isLink":"","isHide":false,"isKeepAlive":false,"isAffix":false,"isFrame":false,"auth":["system:post:list"],"icon":"el-icon-s-flag"},"children":[]},{"name":"/system/dict","path":"/system/dict","redirect":"","component":"/system/dict/index","meta":{"title":"字典管理","isLink":"","isHide":false,"isKeepAlive":false,"isAffix":false,"isFrame":false,"auth":["system:dict:list"],"icon":"el-icon-s-order"},"children":[]},{"name":"/system/config","path":"/system/config","redirect":"","component":"/system/config/index","meta":{"title":"参数管理","isLink":"","isHide":false,"isKeepAlive":false,"isAffix":false,"isFrame":false,"auth":["system:config:list"],"icon":"el-icon-document"},"children":[]}]},{"name":"/personal","path":"/personal","redirect":"","component":"/personal/index","meta":{"title":"个人中心","isLink":"","isHide":false,"isKeepAlive":true,"isAffix":true,"isFrame":false,"auth":[],"icon":"el-icon-s-custom"},"children":[]}],"permissions":["system:user:list","system:user:add","system:user:edit","system:role:list","system:menu:list","system:dept:list","system:post:list","system:dict:list","system:config:list","system:config:add","system:config:edit","system:config:delete","system:config:export","system:role:add","system:role:delete","system:role:edit","system:role:export","system:menu:add","system:menu:edit","system:menu:delete","system:dept:add","system:dept:edit","system:dept:delete","system:post:add","system:post:edit","system:post:delete","system:post:export","system:dictT:add","system:dictT:edit","system:dictT:delete","system:dictT:export","system:dictD:add","system:dictD:edit","system:dictD:delete","system:api:list","system:api:add","system:api:edit","system:api:delete"],"token":"eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJVc2VySWQiOjQsIlVzZXJOYW1lIjoiYWRtaW4iLCJSb2xlSWQiOjIsIlJvbGVLZXkiOiJtYW5hZ2UiLCJEZXB0SWQiOjIsIlBvc3RJZCI6MSwiZXhwIjoxNjQwMTcyNTgwLCJpc3MiOiJQYW5kYVgiLCJuYmYiOjE2Mzk1NjY3ODB9.76Tr-FrUVJmrD6FJhs0Tnw2HCRSeVIKqZDBR2h0Z9bE","user":{"userId":4,"nickName":"Pandax","phone":"","roleId":2,"salt":"","avatar":"","sex":"0","email":"2417920382@qq.com","deptId":2,"postId":1,"roleIds":"2","postIds":"1,4","createBy":"admin","updateBy":"1","remark":"","status":"0","create\_time":"2021-12-03T09:46:55Z","update\_time":"2021-12-14T12:22:55Z","username":"admin","password":"$2a$10$mSw4ioPf6adomWNwDHjBNORgua3Z0Tll1DM7O2Dgyaj3mIKT8kocG"}}}

这是test2账户登录时（只有2个菜单，岗位管理和个人中心叶子节点）返回的后台数据：

{"code":200,"msg":"success","data":{"expire":1640232463,"menus":[{"name":"/system","path":"/system","redirect":"","component":"Layout","meta":{"title":"系统设置","isLink":"","isHide":false,"isKeepAlive":true,"isAffix":false,"isFrame":false,"auth":[],"icon":"el-icon-s-tools"},"children":[{"name":"/system/post","path":"/system/post","redirect":"","component":"/system/post/index","meta":{"title":"岗位管理","isLink":"","isHide":false,"isKeepAlive":false,"isAffix":false,"isFrame":false,"auth":["system:post:list"],"icon":"el-icon-s-flag"},"children":[]}]},{"name":"/personal","path":"/personal","redirect":"","component":"/personal/index","meta":{"title":"个人中心","isLink":"","isHide":false,"isKeepAlive":true,"isAffix":true,"isFrame":false,"auth":[],"icon":"el-icon-s-custom"},"children":[]}],"permissions":["system:post:list","system:post:add","system:post:edit","system:post:delete","system:post:export"],"token":"eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJVc2VySWQiOjUsIlVzZXJOYW1lIjoidGVzdDIiLCJSb2xlSWQiOjMsIlJvbGVLZXkiOiJjb21tb24iLCJEZXB0SWQiOjIsIlBvc3RJZCI6MCwiZXhwIjoxNjQwMjMyNDYzLCJpc3MiOiJQYW5kYVgiLCJuYmYiOjE2Mzk2MjY2NjN9.Zi272E\_Oc6iGcllw7bUeRS0zTApTPveB64GjrJ\_kZvg","user":{"userId":5,"nickName":"test2","phone":"","roleId":3,"salt":"","avatar":"","sex":"","email":"","deptId":2,"postId":0,"roleIds":"3","postIds":"","createBy":"test","updateBy":"","remark":"","status":"0","create\_time":"2021-12-16T04:05:53Z","update\_time":"2021-12-16T04:06:29Z","username":"test2","password":"$2a$10$gn1OxmDlLqIWHWu.2faT.OUU0QTJe85PbJxV/4WfsxHpF1pOD.Kv6"}}}

另外：上面的：

"auth":["system:post:list"],"icon":"el-icon-s-flag"},"children":[]

system:post:list表示post可读？因为权限：list ？

## 菜单

1.数据获取

显示的菜单受控于登录的时候后台返回的菜单项（可见）

2.前台存储

后台数据放入客户端（浏览器）临时缓存Session对象中，Session定义在：src\utils\storage.ts【参加存储章节解释】，

菜单数据在src\views\login\component\account.vue中存入Session对象：

Session.set('menus', loginRes.menus);

3.读取显示

src\router\index.ts

let menuRoute = Session.get('menus')

然后连同静态菜单（写死的）放到:drs[0].children

drs[0].children = drs[0].children?.concat(backEndRouterConverter(menuRoute))

显示？？这里不理解：

store.dispatch('routesList/setRoutesList', drs[0].children);

4.如果没有，再次后端读取

if (!menuRoute) {//没有获取到 he

        menuRoute = getBackEndControlRoutes(); // 去后端获取路由

    }

这个最后会调用src\api\menu\index.ts中的getRouters去后端获取

问题：这里没有传用户信息（难道后端根据session读取？），因为菜单需要用户信息才能读到（用户权限）

// 获取路由

export const getRouters = () => {

    return request({

        url: '/getRouters',

        method: 'get'

    })

}

5.菜单链接

权限管理里面后台返回的数据中有一项：

{"name":"/system/post","path":"/system/post","redirect":"","component":"/system/post/index","meta":{"title":"岗位管理","isLink":"","isHide":false,"isKeepAlive":false,"isAffix":false,"isFrame":false,"auth":["system:post:list"],"icon":"el-icon-s-flag"},"children":[]}

上面的/system/post就是链接（不确定？），指向：src\views\system\post\index.vue

# 后台数据格式

## 登录

### 验证码

获取验证码：system/user/getCaptcha 方式：get

{"base64Captcha":"data:image/png;base64,","captchaId":"1zO86oZzuKkE6ovqdrok"}

### 登录用户

获取用户：system/user/login post方式：

参数：{

"username": "admin",

"password": "123456",

"captcha": "HJ00",

"captchaId": "SEowMA=="

}

返回结果：

返回错误：

{"code":400,"msg":"验证码认证失败","data":null}

正确：

{"code":200,"msg":"success","data":{"expire":1640401121,"menus":[{"name":"/system","path":"/system","redirect":"","component":"Layout","meta":{"title":"系统设置","isLink":"","isHide":false,"isKeepAlive":true,"isAffix":false,"isFrame":false,"auth":[],"icon":"el-icon-s-tools"},"children":[{"name":"/system/api","path":"/system/api","redirect":"","component":"/system/api/index","meta":{"title":"API管理","isLink":"","isHide":false,"isKeepAlive":false,"isAffix":false,"isFrame":false,"auth":["system:api:list"],"icon":"iconfont icon-siweidaotu"},"children":[]},{"name":"/system/role","path":"/system/role","redirect":"","component":"/system/role/index","meta":{"title":"角色管理","isLink":"","isHide":false,"isKeepAlive":false,"isAffix":false,"isFrame":false,"auth":["system:role:list"],"icon":"el-icon-user-solid"},"children":[]},{"name":"/system/user","path":"/system/user","redirect":"","component":"/system/user/index","meta":{"title":"用户管理","isLink":"","isHide":false,"isKeepAlive":false,"isAffix":false,"isFrame":false,"auth":["system:user:list"],"icon":"el-icon-user"},"children":[]},{"name":"/system/menu","path":"/system/menu","redirect":"","component":"/system/menu/index","meta":{"title":"菜单管理","isLink":"","isHide":false,"isKeepAlive":false,"isAffix":false,"isFrame":false,"auth":["system:menu:list"],"icon":"el-icon-menu"},"children":[]},{"name":"/system/dept","path":"/system/dept","redirect":"","component":"/system/dept/index","meta":{"title":"部门管理","isLink":"","isHide":false,"isKeepAlive":false,"isAffix":false,"isFrame":false,"auth":["system:dept:list"],"icon":"el-icon-set-up"},"children":[]},{"name":"/system/post","path":"/system/post","redirect":"","component":"/system/post/index","meta":{"title":"岗位管理","isLink":"","isHide":false,"isKeepAlive":false,"isAffix":false,"isFrame":false,"auth":["system:post:list"],"icon":"el-icon-s-flag"},"children":[]},{"name":"/system/dict","path":"/system/dict","redirect":"","component":"/system/dict/index","meta":{"title":"字典管理","isLink":"","isHide":false,"isKeepAlive":false,"isAffix":false,"isFrame":false,"auth":["system:dict:list"],"icon":"el-icon-s-order"},"children":[]},{"name":"/system/config","path":"/system/config","redirect":"","component":"/system/config/index","meta":{"title":"参数管理","isLink":"","isHide":false,"isKeepAlive":false,"isAffix":false,"isFrame":false,"auth":["system:config:list"],"icon":"el-icon-document"},"children":[]}]},{"name":"/personal","path":"/personal","redirect":"","component":"/personal/index","meta":{"title":"个人中心","isLink":"","isHide":false,"isKeepAlive":true,"isAffix":true,"isFrame":false,"auth":[],"icon":"el-icon-s-custom"},"children":[]}],"permissions":["system:user:list","system:user:add","system:user:edit","system:role:list","system:menu:list","system:dept:list","system:post:list","system:dict:list","system:config:list","system:config:add","system:config:edit","system:config:delete","system:config:export","system:role:add","system:role:delete","system:role:edit","system:role:export","system:menu:add","system:menu:edit","system:menu:delete","system:dept:add","system:dept:edit","system:dept:delete","system:post:add","system:post:edit","system:post:delete","system:post:export","system:dictT:add","system:dictT:edit","system:dictT:delete","system:dictT:export","system:dictD:add","system:dictD:edit","system:dictD:delete","system:api:list","system:api:add","system:api:edit","system:api:delete"],"token":"eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJVc2VySWQiOjQsIlVzZXJOYW1lIjoiYWRtaW4iLCJSb2xlSWQiOjIsIlJvbGVLZXkiOiJtYW5hZ2UiLCJEZXB0SWQiOjIsIlBvc3RJZCI6MSwiZXhwIjoxNjQwNDAxMTIxLCJpc3MiOiJQYW5kYVgiLCJuYmYiOjE2Mzk3OTUzMjF9.BXW9grOYUyBRRtdifU250QvoN\_pdifO1UFTagslSEgE","user":{"userId":4,"nickName":"Pandax","phone":"","roleId":2,"salt":"","avatar":"","sex":"0","email":"2417920382@qq.com","deptId":2,"postId":1,"roleIds":"2","postIds":"1,4","createBy":"admin","updateBy":"1","remark":"","status":"0","create\_time":"2021-12-03T09:46:55Z","update\_time":"2021-12-14T12:22:55Z","username":"admin","password":"$2a$10$mSw4ioPf6adomWNwDHjBNORgua3Z0Tll1DM7O2Dgyaj3mIKT8kocG"}}}