# 一、主界面代码

登录界面代码

<template>

<div class ="all">

<div>

<div class = "logo">

</div>

<h1 class ="headcolor">到云</h1>

</div>

<div class="login-wrap">

<div class="login-form">

<div class="ms-title">到云后台管理系统</div>

<!-- <div class="login-select">

<el-tabs v-model="activeName" @tab-click="handleClick">

<el-tab-pane label="密码登录" name="1"></el-tab-pane>

<el-tab-pane id="yzma" label="验证码登录" name="2"></el-tab-pane>

</el-tabs>

</div> -->

<div>

<el-tabs v-model="activeName" @tab-click="handleClick">

<el-tab-pane label="密码登录" name="1"></el-tab-pane>

<el-tab-pane label="手机号登录" name="2"></el-tab-pane>

</el-tabs>

</div>

<el-form

v-show="activeName == '1'"

autocomplete="on"

:model="loginForm"

:rules="loginRules"

ref="loginForm"

label-position="left"

>

<el-form-item prop="username">

<el-input

prefix-icon="el-icon-user-solid"

name="username"

type="text"

v-model="loginForm.username"

autocomplete="on"

placeholder="请输入邮箱/昵称/手机号"

></el-input>

</el-form-item>

<el-form-item prop="password">

<el-input

prefix-icon="el-icon-lock"

name="password"

:type="passwordType"

v-model="loginForm.password"

autocomplete="on"

placeholder="请输入密码"

>

<i slot="suffix" @click="showPwd">

<!-- <svg-icon icon-class="eye" /> -->

</i>

</el-input>

</el-form-item>

<el-button

type="primary"

style="width: 100%; margin-bottom: 10px"

:loading="loading"

@click="handleLogin"

>登录</el-button

>

<el-link type="primary" style="float: left" @click="signup()"

>注册</el-link

>

<el-link type="primary" style="float: right" @click="forgetPass()"

>忘记密码</el-link

>

</el-form>

<el-form

v-show="activeName == '2'"

autocomplete="on"

:model="loginForm1"

:rules="loginRules1"

ref="loginForm1"

label-position="left"

>

<el-form-item prop="username">

<el-input

prefix-icon="el-icon-user-solid"

name="username"

type="text"

v-model="loginForm1.username"

autocomplete="on"

placeholder="请输入手机号"

></el-input>

</el-form-item>

<el-form-item prop="message">

<el-row>

<el-col :span="15">

<el-input

prefix-icon="el-icon-message"

name="password"

v-model="loginForm1.message"

autocomplete="on"

placeholder="请输入验证码"

/>

</el-col>

<el-col :span="8" :offset="1">

<el-button

type="primary"

plain

:disabled="isDisabled"

@click="getMessage()"

id="dyMobileButton"

>{{ butName }}</el-button

>

</el-col>

</el-row>

<el-row id="yzm">

<el-col>

<label v-show="isdisplay == true">{{ yzmnum }}</label>

</el-col>

</el-row>

</el-form-item>

<el-button

type="primary"

style="width: 100%; margin-bottom: 10px"

:loading="loading"

@click="handleLogin"

>登录</el-button

>

<el-link type="primary" style="float: right" @click="signup"

>注册</el-link

>

</el-form>

</div>

</div>

<div>

<h1 class ="tailcolor">为了获得最佳浏览效果，建议在分辨率大于或等于 1024\*768 的屏幕下浏览网站<br>

2020 - 2021 福州大学数学与计算机学院工程训练2\_209<br>

地址：福建省福州市闽侯县上街镇福州大学数计学院2号楼 联系电话：13774763228

</h1>

</div>

</div>

</template>

<script>

import qs from "qs";

import {get} from '../../../utils/axios'

export default {

inject: ["reload"], // 使用 inject 注入 reload 变量

data() {

return {

loginForm: {

username: "",

password: "",

},

loginForm1: {

username: "",

message: "",

},

loginRules: {

username: [{ required: true, trigger: "blur", message: "请输入邮箱" }],

password: [{ required: true, trigger: "blur", message: "请输入密码" }],

},

loginRules1: {

username: [

{ required: true, trigger: "blur", message: "请输入手机号" },

{

// type: "email",

message: "请输入正确的手机号码",

trigger: ["blur", "change"],

},

],

message: [{ required: true, trigger: "blur", message: "请输入验证码" }],

},

passwordType: "password",

loading: false,

activeName: "1",

butName: "获取验证码",

isDisabled: false,

isdisplay: false, //显示验证码

yzmnum: 1234,

};

},

created(){

this.confirmAutoLogin();

},

methods: {

handleClick(tab, event) {},

loginByThird(){//第三方登录

this.$http.get("/api/getQQCode").then(

res => {

window.open(res.data.url)

},

res => {

this.$router.push({

path: "/" + res

});

})

},

confirmAutoLogin() {

var isLogin = JSON.parse(localStorage.getItem("isLogin")); //获取缓存看是否登录过

var time = localStorage.getItem("loginTime");

var nowTime = new Date().getTime();

let token = localStorage.getItem("Authorization");

if (

isLogin === true &&

nowTime <= time + 2592000000 &&

token != null &&

token != ""

) {

//登录过来直接进去，30天内登录不需要重新登录

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

}

},

getMessage() {

var time = 60;

this.$refs.loginForm1.validateField("username", errMsg => {

if (errMsg) {

} else {

console.log("this.loginForm1.username = "+this.loginForm1.username)

var data = {

// email: this.loginForm1.username

telephone: this.loginForm1.username

};

this.$http.post("/api/sendMessage", null, {

params: data

}).then(

res => {

console.log("sendcoderes.data = "+JSON.stringify(res.data))

if (res.data.respCode == "请输入真实手机号") {

this.$alert("请输入真实手机号!", "失败", {

confirmButtonText: "确定"

});

} else {

localStorage.setItem("validateCode", res.data.respCode);

}

},

res => {

this.$router.push({

path: "/" + res

});

}

);

// this.$axios.post("/api/sendCode", data, this.config).then(res => {

// // this.validateCode = res.data;

// console.log("http拦截器调用")

// console.log("res.data = "+JSON.stringify(res.data))

// localStorage.setItem("validateCode", res.data);

// });

//倒计时

let timer = setInterval(() => {

if (time == 0) {

clearInterval(timer);

this.isDisabled = false;

this.butName = "获取验证码";

} else {

this.butName = time + "秒后重试";

this.isDisabled = true;

time--;

}

}, 1000);

}

});

},

showPwd() {

if (this.passwordType === "password") {

this.passwordType = "";

} else {

this.passwordType = "password";

}

},

handleLogin() {

// console.log(this.config);

var data = {

email: this.loginForm1.username

};

//////////////////////////////////////////////////////////////////////////

// this.$axios.post("/api/sendCode", data, this.config).then(res => {

// // this.validateCode = res.data;

// console.log("登录http拦截器调用")

// localStorage.setItem("validateCode", res.data);

// });

console.log("准备调用")

// get('/api/sendCode',data).then(res =>{console.log("res ="+JSON.stringify(res))})

console.log("准备结束")

/////////////////////////////////////////////////////////////////////////

if (this.activeName == "1") {

this.$refs.loginForm.validate(valid => {

if (valid) {

this.loading = true;

//密码登录

var data = {

account: this.loginForm.username,

password: this.loginForm.password

};

localStorage.setItem("roleEmail", data.account);

// this.$http.post("/api/loginByPassword", data).then(

this.$http.post("/api/loginByPassword", null, {

params: data

}).then(

res => {

if (res.data.result == true) {

this.loading = false;

//登录成功

if (res.data.dataList[0].role != "3") {

console.log("res.data = "+JSON.stringify(res.data));

localStorage.setItem("roleId", res.data.role);

console.log("res.data.role = = "+res.data.role)

console.log("res.data.dataList = "+JSON.stringify(res.data.dataList[0]))

if (res.data.dataList[0].role == "0") {

//登录角色

localStorage.setItem("roles", "teacher");

localStorage.setItem("roleId", 0);

} else if (res.data.dataList[0].role == "1") {

localStorage.setItem("roles", "admin");

localStorage.setItem("roleId", 1);

} else if (res.data.dataList[0].role == "2") {

localStorage.setItem("roles", "superAdmin");

localStorage.setItem("roleId", 2);

}

var date = new Date();

localStorage.setItem("loginTime", date.getTime()); //登录时间

localStorage.setItem("Authorization", res.data.token);

localStorage.setItem("account", res.data.account);

localStorage.setItem("isLogin", true);

// console.log("this.localstorage = "+JSON.stringify(localStorage))

this.$http.get("/api/menus").then(res => {

// console.log("菜单列表data = "+JSON.stringify(res.data))

localStorage.setItem(

"menuList",

JSON.stringify(res.data)

);

// location.reload();

});

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

} else {

//学生

this.$alert(

"该账号为学生账号，没有权限登录后台管理系统，如有疑问请联系管理员！",

"提示",

{

confirmButtonText: "确定"

}

);

}

} else {

this.loading = false;

this.$alert(res.data.respCode, "登录失败", {

confirmButtonText: "确定"

});

}

},

res => {

this.$router.push({

path: "/" + res

});

}

);

}

});

} else {

//验证码登录

this.$refs.loginForm1.validate(valid => {

if (valid) {

if (

localStorage.getItem("validateCode") != this.loginForm1.message

) {

this.$alert("验证码错误，请重新输入", "注册失败", {

confirmButtonText: "确定"

});

} else {

this.loading = true;

var data = {

telephone: this.loginForm1.username

};

this.$http.post("/api/loginByMessage", null, {

params: data

}).then(

// this.$http.post("/api/loginByCode", data).then(

res => {

console.log("res.data = "+JSON.stringify(res.data));

this.loading = false;

if (res.data.result == true) {

//登录成功914392

if (res.data.dataList[0].role != "3") {

localStorage.setItem("roleId", res.data.role);

if (res.data.dataList[0].role == "0") {

//登录角色

localStorage.setItem("roles", "teacher");

} else if (res.data.dataList[0].role == "1") {

localStorage.setItem("roles", "admin");

} else if (res.data.dataList[0].role == "2") {

localStorage.setItem("roles", "superAdmin");

}

var date = new Date();

localStorage.setItem("loginTime", date.getTime()); //登录时间

this.loading = false;

localStorage.setItem("Authorization", res.data.token);

localStorage.setItem("account", res.data.email);

localStorage.setItem("isLogin", true);

this.$http.get("/api/menus").then(res => {

localStorage.setItem(

"menuList",

JSON.stringify(res.data)

);

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

});

} else {

//学生

this.$alert(

"该账号为学生账号，没有权限登录后台管理系统，如有疑问请联系管理员！",

"提示",

{

confirmButtonText: "确定"

}

);

}

} else {

this.loading = false;

this.$alert(res.data.respCode, "登录失败", {

confirmButtonText: "确定"

});

}

},

res => {

this.$router.push({

path: "/" + res

});

}

);

}

}

});

}

},

signup() {

this.$router.push("/signup");

},

forgetPass() {

this.$router.push("/forgetPassword");

}

}

};

</script>

注册页面代码

<template>

<div class="login-container">

<!-- <div class="login-form"> -->

<!-- <el-row> -->

<el-form

class="login-form"

autocomplete="on"

:model="loginForm"

:rules="loginRules"

ref="loginForm"

label-position="left"

>

<div class="title-container">

<img src="../../assets/cloud.png" style="width:250px" />

<!-- <h3 class="title">注册</h3> -->

</div>

<el-form-item prop="username">

<el-input

prefix-icon="el-icon-user-solid"

name="username"

type="text"

v-model="loginForm.username"

autocomplete="on"

placeholder="请输入邮箱"

></el-input>

</el-form-item>

<el-form-item prop="message">

<el-row>

<el-col :span="15">

<el-input

prefix-icon="el-icon-message"

name="password"

v-model="loginForm.message"

autocomplete="on"

placeholder="请输入验证码"

/>

</el-col>

<el-col :span="8" :offset="1">

<el-button

type="primary"

plain

:disabled="isDisabled"

@click="getMessage()"

id="dyMobileButton"

>{{butName}}</el-button>

</el-col>

</el-row>

</el-form-item>

<el-form-item prop="pass">

<el-input

prefix-icon="el-icon-lock"

type="password"

v-model="loginForm.pass"

autocomplete="off"

placeholder="请输入密码"

></el-input>

</el-form-item>

<el-form-item prop="checkPass">

<el-input

prefix-icon="el-icon-finished"

type="password"

v-model="loginForm.checkPass"

autocomplete="off"

placeholder="请确认密码"

></el-input>

</el-form-item>

<el-button

type="primary"

style="width:100%;margin-bottom:10px;"

:loading="loading"

@click="signup"

>注册</el-button>

<el-link type="primary" style="float:right" @click="login">已有账号，去登录</el-link>

</el-form>

<!-- </el-row> -->

<!-- </div> -->

</div>

</template>

<script>

export default {

data() {

var validatePass = (rule, value, callback) => {

var pattern=/^(?!^[0-9]+$)(?!^[A-z]+$)(?!^[^A-z0-9]+$)^[^\s\u4e00-\u9fa5]{6,16}$/

if (value === "") {

callback(new Error("请输入密码"));

} else {

// if (this.loginForm.checkPass !== "") {

// this.$refs.loginForm.validateField("checkPass");

// }

if(!pattern.exec(value)){

callback(new Error("6-16位，由数字、英文、符号三种类型构成，至少包含两种类型字符"));

}

callback();

}

};

var validatePass2 = (rule, value, callback) => {

if (value === "") {

callback(new Error("请再次输入密码"));

} else if (value !== this.loginForm.pass) {

callback(new Error("两次输入密码不一致!"));

} else {

callback();

}

};

return {

loginForm: {

username: "",

message: "",

pass: "",

checkPass: ""

},

loginRules: {

username: [

{ required: true, message: "请输入邮箱", trigger: "blur" },

{

type: "email",

message: "请输入正确的邮箱",

trigger: ["blur", "change"]

}

],

message: [{ required: true, trigger: "blur", message: "请输入验证码" }],

pass: [{ required: true, validator: validatePass, trigger: "blur" }],

checkPass: [

{ required: true, validator: validatePass2, trigger: "blur" }

]

},

loading: false,

activeName: "1",

butName: "获取验证码",

isDisabled: false,

validateCode: ""

};

},

methods: {

getMessage() {

var time = 60;

this.$refs.loginForm.validateField("username", errMsg => {

if (errMsg) {

} else {

var data = {

email: this.loginForm.username

};

this.$http.post("/api/sendCode", data).then(

res => {

if (res.data.respCode == "请输入真实邮箱") {

this.$alert("请输入真实邮箱!", "失败", {

confirmButtonText: "确定"

});

} else {

localStorage.setItem("validateCode", res.data.respCode);

}

},

res => {

this.$router.push({

path: "/" + res

});

}

);

//倒计时

let timer = setInterval(() => {

if (time == 0) {

clearInterval(timer);

this.isDisabled = false;

this.butName = "获取验证码";

} else {

this.butName = time + "秒后重试";

this.isDisabled = true;

time--;

}

}, 1000);

}

});

},

signup() {

var flag = false;

var flag1 = false;

this.$refs.loginForm.validate(valid => {

if (valid) {

if (localStorage.getItem("validateCode") != this.loginForm.message) {

this.$alert("验证码错误，请重新输入", "注册失败", {

confirmButtonText: "确定"

});

} else {

this.loading = true;

var data = {

email: this.loginForm.username,

password: this.loginForm.pass

};

this.$http.post("/api/register", data).then(

res => {

this.loading = false;

if (res.data.respCode == "1") {

//注册成功 角色默认为教师

localStorage.setItem("roleId", "0");

// if (res.data.role == "1") {

localStorage.setItem("roles", "teacher");

// }

var date = new Date();

localStorage.setItem("loginTime", date.getTime()); //注册时间

localStorage.setItem("isLogin", true);

localStorage.setItem("Authorization", res.data.token);

localStorage.setItem("account", this.loginForm.username);

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

} else {

this.loading = false;

this.$alert(res.data.respCode, "注册失败", {

confirmButtonText: "确定"

});

}

},

res => {

this.$router.push({

path: "/" + res

});

}

);

}

} else {

return false;

}

});

},

login() {

this.$router.push("/login");

}

}

};

</script>

首页界面代码

<template>

<div class="container">

<div class="main">

<h1 class="title">Welcome to 到云!</h1>

<el-divider></el-divider>

<div style="margin-left:20px">

<h3>产品简介</h3>

<div

class="des"

>&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;到云是一款免费课堂互动教学 App。它基于移动互联环境，实现老师与学生之间的即时互动、签到，完善的激励与评价体系激发学生在移动设备上的自主学习兴趣，完整的学习行为记录实现对学生学习的过程性考核，更能为老师提供高质量的教学研究大数据，并实现个性化教学和助教功能。</div>

<h3>技术顾问联系方式</h3>

<el-table :data="tableData" style="width: 100%;" border>

<el-table-column prop="name" label="成员" min-width="180" align="center"></el-table-column>

<el-table-column prop="sno" label="学号" min-width="180" align="center"></el-table-column>

<el-table-column prop="telephone" label="联系方式" min-width="180" align="center"></el-table-column>

</el-table>

</div>

</div>

</div>

</template>

<script>

export default {

inject: ["reload"], // 使用 inject 注入 reload 变量

data() {

return {

tableData: [

{

name: "林高升",

sno: "200327170",

telephone: "17859918979"

},

{

name: "张先涌",

sno: "200327155",

telephone: "18760372609"

},

{

name: "欧阳育斌",

sno: "200327144",

telephone: "18987588901"

},

{

name: "杨铭",

sno: "200327105",

telephone: "13385989351"

}

]

};

},

beforeRouteEnter(to, from, next) {

// 在当前路由改变，路由参数发生变化，组件被复用时调用

// 里面写获取刷新数据的方法

next();

},

beforeRouteUpdate(to, from, next) {

next();

},

created() {

this.showAuthorityList();

// this.getMenuList();

// console.log("menulist = "+localStorage.getItem("menuList"))

// if(localStorage.getItem("isUpdate")==null||localStorage.getItem("isUpdate")==undefined){

// localStorage.setItem("isUpdate",1);

// this.reload();

// // location.reload();

// }

},

methods: {

showAuthorityList() {

var roleId = parseInt(localStorage.getItem("roleId"));

// localStorage.setItem("roleId", 1);

console.log("roleId = "+JSON.stringify(roleId))

this.$http.get("/api/rolePower?role\_id=" + roleId).then(res => {

const value = JSON.stringify(res.data);

localStorage.setItem("authority", value);

});

},

// getMenuList() {

// this.$http.get("/api/menus").then(res => {

// console.log(res.data)

// localStorage.setItem("menuList", JSON.stringify(res.data));

// });

// }

}

};

</script>

个人信息界面代码

<!--指示板-->

<template>

<div class="dashboard-editor-container">

<el-row class="form-style">

<el-col :span="11" class="form1">

<el-button type="primary" style="width:100%;margin-bottom:20px;">个人信息</el-button>

<el-form

label-width="80px"

:model="ruleForm"

:rules="rules"

ref="ruleForm"

style="width:90%"

>

<!-- <el-form-item> -->

<!-- <el-upload

class="avatar-uploader"

action="https://jsonplaceholder.typicode.com/posts/"

:show-file-list="false"

:on-success="handleAvatarSuccess"

:before-upload="beforeAvatarUpload"

>

<img v-if="imageUrl" :src="imageUrl" class="avatar" />

<i v-else class="el-icon-plus avatar-uploader-icon"></i>

</el-upload>-->

<!-- </el-form-item> -->

<el-form-item label="账号/邮箱">

<el-input v-model="ruleForm.email" disabled></el-input>

</el-form-item>

<el-form-item label="名称" prop="name">

<el-input v-model="ruleForm.name" placeholder="请输入名称"></el-input>

</el-form-item>

<el-form-item label="工号" prop="sno">

<el-input v-model="ruleForm.sno" placeholder="请输入工号"></el-input>

</el-form-item>

<el-form-item label="昵称" prop="nickname">

<el-input v-model="ruleForm.nickname" placeholder="请输入昵称"></el-input>

</el-form-item>

<el-form-item label="性别">

<el-radio-group v-model="ruleForm.sex">

<el-radio label="0">男</el-radio>

<el-radio label="1">女</el-radio>

</el-radio-group>

</el-form-item>

<el-form-item label="学院" prop="school">

<el-cascader v-model="ruleForm.school" :options="options" @change="handleChange"></el-cascader>

</el-form-item>

<el-form-item label="出生日期">

<el-date-picker

value-format="yyyy-MM-dd"

v-model="ruleForm.birth"

type="date"

placeholder="选择日期"

></el-date-picker>

</el-form-item>

<el-form-item label="用户角色">

<el-tag type="success" v-if="ruleForm.role=='0'">老师</el-tag>

<el-tag type="success" v-if="ruleForm.role=='1'">管理员</el-tag>

<el-tag type="success" v-if="ruleForm.role=='2'">超级管理员</el-tag>

</el-form-item>

<el-form-item label="经验值">

<el-input v-model="ruleForm.exp" disabled></el-input>

</el-form-item>

<el-form-item label="电话号码" prop="phone">

<el-input v-model="ruleForm.phone"></el-input>

</el-form-item>

<el-form-item>

<el-button type="primary" @click="submitForm('ruleForm')" style="width:40%">提交</el-button>

<el-button @click="resetForm('ruleForm')" style="width:40%">重置</el-button>

</el-form-item>

</el-form>

</el-col>

<el-col :span="11" class="form2" :offset="1">

<el-button type="primary" style="width:100%;margin-bottom:20px;">密码修改</el-button>

<el-form

:model="ruleForm2"

status-icon

:rules="rules2"

ref="ruleForm2"

label-width="100px"

class="demo-ruleForm"

style="width:90%"

>

<el-form-item label="原密码" required>

<el-input v-model="ruleForm2.oldPass"></el-input>

</el-form-item>

<el-form-item label="新密码" prop="pass" required>

<el-input v-model="ruleForm2.pass"></el-input>

</el-form-item>

<el-form-item label="确认密码" prop="checkPass" required>

<el-input v-model="ruleForm2.checkPass"></el-input>

</el-form-item>

<el-form-item>

<el-button type="primary" @click="submitForm2('ruleForm2')" style="width:40%">提交</el-button>

<el-button @click="resetForm2('ruleForm2')" style="width:40%">重置</el-button>

</el-form-item>

</el-form>

</el-col>

</el-row>

</div>

</template>

<script>

export default {

data() {

var validateSchool = (rule, value, callback) => {

if (value === "") {

callback(new Error("请选择学校及学院"));

} else {

callback();

}

};

var validatePass = (rule, value, callback) => {

var pattern=/^(?!^[0-9]+$)(?!^[A-z]+$)(?!^[^A-z0-9]+$)^[^\s\u4e00-\u9fa5]{6,16}$/

if (value === "") {

callback(new Error("请输入密码"));

} else {

// if (this.loginForm.checkPass !== "") {

// this.$refs.loginForm.validateField("checkPass");

// }

if(!pattern.exec(value)){

callback(new Error("6-16位，由数字、英文、符号三种类型构成，至少包含两种类型字符"));

}

callback();

}

};

var validatePass2 = (rule, value, callback) => {

if (value === "") {

callback(new Error("请再次输入密码"));

} else if (value !== this.ruleForm2.pass) {

callback(new Error("两次输入密码不一致!"));

} else {

callback();

}

};

var checkPhone = (rule, value, callback) => {

if (!value) {

// return callback(new Error("手机号不能为空"));

callback();

} else {

const reg = /^1[3|4|5|7|8][0-9]\d{8}$/;

console.log(reg.test(value));

if (reg.test(value)) {

callback();

} else {

return callback(new Error("请输入正确的手机号"));

}

}

};

return {

ruleForm: {

email: "",

nickname: "",

exp: "",

name: "",

birth: "",

phone: "",

sno: "",

sex: "1",

type: 1,

school: []

},

ruleForm2: {

oldPass: "",

pass: "",

checkPass: ""

},

rules: {

name: [{ required: true, message: "请输入用户名", trigger: "blur" }],

nickname: [{ required: true, message: "请输入昵称", trigger: "blur" }],

school: [

{ required: false, validator: validateSchool, trigger: "change" }

],

phone: [{ validator: checkPhone, trigger: "blur" }]

},

rules2: {

pass: [{ validator: validatePass, trigger: "blur" }],

checkPass: [{ validator: validatePass2, trigger: "blur" }]

},

options: [],

imageUrl: ""

};

},

created() {

this.showSchool();

this.showUserInfo();

},

methods: {

showUserInfo() {

this.$http

.get("/api/user/info?" + "email=" + localStorage.getItem("account"))

.then(

res => {

this.ruleForm.email = localStorage.getItem("account");

this.ruleForm.name = res.data.name;

this.ruleForm.sno = res.data.sno;

this.ruleForm.nickname = res.data.nickname;

// console.log("res.data.sex = "+JSON.stringify(res.data.sex))

// this.ruleForm.sex = res.data.sex.toString();

this.ruleForm.sex = '男';

this.ruleForm.role = res.data.role;

if (res.data.school != "" && res.data.school != "0") {

// this.ruleForm.school = res.data.school.split("/"); //字符分割

this.ruleForm.school = '福州大学'

}

console.log( this.ruleForm.school);

this.ruleForm.birth = res.data.birth;

this.ruleForm.phone = res.data.telphone;

this.ruleForm.exp = res.data.exp;

},

res => {

this.$router.push({

path: "/" + res

});

}

);

},

showSchool() {

this.$http.get("/api/schools?info=1").then(

res => {

console.log("res.data = "+JSON.stringify(res.data))

let temarr= [];

for(let i in res.data){

temarr.push(res.data[i]);

}

this.options = temarr;

},

res => {

this.$router.push({

path: "/" + res

});

}

);

},

submitForm(formName) {

this.$refs[formName].validate(valid => {

if (valid) {

var data = {

email: this.ruleForm.email,

nickname: this.ruleForm.nickname,

sex: this.ruleForm.sex,

school: this.ruleForm.school,

telphone: this.ruleForm.phone,

birth: this.ruleForm.birth,

name: this.ruleForm.name,

sno: this.ruleForm.sno,

image: "0"

};

this.$http.put("/api/user/info", data).then(

res => {

if (res.data.respCode == "1") {

this.$alert("用户信息修改成功", "成功", {

confirmButtonText: "确定"

});

}else{

this.$alert(res.data.respCode, "失败", {

confirmButtonText: "确定"

});

}

},

res => {

this.$router.push({

path: "/" + res

});

}

);

return false;

}

});

},

resetForm(formName) {

this.$refs[formName].resetFields();

this.ruleForm.name1 = "";

this.ruleForm.sex = "1";

},

submitForm2(formName) {

this.$refs[formName].validate(valid => {

if (valid) {

var data = {

telephone: localStorage.getItem("roleEmail"),

newpassword1: this.ruleForm2.pass,

newpassword2: this.ruleForm2.checkPass,

oldpassword: this.ruleForm2.oldPass

};

console.log("this.localStorage = "+JSON.stringify(localStorage))

console.log("修改密码data = "+JSON.stringify(data))

this.$http.post("/api/user/updatePassword", data).then(

res => {

if (res.data.respCode == "1") {

this.$alert("密码修改成功，跳转到登录页重新登录", "成功", {

confirmButtonText: "确定"

});

localStorage.removeItem("Authorization");

localStorage.removeItem("isLogin");

this.$router.push("/login");

} else {

this.$alert(res.data.respCode, "失败", {

confirmButtonText: "确定"

});

}

},

res => {

this.$router.push({

path: "/" + res

});

}

);

} else {

this.$alert("有必填项未填写或者填写错误", "警告", {

confirmButtonText: "确定"

});

return false;

}

});

},

resetForm2(formName) {

this.$refs[formName].resetFields();

this.ruleForm2.oldPass = "";

},

handleAvatarSuccess(res, file) {

this.imageUrl = URL.createObjectURL(file.raw);

},

beforeAvatarUpload(file) {

const isJPG = file.type === "image/jpeg";

const isLt2M = file.size / 1024 / 1024 < 2;

if (!isJPG) {

this.$message.error("上传头像图片只能是 JPG 格式!");

}

if (!isLt2M) {

this.$message.error("上传头像图片大小不能超过 2MB!");

}

return isJPG && isLt2M;

},

handleChange(value) {

this.ruleForm.school = "";

for (var i = 0; i < value.length; i++) {

this.ruleForm.school += value[i];

if (i != value.length - 1) {

this.ruleForm.school += "/";

}

}

}

}

};

</script>

# 二、功能代码

角色管理功能代码

<template>

<div class="app-container">

<div class="form-style">

<el-form :inline="true" :model="formInline" class="demo-form-inline">

<el-form-item>

<el-button type="primary" size="small" @click="addData()" icon="el-icon-plus">新增</el-button>

</el-form-item>

<el-form-item style="float:right">

<el-button type="primary" size="small" @click="resetData()">重置</el-button>

</el-form-item>

<el-form-item style="float:right">

<el-button type="primary" size="small" @click="searchData()">查询</el-button>

</el-form-item>

<el-form-item label="状态：" style="float:right">

<el-select v-model="formInline.state" placeholder="请选择状态" size="small">

<el-option label="禁用" value="0"></el-option>

<el-option label="启用" value="1"></el-option>

</el-select>

</el-form-item>

<el-form-item label="角色名称：" style="float:right">

<el-input placeholder="请输入角色名称" size="small" v-model="formInline.role"></el-input>

</el-form-item>

</el-form>

<el-table

:data="list"

v-loading="listLoading"

element-loading-text="Loading"

fit

highlight-current-row

tooltip-effect="dark"

style="width: 100%"

:header-cell-style="{background:'#eef1f6',color:'#606266'}"

>

<!-- <el-table-column type="selection" width="50"></el-table-column> -->

<el-table-column label="序号" type="index" width="50" align="center"></el-table-column>

<el-table-column label="角色名称" min-width="110" align="center">

<template slot-scope="scope">

<span>{{scope.row.name}}</span>

</template>

</el-table-column>

<el-table-column label="备注" min-width="110" align="center">

<template slot-scope="scope">

<span>{{scope.row.description}}</span>

</template>

</el-table-column>

<el-table-column label="角色状态" min-width="50" align="center">

<template slot-scope="scope">

<el-button

:type="btnState(scope.row.state)"

plain

@click="changeState(scope.row)"

size="small"

>{{filterState(scope.row.state)}}</el-button>

</template>

</el-table-column>

<el-table-column align="center" label="操作" min-width="100">

<template slot-scope="scope">

<div>

<el-link type="primary" @click="editData(scope.row)">编辑</el-link>

<el-divider direction="vertical"></el-divider>

<el-link type="primary" @click="permissionAssign(scope.row)">分配权限</el-link>

<el-divider direction="vertical"></el-divider>

<el-link type="danger" @click="deleteData(scope.row)">删除</el-link>

</div>

</template>

</el-table-column>

</el-table>

<el-pagination

background

@current-change="handleCurrentChange"

layout="total, prev, pager, next"

:total="totalNum"

v-if="totalNum!=0"

:page-size="pageSize"

></el-pagination>

</div>

<el-dialog :title="title" :visible.sync="dialogFormVisible">

<el-form

:model="roleForm"

:rules="rules"

ref="roleForm"

label-width="100px"

class="demo-roleForm"

>

<el-form-item label="角色名称" prop="name">

<el-input v-model="roleForm.name"></el-input>

</el-form-item>

<el-form-item label="角色状态" v-if="title=='编辑角色'">

<el-radio v-model="roleForm.state" label="0">禁用</el-radio>

<el-radio v-model="roleForm.state" label="1">启用</el-radio>

</el-form-item>

<el-form-item label="备注">

<el-input v-model="roleForm.description" type="textarea" :rows="4"></el-input>

</el-form-item>

</el-form>

<div slot="footer" class="dialog-footer" style="text-align: center;margin-bottom:10px">

<el-button type="primary" @click="submitForm('roleForm')" style="width:180px">确定</el-button>

<el-button @click="resetForm('roleForm')" style="width:180px">取消</el-button>

</div>

<div

slot="footer"

class="dialog-footer"

style="text-align: center;color: #5d5b5b;font-size:14px"

v-if="title=='新增角色'"

>

<span>新增角色默认为已启用</span>

</div>

</el-dialog>

</div>

</template>

<script>

export default {

data() {

return {

list: [],

AllData: [],

listLoading: false,

dialogFormVisible: false,

roleForm: {

name: "",

state: "1",

description: ""

},

rules: {

name: [{ required: true, message: "请输入角色名称", trigger: "blur" }]

},

totalNum: 1, //0

title: "新增角色",

pageSize: 10,

formInline: {

role: "",

state: ""

},

page: 1

};

},

created() {

this.showRoleInfo();

},

methods: {

showData() {

this.list = [];

this.listLoading = true;

//获取角色信息

},

addData() {

//添加新角色

this.reset();

this.dialogFormVisible = true;

this.title = "新增角色";

},

searchData() {

//搜索

this.list = [];

this.listLoading = true;

this.page = 1;

var data = {

page: this.page,

name: this.formInline.role,

state: this.formInline.state

};

this.$http

.get(

"/api/role?page=" +

data.page +

"&name=" +

data.name +

"&state=" +

data.state

)

.then(res => {

this.listLoading = false;

this.totalNum = res.data.total;

if (this.totalNum != 0) {

delete res.data[0];

this.list = res.data.records;

}

});

},

resetData() {

this.formInline.role = "";

this.formInline.state = "";

this.showRoleInfo(this.page);

},

reset() {

this.roleForm.name = "";

this.roleForm.note = "";

},

btnState(state) {

if (state == "1") {

return "primary";

} else {

return "danger";

}

},

filterState(state) {

if (state == "1") {

return "启用";

} else {

return "禁用";

}

},

changeState(row) {

this.$http

.patch("/api/role?id=" + row.id + "&state=" + row.state)

.then(res => {

if (res.data.respCode == "1") {

this.$alert("状态修改成功", "成功", {

confirmButtonText: "确定"

});

if (row.state == "1") {

row.state = "0";

row.stateName = "禁用";

row.type = "0";

} else {

row.state = "1";

row.stateName = "启用";

row.type = "1";

}

} else {

this.$alert(res.data.respCode, "失败", {

confirmButtonText: "确定"

});

}

});

},

editData(row) {

this.roleForm = row;

this.roleForm.state = this.roleForm.state.toString();

this.dialogFormVisible = true;

this.title = "编辑角色";

},

deleteData(row) {

//单个删除

this.$confirm("确定要删除该角色？", "删除角色", {

confirmButtonText: "确定",

cancelButtonText: "取消",

type: "info"

})

.then(() => {

this.$http.delete("/api/role?id=" + row.id).then(res => {

if (res.data.respCode == "1") {

this.$alert("删除成功", "成功", {

confirmButtonText: "确定"

});

}

this.showRoleInfo(this.page);

});

})

.catch(() => {});

},

submitForm(formName) {

var data = {

id: this.roleForm.id,

description: this.roleForm.description,

name: this.roleForm.name,

state: this.roleForm.state

};

var addData = {

description: this.roleForm.description,

name: this.roleForm.name

};

this.$refs[formName].validate(valid => {

if (valid) {

this.dialogFormVisible = false;

if (this.title == "新增角色") {

this.$http.post("/api/role", addData).then(res => {

if (res.data.respCode == "1") {

this.$alert("角色新增成功", "成功", {

confirmButtonText: "确定"

});

this.showRoleInfo(this.page);

} else {

this.$alert("角色新增失败", "失败", {

confirmButtonText: "确定"

});

this.showRoleInfo(this.page);

}

this.$refs[formName].resetFields();

});

} else {

this.$http.put("/api/role", data).then(res => {

if (res.data.respCode == "1") {

this.$alert("角色修改成功", "成功", {

confirmButtonText: "确定"

});

this.showRoleInfo(this.page);

} else {

this.$alert("角色修改失败", "失败", {

confirmButtonText: "确定"

});

this.showRoleInfo(this.page);

}

this.$refs[formName].resetFields();

});

}

}

});

},

resetForm(formName) {

this.dialogFormVisible = false;

this.$refs[formName].resetFields();

this.reset();

this.showRoleInfo(this.page);

},

showRoleInfo(page) {

this.list = [];

this.listLoading = true;

this.page = 1;

var name = "";

var state = "";

//获取角色信息

var data = {

page: this.page,

name: name,

state: state

};

this.$http

.get(

"/api/role?page=" +

data.page +

"&name=" +

data.name +

"&state=" +

data.state

)

.then(res => {

this.listLoading = false;

this.totalNum = res.data.total;

if (this.totalNum != 0) {

delete res.data[0];

this.list = res.data.records;

}

});

},

handleCurrentChange(val) {

// console.log(val);

//根据页数显示页面上的数据。

this.list = [];

var mapsize = Object.keys(this.AllData).length;

if (mapsize - (val - 1) \* this.pageSize < this.pageSize) {

for (var i = (val - 1) \* this.pageSize + 1; i <= mapsize; i++) {

this.list.push(this.AllData[i]);

}

} else {

for (

var i = (val - 1) \* this.pageSize + 1;

i <= val \* this.pageSize;

i++

) {

this.list.push(this.AllData[i]);

}

}

},

permissionAssign(row) {

var description = [];

this.$http.get("/api/rolePower?role\_id=" + row.id).then(res => {

description = res.data;

this.$router.push({

path: "/roleManage/permissionAssignment",

query: { username: row.name, id: row.id, description }

});

});

}

}

};

</script>

角色权限分配功能代码

<template>

<div class="app-container">

<div class="form-style">

<div></div>

<el-form :inline="true" class="demo-form-inline">

<el-form-item>

<span>当前正在为</span>

</el-form-item>

<el-form-item>

<el-tag>{{ this.$route.query.username}}</el-tag>

</el-form-item>

<el-form-item>

<span>分配权限</span>

</el-form-item>

</el-form>

<el-divider></el-divider>

<h5>菜单显示</h5>

<el-checkbox-group v-model="showList">

<el-checkbox label="1">首页</el-checkbox>

<el-checkbox label="2">角色管理</el-checkbox>

<el-checkbox label="3">用户管理</el-checkbox>

<el-checkbox label="4">数据字典管理</el-checkbox>

<el-checkbox label="5">菜单管理</el-checkbox>

<el-checkbox label="6">系统参数管理</el-checkbox>

<el-checkbox label="7">学校管理</el-checkbox>

<el-checkbox label="8">班课管理</el-checkbox>

</el-checkbox-group>

<div class = "regular">

<span>规则：点击对应菜单，为该角色分配相对应的菜单</span>

</div>

<el-divider></el-divider>

<h5>用户管理</h5>

<el-checkbox-group v-model="checkList">

<el-checkbox label="1">新增用户</el-checkbox>

<el-checkbox label="2">删除用户</el-checkbox>

<el-checkbox label="3">编辑用户</el-checkbox>

<el-checkbox label="4">分配角色</el-checkbox>

<el-checkbox label="5">恢复角色状态</el-checkbox>

<el-checkbox label="6">重置密码</el-checkbox>

<el-checkbox label="7">查询</el-checkbox>

<el-checkbox label="8">禁用用户</el-checkbox>

</el-checkbox-group>

<div class = "regular">

<span>规则：点击对应功能，为该用户分配相对应的权限</span>

</div>

<el-divider></el-divider>

<h5>菜单管理</h5>

<el-checkbox-group v-model="menuList">

<el-checkbox label="1">新增菜单</el-checkbox>

<el-checkbox label="2">编辑菜单</el-checkbox>

<el-checkbox label="3">删除菜单</el-checkbox>

<el-checkbox label="4">查询菜单</el-checkbox>

</el-checkbox-group>

<div class = "regular">

<span>规则：点击对应菜单选项，为该用户分配相对应的权限</span>

</div>

<el-divider></el-divider>

<!--

<h5>班课管理</h5>

<el-checkbox-group v-model="classList">

<el-checkbox label="1">发起签到</el-checkbox>

<el-checkbox label="2">创建班课</el-checkbox>

<el-checkbox label="3">创建活动</el-checkbox>

<el-checkbox label="4">分配小组</el-checkbox>

</el-checkbox-group>

-->

<div class="foot">

<el-button type="primary" class="but" @click="saveRolePer()">保存</el-button>

<el-button type="info" plain class="but" @click="reset()">重置</el-button>

</div>

</div>

</div>

</template>

<script>

export default {

data() {

return {

showList: [],

checkList: [],

menuList: [],

classList: [],

perList: [],

saveList: ""

};

},

created() {

this.showRolePer();

},

methods: {

reset() {

this.checkList = [];

this.menuList = [];

this.classList = [];

this.showList = [];

},

showRolePer() {

this.perList = [1,2,3,4,10,12];

var curroles = this.$route.query.username;

if(curroles == "superadmin"){

this.showList = ["1","2","3","4","5","6","7","8"];

// this.showList = ["1","4","5","6","7","8"];

this.checkList = ["1","2","3","4","5","6","7","8"];

this.menuList = ["1","2","3","4"];

}else if(curroles == "admin"){

this.showList = ["1","4","5","6","7","8"];

this.checkList = ["1","2","3","6","7","8"];

this.menuList = ["1","2","3","4"];

}

else if(curroles == "teacher"){

this.showList = ["1","5","6","7","8"];

this.checkList = ["2","6","7"];

this.menuList = ["3","4"];

}

// if (this.perList == 0) {

// this.checkList = [];

// this.menuList = [];

// } else {

// for (var i = 0; i < 12; i++) {

// if (this.perList[i] != 0 && this.perList[i] <= 8) {

// this.checkList.push(this.perList[i]);

// }

// if (this.perList[i] > 8) {

// var menuNum = (this.perList[i] - 8).toString();

// this.menuList.push(menuNum);

// }

// }

// }

},

saveRolePer() {

// if (this.checkList.length == 0 && this.menuList.length == 0) {

// this.saveList = 0;

// } else {

// if (this.menuList.length != 0) {

// for (var i = 0; i < this.menuList.length; i++) {

// this.menuList[i] = (parseInt(this.menuList[i]) + 8).toString();

// }

// }

// if (this.checkList.length != 0) {

// this.saveList = this.checkList[0];

// for (var i = 1; i < this.checkList.length; i++) {

// this.saveList += ",";

// this.saveList += this.checkList[i];

// }

// }

// if (this.menuList.length != 0) {

// this.saveList += ",";

// this.saveList += this.menuList[0];

// for (var i = 1; i < this.menuList.length; i++) {

// this.saveList += ",";

// this.saveList += this.menuList[i];

// }

// }

// }

var power\_id = this.saveList;

this.$alert("保存成功", "成功", {

confirmButtonText: "确定"

});

// this.$http

// .put(

// "/api/rolePower?role\_id=" +

// this.$route.query.id +

// "&power\_id=" +

// power\_id

// )

// .then(res => {

// if (res.data.respCode == "1") {

// this.$http

// .get("/api/rolePower?role\_id=" + this.$route.query.id)

// .then(res => {

// localStorage.setItem("authority", JSON.stringify(res.data));

// });

// this.$alert("保存成功", "成功", {

// confirmButtonText: "确定"

// });

// this.$router.push("/roleManage");

// } else {

// this.$alert("保存失败", "失败", {

// confirmButtonText: "确定"

// });

// }

// });

}

}

};

</script>

用户管理功能代码

<template>

<div class="app-container">

<div class="form-style">

<el-form :inline="true" :model="formInline" class="demo-form-inline">

<el-form-item>

<el-button type="primary" size="small" @click="addData()" icon="el-icon-plus">新增</el-button>

</el-form-item>

<el-form-item>

<el-button type="danger" size="small" @click="deleteData()" icon="el-icon-delete">删除</el-button>

</el-form-item>

<el-form-item style="float:right">

<el-button type="primary" size="small" @click="resetData()">重置</el-button>

</el-form-item>

<el-form-item style="float:right">

<el-button type="primary" size="small" @click="searchData()">查询</el-button>

</el-form-item>

<el-form-item label="状态：" style="float:right">

<el-select v-model="formInline.state" placeholder="请选择状态" size="small">

<el-option label="正常" value="0"></el-option>

<el-option label="禁用" value="1"></el-option>

</el-select>

</el-form-item>

<el-form-item label="用户名：" style="float:right">

<el-input placeholder="请输入用户名" size="small" v-model="formInline.username"></el-input>

</el-form-item>

</el-form>

<el-table

:data="list"

v-loading="listLoading"

element-loading-text="Loading"

fit

highlight-current-row

ref="multipleTable"

tooltip-effect="dark"

style="width: 100%"

@selection-change="handleSelectionChange"

:header-cell-style="{background:'#eef1f6',color:'#606266'}"

>

<el-table-column type="selection" width="50"></el-table-column>

<el-table-column label="序号" type="index" width="50" align="center">

<template slot-scope="scope">

<span>{{(page - 1) \* pageSize + scope.$index + 1}}</span>

</template>

</el-table-column>

<el-table-column label="用户名" min-width="80" align="center">

<template slot-scope="scope">

<span>{{scope.row.name}}</span>

</template>

</el-table-column>

<el-table-column label="性别" min-width="50" align="center">

<template slot-scope="scope">

<span v-if="scope.row.sex=='1'">男</span>

<span v-if="scope.row.sex=='0'">女</span>

</template>

</el-table-column>

<!-- <el-table-column label="电话号码" min-width="100" align="center">

<template slot-scope="scope">

<span>{{scope.row.tel}}</span>

</template>

</el-table-column>-->

<el-table-column label="手机号码" min-width="100" align="center">

<template slot-scope="scope">

<span>{{scope.row.telephone}}</span>

</template>

</el-table-column>

<el-table-column label="角色" min-width="100" align="center">

<template slot-scope="scope">

<span v-if="scope.row.roleId=='0'">教师</span>

<span v-if="scope.row.roleId=='1'">管理员</span>

<span v-if="scope.row.roleId=='2'">超级管理员</span>

<!-- <span>{{scope.row.role}}</span> -->

</template>

</el-table-column>

<el-table-column label="所属机构" min-width="80" align="center">

<template slot-scope="scope">

<span>{{scope.row.schoolName}}</span>

</template>

</el-table-column>

<el-table-column label="状态" min-width="50" align="center">

<template slot-scope="scope">

<el-button

:type="btnState(scope.row.state)"

plain

@click="changeState(scope.row)"

size="small"

>{{filterState(scope.row.state)}}</el-button>

</template>

</el-table-column>

<!-- <el-table-column label="所属机构" min-width="100" align="center">

<template slot-scope="scope">

<span>{{scope.row.schoolName}}</span>

</template>

</el-table-column>-->

<el-table-column align="center" label="操作" min-width="150">

<template slot-scope="scope">

<div>

<!-- <el-link type="primary" @click="permissionAssign(scope.row)">分配权限</el-link>

<el-divider direction="vertical"></el-divider>-->

<el-link type="primary" @click="editData(scope.row)">编辑</el-link>

<el-divider direction="vertical"></el-divider>

<el-link type="danger" @click="resetPass(scope.row)">重置密码</el-link>

</div>

</template>

</el-table-column>

</el-table>

<el-pagination

background

@current-change="handleCurrentChange"

layout="total, prev, pager, next"

:total="totalNum"

v-if="totalNum!=0"

:page-size="pageSize"

></el-pagination>

</div>

<el-dialog :title="title" :visible.sync="dialogFormVisible">

<el-form

:model="ruleForm"

:rules="rules"

ref="ruleForm"

label-width="100px"

class="demo-ruleForm"

>

<el-form-item label="手机号码" prop="telephone">

<el-input v-model="ruleForm.telephone" :disabled="title=='修改用户信息'" placeholder="请输入手机号码"></el-input>

</el-form-item>

<el-form-item label="用户名称" prop="name">

<el-input v-model="ruleForm.name" placeholder="请输入用户名称"></el-input>

</el-form-item>

<el-form-item label="所属机构" prop="schoolName">

<el-input v-model="ruleForm.schoolName" placeholder="请输入用户所属机构"></el-input>

</el-form-item>

<el-form-item label="性别">

<el-radio v-model="ruleForm.sex" label="1">男</el-radio>

<el-radio v-model="ruleForm.sex" label="0">女</el-radio>

</el-form-item>

<el-form-item label="角色" prop="roleId">

<!-- <el-select v-model="ruleForm.role" placeholder="请选择角色" style="width:100%">

<el-option

v-for="item in ruleForm.roles"

:key="item.value"

:label="item.label"

:value="item.value"

></el-option>

</el-select>-->

<!-- <el-radio v-model="ruleForm.roleId" label="0" :disabled="roleAu==false">老师</el-radio>

<el-radio v-model="ruleForm.roleId" label="1" :disabled="roleAu==false">管理员</el-radio> -->

<el-radio v-model="ruleForm.roleId" label="0" >老师</el-radio>

<el-radio v-model="ruleForm.roleId" label="1" >管理员</el-radio>

</el-form-item>

</el-form>

<div slot="footer" class="dialog-footer" style="text-align: center;margin-bottom:10px">

<el-button type="primary" @click="submitForm('ruleForm')" style="width:180px">提交</el-button>

<el-button @click="resetForm('ruleForm')" style="width:180px">取消</el-button>

</div>

<div

slot="footer"

class="dialog-footer"

style="text-align: center;color: #5d5b5b;font-size:14px"

v-if="title=='新增用户'"

>

<span>新增用户状态默认为正常</span>

</div>

</el-dialog>

</div>

</template>

<script>

export default {

data() {

// var checkPhone = (rule, value, callback) => {

// if (!value) {

// return callback(new Error("手机号不能为空"));

// } else {

// const reg = /^1[3|4|5|7|8][0-9]\d{8}$/;

// console.log(reg.test(value));

// if (reg.test(value)) {

// callback();

// } else {

// return callback(new Error("请输入正确的手机号"));

// }

// }

// };

return {

list: [],

AllData: [],

listLoading: false,

multipleSelection: [],

dialogFormVisible: false,

ruleForm: {

name: "",

sex: "0",

roleId: "0",

telephone: ""

},

roleAu: false,

rules: {

name: [{ required: true, message: "请输入用户名称", trigger: "blur" }],

roleId: [{ required: true, message: "请选择角色", trigger: "change" }],

// tel: [{ required: true, validator: checkPhone, trigger: "blur" }],

telephone: [

{ required: true, message: "请输入手机号码", trigger: "blur" },

{

// type: "email",

message: "请输入正确的手机号码",

trigger: ["blur", "change"]

}

]

},

totalNum: 0,

title: "新增用户",

pageSize: 10,

formInline: {

username: "",

state: ""

},

page: 1

};

},

mounted() {

this.getStateAu();

},

created() {

this.showUserInfo(this.page);

this.getStateAu();

},

methods: {

filterState(state) {

if (state == "0") {

return "正常";

} else {

return "禁用";

}

},

btnState(state) {

if (state == "0") {

return "primary";

} else {

return "danger";

}

},

getStateAu() {

var authority = JSON.parse(localStorage.getItem("authority"));

if (authority) {

for (var i = 0; i < authority.length; i++) {

if (authority[i] == "4") {

this.roleAu = true;

}

}

}

},

changeState(row) {

var data = {

telephone: row.telephone

};

var auth = 0;

//恢复权限

var authority = JSON.parse(localStorage.getItem("authority"));

if (authority) {

for (var i = 0; i < authority.length; i++) {

if (authority[i] == "5") {

auth = 1;

}

}

}

var auth1 = 0;

//禁用权限

if (authority) {

for (var i = 0; i < authority.length; i++) {

if (authority[i] == "8") {

auth1 = 1;

}

}

}

if ((!auth && row.state == 1) || (!auth1 && row.state == 0)) {

console.log("patch/userdata = "+JSON.stringify(data));

this.$http.patch("/api/user", null, {

params: data

}).then(

res => {

// success callback

if (res.data.respCode == "1") {

this.$alert("状态修改成功", "成功", {

confirmButtonText: "确定"

});

row.state = !row.state;

this.showUserInfo(this.page);

} else {

this.$alert(res.data.respCode, "失败", {

confirmButtonText: "确定"

});

this.showUserInfo(this.page);

}

this.listLoading = false;

},

res => {

this.$router.push({

path: "/" + res

});

}

);

} else {

if (row.state == 0) {

this.$alert("你没有禁用用户状态权限", {

confirmButtonText: "确定"

});

} else {

this.$alert("你没有恢复用户状态权限", {

confirmButtonText: "确定"

});

}

}

},

resetPass(row) {

//重置密码

var auth = 0;

var authority = JSON.parse(localStorage.getItem("authority"));

if (authority) {

for (var i = 0; i < authority.length; i++) {

if (authority[i] == "6") {

auth = 1;

}

}

}

if (!auth) {

console.log("currow.telephone = "+row.telephone)

this.$http

.put("/api/user/resetPassword?email=" + row.telephone)

.then(res => {

if (res.data.respCode == "1") {

this.$alert("密码重置成功", "成功", {

confirmButtonText: "确定"

});

} else {

this.$alert(res.data.respCode, "失败", {

confirmButtonText: "确定"

});

}

});

} else {

this.$alert("你没有重置密码权限", {

confirmButtonText: "确定"

});

}

},

handleSelectionChange(val) {

this.multipleSelection = val;

},

deleteData() {

//批量删除

var auth = 0;

var authority = JSON.parse(localStorage.getItem("authority"));

if (authority) {

for (var i = 0; i < authority.length; i++) {

if (authority[i] == "2") {

auth = 1;

}

}

}

if (!auth) {

if (this.multipleSelection.length == 0) {

this.$alert("请至少选中一条数据", "批量删除", {

confirmButtonText: "确定"

});

} else {

var telephones = [];

for (var i in this.multipleSelection) {

telephones.push(this.multipleSelection[i].telephone);

}

this.$confirm("确定要删除选择的用户？", "删除用户", {

confirmButtonText: "确定",

cancelButtonText: "取消",

type: "info"

})

.then(() => {

// for (var i in this.multipleSelection) {

// console.log("this.multipleSelection[i] = "+JSON.stringify(this.multipleSelection[i]))

// }

var data = telephones;

console.log("deleteData = "+JSON.stringify(data))

this.$http.delete("/api/user?del\_list=" + data).then(

res => {

if (res.data.respCode == "1") {

this.$alert("用户删除成功", "成功", {

confirmButtonText: "确定"

});

this.page = 1;

this.showUserInfo(this.page);

} else {

this.$alert(res.data.respCode, "失败", {

confirmButtonText: "确定"

});

this.showUserInfo(this.page);

}

this.listLoading = false;

},

res => {

this.$router.push({

path: "/" + res

});

}

);

})

.catch(() => {});

}

} else {

this.$alert("你没有删除用户权限", {

confirmButtonText: "确定"

});

}

},

searchData() {

var auth = 0;

var authority = JSON.parse(localStorage.getItem("authority"));

if (authority) {

for (var i = 0; i < authority.length; i++) {

if (authority[i] == "7") {

auth = 1;

}

}

}

if (!auth) {

this.list = [];

this.listLoading = true;

var roleId = localStorage.getItem("roleId");

if (this.formInline.username == "" && this.formInline.state == "") {

this.showUserInfo(this.page);

} else {

this.page = 1;

roleId = 1;

this.$http

.get(

"/api/user?page=" +

this.page +

"&state=" +

this.formInline.state +

"&name=" +

this.formInline.username +

"&size=" +

10 +

"&roleId=" +

roleId

)

.then(

res => {

this.listLoading = false;

// this.totalNum = res.data[0].totalCount;

if (this.totalNum != 0) {

// delete res.data[0];

this.list = res.data.dataList;

}

},

res => {

this.$router.push({

path: "/" + res

});

}

);

}

} else {

this.$alert("你没有查询用户权限", {

confirmButtonText: "确定"

});

}

},

showUserInfo(page) {

this.list = [];

this.listLoading = true;

this.page = page;

//获取用户信息

var data = {

page: this.page

};

var roleId = localStorage.getItem("roleId");

roleId = 1;

console.log("this.page = "+this.page)

console.log("this.roldId = "+roleId)

var size =10;

var data = {

name: "野人",

page: 1,

roleId: 1,

size: 10,

state:"0"

};

this.$http

.get("/api/user?page=" + this.page + "&state=&name=&roleId=" + 1 +"&size=" + 10)

// .get("/api/user", data)

// this.$http.get("/api/user", null, {

// params: data

// })

.then(

res => {

console.log("succeed succeed succeed succeed succeed")

console.log("Uersres.data = "+JSON.stringify(res.data))

this.list = res.data.dataList;

this.listLoading = false;

this.totalNum = res.data.total;

// this.totalNum = res.data[0].totalCount;

if (this.totalNum != 0) {

// delete res.data[0];

this.list = res.data.dataList;

}

},

res => {

this.$router.push({

path: "/" + res

});

}

);

},

reset() {

this.ruleForm.name = "";

this.ruleForm.sex = "1";

this.ruleForm.roleId = "0";

this.ruleForm.telephone = "";

},

addData() {

var auth = 0;

var authority = JSON.parse(localStorage.getItem("authority"));

if (authority) {

for (var i = 0; i < authority.length; i++) {

if (authority[i] == "1") {

auth = 1;

}

}

}

if (!auth) {

this.reset();

this.dialogFormVisible = true;

this.title = "新增用户";

} else {

this.$alert("你没有新增用户权限", {

confirmButtonText: "确定"

});

}

},

submitForm(formName) {

this.$refs[formName].validate(valid => {

if (valid) {

this.dialogFormVisible = false;

if (this.title == "新增用户") {

var data = {

telephone: this.ruleForm.telephone,

name: this.ruleForm.name,

roleId: this.ruleForm.roleId,

sex: this.ruleForm.sex,

};

this.$http.post("/api/user", data).then(

res => {

if (res.data.respCode == "1") {

this.$alert("用户添加成功", "成功", {

confirmButtonText: "确定"

});

this.showUserInfo(this.page);

} else {

this.$alert(res.data.respCode, "失败", {

confirmButtonText: "确定"

});

this.showUserInfo(this.page);

}

this.listLoading = false;

},

res => {

this.$router.push({

path: "/" + res

});

}

);

} else {

//修改信息

var data = {

name: this.ruleForm.name,

sex: this.ruleForm.sex,

roleId: this.ruleForm.roleId,

telephone: this.ruleForm.telephone

};

this.$http.put("/api/user", data).then(

res => {

if (res.data.respCode == "1") {

this.$alert("用户修改成功", "成功", {

confirmButtonText: "确定"

});

this.showUserInfo(this.page);

} else {

this.$alert(res.data.respCode, "失败", {

confirmButtonText: "确定"

});

this.showUserInfo(this.page);

}

this.listLoading = false;

},

res => {

this.$router.push({

path: "/" + res

});

}

);

}

}

});

},

resetForm(formName) {

this.dialogFormVisible = false;

this.$refs[formName].resetFields();

this.reset();

this.showUserInfo(this.page);

},

handleCurrentChange(val) {

console.log(val);

this.page = val;

this.showUserInfo(this.page);

},

resetData() {

this.formInline.username = "";

this.formInline.state = "";

this.page = 1;

this.showUserInfo(this.page);

},

editData(row) {

this.getStateAu();

var auth = 0;

var authority = JSON.parse(localStorage.getItem("authority"));

console.log(authority);

if (authority) {

for (var i = 0; i < authority.length; i++) {

if (authority[i] == "3") {

auth = 1;

}

}

}

if (!auth) {

this.ruleForm = row;

this.ruleForm.sex = this.ruleForm.sex.toString();

this.ruleForm.roleId = this.ruleForm.roleId.toString();

this.title = "修改用户信息";

this.dialogFormVisible = true;

} else {

this.$alert("你没有编辑用户权限", {

confirmButtonText: "确定"

});

}

},

permissionAssign(row) {

this.$router.push({

path: "/userManage/permissionAssignment",

query: { username: row.name }

});

}

}

};

</script>

数据字典管理功能代码

<template>

<div class="app-container">

<div class="form-style">

<el-form :inline="true" :model="formInline" class="demo-form-inline">

<el-form-item>

<el-button type="primary" size="small" @click="addData()" icon="el-icon-plus">新增</el-button>

</el-form-item>

<el-form-item>

<el-button type="danger" size="small" @click="deleteDatas()" icon="el-icon-delete">删除</el-button>

</el-form-item>

<el-form-item style="float:right">

<el-button type="primary" size="small" @click="resetData()">重置</el-button>

</el-form-item>

<el-form-item style="float:right">

<el-button type="primary" size="small" @click="searchData()">查询</el-button>

</el-form-item>

<el-form-item label="名称：" style="float:right">

<el-input placeholder="请输入名称" size="small" v-model="formInline.chineseMark"></el-input>

</el-form-item>

<el-form-item label="关键字：" style="float:right">

<el-input placeholder="请输入关键字" size="small" v-model="formInline.englishMark"></el-input>

</el-form-item>

</el-form>

<el-table

:data="list"

v-loading="listLoading"

element-loading-text="Loading"

fit

highlight-current-row

ref="multipleTable"

tooltip-effect="dark"

style="width: 100%"

@selection-change="handleSelectionChange"

:header-cell-style="{background:'#eef1f6',color:'#606266'}"

>

<el-table-column type="selection" width="50"></el-table-column>

<el-table-column label="序号" type="index" width="50" align="center">

<template slot-scope="scope">

<span>{{(page - 1) \* pageSize + scope.$index + 1}}</span>

</template>

</el-table-column>

<el-table-column label="名称" min-width="80" align="center">

<template slot-scope="scope">

<span>{{scope.row.name}}</span>

</template>

</el-table-column>

<el-table-column label="关键字" min-width="80" align="center">

<template slot-scope="scope">

<span>{{scope.row.code}}</span>

</template>

</el-table-column>

<el-table-column label="说明" min-width="80" align="center">

<template slot-scope="scope">

<span>{{scope.row.description}}</span>

</template>

</el-table-column>

<el-table-column align="center" label="操作" min-width="150">

<template slot-scope="scope">

<div>

<el-link type="primary" @click="details(scope.row)">详情</el-link>

<el-divider direction="vertical"></el-divider>

<el-link type="primary" @click="editData(scope.row)">编辑</el-link>

<el-divider direction="vertical"></el-divider>

<el-link type="danger" @click="deleteData(scope.row)">删除</el-link>

</div>

</template>

</el-table-column>

</el-table>

<el-pagination

background

@current-change="handleCurrentChange"

layout="total, prev, pager, next"

:total="totalNum"

v-if="totalNum!=0"

:page-size="pageSize"

></el-pagination>

</div>

<el-dialog title="数据字典详情" :visible.sync="dialogFormVisible">

<span>数据字典</span>

<el-divider></el-divider>

<el-form :model="itemForm" ref="itemForm" label-width="100px" class="demo-ruleForm">

<el-row>

<el-col :span="12">

<el-form-item label="名称">

<el-input v-model="itemForm.name" disabled></el-input>

</el-form-item>

</el-col>

<el-col :span="12">

<el-form-item label="关键字">

<el-input v-model="itemForm.code" disabled></el-input>

</el-form-item>

</el-col>

</el-row>

<el-form-item label="说明">

<el-input v-model="itemForm.description" disabled></el-input>

</el-form-item>

</el-form>

<span>数据项</span>

<el-divider></el-divider>

<el-table

:data="dataItem"

v-loading="dataItemLoading"

element-loading-text="Loading"

fit

highlight-current-row

tooltip-effect="dark"

style="width: 100%"

:header-cell-style="{background:'#eef1f6',color:'#606266'}"

>

<el-table-column label="序号" type="index" width="50" align="center">

<template slot-scope="scope">

<span>{{scope.$index}}</span>

</template>

</el-table-column>

<el-table-column label="值" min-width="100" align="center">

<template slot-scope="scope">

<span>{{scope.row.value}}</span>

</template>

</el-table-column>

<el-table-column label="文本" min-width="100" align="center">

<template slot-scope="scope">

<span>{{scope.row.name}}</span>

</template>

</el-table-column>

<el-table-column label="默认值" min-width="100" align="center">

<template slot-scope="scope">

<i class="el-icon-check" v-if="filterState(scope.row.isDefault)"></i>

</template>

</el-table-column>

</el-table>

<div slot="footer" class="dialog-footer">

<el-button @click="dialogFormVisible = false">关闭</el-button>

</div>

</el-dialog>

</div>

</template>

<script>

export default {

data() {

return {

list: [

{

name: "性别",

code: "sex",

description: ""

}

],

listLoading: false,

multipleSelection: [],

totalNum: 0,

pageSize: 10,

formInline: {

chineseMark: "",

englishMark: ""

},

page: 1,

dialogFormVisible: false,

itemForm: {

name: "",

code: "",

description: ""

},

dataItem: [],

dataItemLoading: false

};

},

created() {

this.showDictionary(this.page);

},

methods: {

filterState(state) {

if (state == "1") {

return true;

} else {

return false;

}

},

handleSelectionChange(val) {

this.multipleSelection = val;

},

deleteDatas() {

//批量删除

if (this.multipleSelection.length == 0) {

this.$alert("请至少选中一条数据", "批量删除", {

confirmButtonText: "确定"

});

} else {

var codes = [];

codes.push("-1");

for (var i in this.multipleSelection) {

codes.push(this.multipleSelection[i].code);

}

this.$confirm("确定要删除选择的数据字典？", "删除数据字典", {

confirmButtonText: "确定",

cancelButtonText: "取消",

type: "info"

})

.then(() => {

var data = codes;

this.$http.delete("/api/dictionaries?del\_list=" + codes).then(

res => {

if (res.data.respCode == "1") {

this.$alert("数据字典删除成功", "成功", {

confirmButtonText: "确定"

});

this.page = 1;

this.showDictionary(this.page);

} else {

this.$alert(res.data.respCode, "失败", {

confirmButtonText: "确定"

});

this.showDictionary(this.page);

}

this.listLoading = false;

},

res => {

this.$router.push({

path: "/" + res

});

}

);

})

.catch(() => {});

}

},

deleteData(row) {

//删除

var codes = [];

codes.push("-1");

codes.push(row.code);

this.$confirm("确定要删除该项数据字典？", "删除数据字典", {

confirmButtonText: "确定",

cancelButtonText: "取消",

type: "info"

})

.then(() => {

var data = codes;

this.$http.delete("/api/dictionaries?del\_list=" + codes).then(

res => {

if (res.data.respCode == "1") {

this.$alert("数据字典删除成功", "成功", {

confirmButtonText: "确定"

});

this.page = 1;

this.showDictionary(this.page);

} else {

this.$alert(res.data.respCode, "失败", {

confirmButtonText: "确定"

});

this.showDictionary(this.page);

}

this.listLoading = false;

},

res => {

// error callback

this.$router.push({

path: "/" + res

});

}

);

})

.catch(() => {});

},

searchData() {

//查询

this.list = [];

this.listLoading = true;

this.page = 1;

var data = {

page: this.page,

name: this.formInline.chineseMark,

code: this.formInline.englishMark

};

this.$http

.get(

"/api/dictionaries?page=" +

this.page +

"&name=" +

this.formInline.chineseMark +

"&code=" +

this.formInline.englishMark

)

.then(

res => {

// success callback

this.listLoading = false;

this.totalNum = res.data.total;

if (this.totalNum != 0) {

this.list = res.data.records;

}

},

res => {

this.$router.push({

path: "/" + res

});

}

);

},

showDictionary(page) {

//获取数据

this.list = [];

this.listLoading = true;

this.page = page;

this.$http.get("/api/dictionaries?page=" + this.page).then(

res => {

// success callback

console.log(res);

this.listLoading = false;

this.totalNum = res.data.total;

if (this.totalNum != 0) {

this.list = res.data.records;

}

},

res => {

// this.$message.error('请求失败! '+res);

this.listLoading = false;

this.$router.push({

path: "/" + res

});

}

);

},

handleCurrentChange(val) {

//分页

console.log(val);

this.page = val;

this.showDictionary(this.page);

},

resetData() {

//重置

this.formInline.chineseMark = "";

this.formInline.englishMark = "";

this.page = 1;

this.showDictionary(this.page);

},

addData() {

//新增数据字典

this.$router.push({

path: "/dataDictionary/addDictionary"

});

},

editData(row) {

//修改数据字典

this.$router.push({

path: "/dataDictionary/addDictionary",

query: { code: row.code }

});

},

details(row) {

//查看详情

this.dialogFormVisible = true;

this.dataItemLoading = true;

this.$http.get("/api/dictionaries?code=" + row.code).then(

res => {

this.dataItemLoading = false;

console.log(res.data.detail);

this.itemForm = res.data.dict[0];

this.dataItem = res.data.detail;

},

res => {

this.$router.push({

path: "/" + res

});

}

);

}

}

};

</script>

菜单管理功能代码

<template>

<div class="app-container">

<div class="form-style">

<el-form :inline="true" :model="formInline" class="demo-form-inline">

<el-form-item>

<el-button type="primary" size="small" @click="addData()" icon="el-icon-plus">新增</el-button>

</el-form-item>

<el-form-item>

<el-button type="danger" size="small" @click="deleteData()" icon="el-icon-delete">删除</el-button>

</el-form-item>

<el-form-item style="float:right">

<el-button type="primary" size="small" @click="resetData()">重置</el-button>

</el-form-item>

<el-form-item style="float:right">

<el-button type="primary" size="small" @click="searchData()">查询</el-button>

</el-form-item>

<el-form-item label="状态：" style="float:right">

<el-select v-model="formInline.state" placeholder="请选择状态" size="small">

<el-option label="正常" value="1"></el-option>

<el-option label="禁用" value="2"></el-option>

</el-select>

</el-form-item>

<el-form-item label="菜单名称：" style="float:right">

<el-input placeholder="请输入菜单名称" size="small" v-model="formInline.menus"></el-input>

</el-form-item>

</el-form>

<el-table

:data="list"

v-loading="listLoading"

element-loading-text="Loading"

border

fit

highlight-current-row

ref="multipleTable"

tooltip-effect="dark"

style="width: 100%"

@selection-change="handleSelectionChange"

row-key="id"

:tree-props="{children: 'children', hasChildren: 'hasChildren'}"

:header-cell-style="{background:'#eef1f6',color:'#606266'}"

>

<el-table-column type="selection" width="50"></el-table-column>

<!--

<el-table-column

v-for="(item,index) in tableList"

:key="index"

:label="item.label"

:prop="item.prop"

></el-table-column>

-->

<el-table-column label="菜单名称" min-width="70" align="center">

<template slot-scope="scope">

<span>{{scope.row.name}}</span>

</template>

</el-table-column>

<el-table-column label="图标" min-width="50" align="center">

<template slot-scope="scope">

<i :class="scope.row.icon"></i>

</template>

</el-table-column>

<el-table-column label="层级" min-width="50" align="center">

<template slot-scope="scope">

<span>{{scope.row.menuLevel}}级</span>

</template>

</el-table-column>

<el-table-column label="父菜单" min-width="50" align="center">

<template slot-scope="scope">

<span>{{scope.row.parentId}}</span>

</template>

</el-table-column>

<el-table-column label="排序" min-width="50" align="center">

<template slot-scope="scope">

<span>{{scope.row.menuOrder}}</span>

</template>

</el-table-column>

<el-table-column label="是否可见" min-width="70" align="center">

<template slot-scope="scope">

<i class="el-icon-check" v-if="scope.row.isVisible=='1'" style="color:green"></i>

<i class="el-icon-close" v-if="scope.row.isVisible=='0'" style="color:red"></i>

</template>

</el-table-column>

<el-table-column label="是否是菜单" min-width="70" align="center">

<template slot-scope="scope">

<i class="el-icon-check" v-if="scope.row.isMenu=='1'" style="color:green"></i>

<i class="el-icon-close" v-if="scope.row.isMenu=='0'" style="color:red"></i>

</template>

</el-table-column>

<el-table-column label="是否是页面" min-width="70" align="center">

<template slot-scope="scope">

<i class="el-icon-check" v-if="scope.row.isPage=='1'" style="color:green"></i>

<i class="el-icon-close" v-if="scope.row.isPage=='0'" style="color:red"></i>

</template>

</el-table-column>

<el-table-column label="请求地址" min-width="80" align="center">

<template slot-scope="scope">

<span>{{scope.row.url}}</span>

</template>

</el-table-column>

<el-table-column align="center" label="操作" min-width="80">

<template slot-scope="scope">

<div>

<el-link type="primary" @click="editData(scope.row)">编辑</el-link>

<el-divider direction="vertical"></el-divider>

<el-link type="danger" @click="deleteUser(scope.row)">删除</el-link>

</div>

</template>

</el-table-column>

</el-table>

<el-pagination

background

@current-change="handleCurrentChange"

layout="total, prev, pager, next"

:total="totalNum"

v-if="totalNum!=0"

:page-size="pageSize"

></el-pagination>

</div>

<el-dialog :title="title" :visible.sync="dialogFormVisible">

<el-form

:model="menuForm"

:rules="menus"

ref="menuForm"

label-width="150px"

class="demo-menuForm"

>

<el-form-item label="上级菜单" prop="parentId">

<el-select v-model="menuForm.parentId" placeholder="请选择上级菜单">

<el-option v-for="item in parentList" :key="item" :value="item"></el-option>

</el-select>

</el-form-item>

<el-form-item label="菜单名称" prop="name">

<el-input v-model="menuForm.name"></el-input>

</el-form-item>

<el-form-item label="图标">

<el-input v-model="menuForm.icon"></el-input>

</el-form-item>

<el-form-item>

<span>图标名：https://element.eleme.cn/#/zh-CN/component/icon</span>

</el-form-item>

<el-form-item label="是否是页面" prop="isPage">

<el-radio v-model="menuForm.isPage" label="1">是</el-radio>

<el-radio v-model="menuForm.isPage" label="0">否</el-radio>

</el-form-item>

<el-form-item label="请求地址">

<el-input v-model="menuForm.url"></el-input>

</el-form-item>

<el-form-item label="是否是菜单" prop="isMenu">

<el-radio v-model="menuForm.isMenu" label="1">是</el-radio>

<el-radio v-model="menuForm.isMenu" label="0">否</el-radio>

</el-form-item>

<el-form-item label="菜单状态(默认可见)">

<el-radio v-model="menuForm.isVisible" label="1">可见</el-radio>

<el-radio v-model="menuForm.isVisible" label="0">不可见</el-radio>

</el-form-item>

<el-form-item label="层级" prop="menuLevel">

<el-input v-model="menuForm.menuLevel"></el-input>

</el-form-item>

<el-form-item label="显示排序" prop="menuOrder">

<el-input v-model="menuForm.menuOrder"></el-input>

</el-form-item>

</el-form>

<div slot="footer" class="dialog-footer">

<el-button type="primary" @click="submitForm('menuForm')">提交</el-button>

<el-button @click="resetForm('menuForm')">取消</el-button>

</div>

</el-dialog>

</div>

</template>

<script>

export default {

data() {

return {

list: [],

tableList: [

{

label: "菜单名称",

prop: "name"

},

{

label: "图标",

prop: "icon"

},

{

label: "层级",

prop: "menuOrder"

},

{

label: "父菜单",

prop: "parentId"

},

{

label: "排序",

prop: "menuOrder"

},

{

label: "是否可见",

prop: "isVisible"

},

{

label: "是否是菜单",

prop: "isMenu"

},

{

label: "是否是页面",

prop: "isPage"

},

{

label: "请求地址",

prop: "url"

}

],

AllData: [],

listLoading: false,

multipleSelection: [],

dialogFormVisible: false,

parentList: [],

menuForm: {

parentId: "",

name: "",

icon: "",

isPage: "",

url: "",

isVisible: "1",

menuOrder: "",

menuLevel: ""

},

menus: {

parentId: [

{ required: true, message: "请选择上级菜单", trigger: "change" }

],

name: [{ required: true, message: "请输入", trigger: "blur" }],

isPage: [{ required: true, message: "请选择", trigger: "blur" }],

isMenu: [{ required: true, message: "请选择", trigger: "blur" }],

menuOrder: [{ required: true, message: "请输入", trigger: "blur" }],

menuLevel: [{ required: true, message: "请输入", trigger: "blur" }]

},

totalNum: 0,

title: "新增用户",

pageSize: 10,

page: 1,

formInline: {

menus: "",

state: ""

}

};

},

filters: {

statusFilter(status) {

const statusMap = {

published: "success",

draft: "gray",

deleted: "danger"

};

return statusMap[status];

}

},

created() {

this.showMenuData(this.page);

},

methods: {

reset() {

this.menuForm.parentId = "";

this.menuForm.name = "";

this.menuForm.icon = "";

this.menuForm.isPage = "";

this.menuForm.url = "";

this.menuForm.isVisible = "1";

this.menuForm.menuOrder = "";

this.menuForm.menuLevel = "";

},

handleSelectionChange(val) {

this.multipleSelection = val;

},

addData() {

var auth = 0;

var authority = JSON.parse(localStorage.getItem("authority"));

for (var i = 0; i < authority.length; i++) {

if (authority[i] == "9") {

auth = 1;

}

}

if (!auth) {

this.reset();

this.dialogFormVisible = true;

this.title = "新增菜单";

} else {

this.$alert("你没有新增菜单权限", {

confirmButtonText: "确定"

});

}

},

searchData() {

var auth = 0;

var authority = JSON.parse(localStorage.getItem("authority"));

if (authority) {

for (var i = 0; i < authority.length; i++) {

if (authority[i] == "12") {

auth = 1;

}

}

}

if (auth) {

this.list = [];

this.listLoading = true;

this.page = 1;

var data = {

page: this.page,

name: this.formInline.menus,

state: this.formInline.state

};

this.$http

.get(

"/api/menus?page=" +

data.page +

"&name=" +

data.name +

"&is\_visible=" +

data.state

)

.then(res => {

this.listLoading = false;

this.totalNum = res.data.total;

if (this.totalNum != 0) {

this.list = res.data.records;

var data = res.data.records;

function toTree(data) {

data.forEach(function(item) {

delete item.children;

});

var map = {};

data.forEach(function(item) {

map[item.id] = item;

});

// console.log(map);

var val = [];

data.forEach(function(item) {

var parent = map[item.parentId];

if (parent) {

(parent.children || (parent.children = [])).push(item);

} else {

val.push(item);

}

});

return val;

}

this.list = toTree(data);

for (var i = 0; i < this.list.length; i++) {

if (this.list[i].parentId == 0) {

this.list[i].parentId = "无";

}

if (this.list[i].children != null) {

for (var j = 0; j < this.list[i].children.length; j++) {

this.list[i].children[j].parentId = this.list[i].name;

}

}

}

}

});

} else {

this.$alert("你没有查询菜单权限", {

confirmButtonText: "确定"

});

}

},

deleteUser(row) {

//单个删除

var auth = 0;

var authority = JSON.parse(localStorage.getItem("authority"));

if (authority) {

for (var i = 0; i < authority.length; i++) {

if (authority[i] == "11") {

auth = 1;

}

}

}

if (!auth) {

var del\_list = [];

del\_list.push(row.id);

this.$confirm("确定要删除该菜单？", "删除菜单", {

confirmButtonText: "确定",

cancelButtonText: "取消",

type: "info"

})

.then(() => {

this.$http.delete("/api/menus?ids=" + del\_list).then(res => {

if (res.data.respCode == "1") {

this.$alert("删除成功", "成功", {

confirmButtonText: "确定"

}).then(() => {

this.listLoading = true;

this.$http.get("/api/menus").then(res => {

localStorage.setItem("menuList", JSON.stringify(res.data));

});

location.reload();

this.showMenuData(this.page);

});

}

});

})

.catch(() => {});

} else {

this.$alert("你没有删除菜单权限", {

confirmButtonText: "确定"

});

}

},

deleteData() {

//批量删除

var auth = 0;

var authority = JSON.parse(localStorage.getItem("authority"));

if (authority) {

for (var i = 0; i < authority.length; i++) {

if (authority[i] == "11") {

auth = 1;

}

}

}

if (!auth) {

if (this.multipleSelection.length == 0) {

this.$alert("请至少选中一条数据", "批量删除", {

confirmButtonText: "确定"

});

} else {

var del\_list = [];

for (var i in this.multipleSelection) {

del\_list.push(this.multipleSelection[i].id);

}

this.$confirm("确定要删除所选择的菜单？", "删除菜单", {

confirmButtonText: "确定",

cancelButtonText: "取消",

type: "info"

})

.then(() => {

this.$http.delete("/api/menus?ids=" + del\_list).then(res => {

if (res.data.respCode == "1") {

this.$alert("删除成功", "成功", {

confirmButtonText: "确定"

}).then(() => {

this.listLoading = true;

this.$http.get("/api/menus").then(res => {

localStorage.setItem(

"menuList",

JSON.stringify(res.data)

);

});

location.reload();

this.showMenuData(this.page);

});

}

});

})

.catch(() => {});

}

} else {

this.$alert("你没有删除菜单权限", {

confirmButtonText: "确定"

});

}

},

parentFind(id) {

//寻找父菜单

var parentName = "";

for (var i = 0; i < this.list.length; i++) {

if (id == this.list[i].id) {

parentName = this.list[i].name;

}

}

return parentName;

},

editData(row) {

var auth = 0;

var authority = JSON.parse(localStorage.getItem("authority"));

if (authority) {

for (var i = 0; i < authority.length; i++) {

if (authority[i] == "10") {

auth = 1;

}

}

}

if (!auth) {

this.menuForm = row;

this.menuForm.parentId = this.menuForm.parentId.toString();

this.menuForm.isPage = this.menuForm.isPage.toString();

this.menuForm.isMenu = this.menuForm.isMenu.toString();

this.menuForm.isVisible = this.menuForm.isVisible.toString();

this.title = "编辑菜单";

this.dialogFormVisible = true;

} else {

this.$alert("你没有编辑菜单权限", {

confirmButtonText: "确定"

});

}

},

showMenuData(page) {

this.list = [];

this.listLoading = true;

this.page = page;

//获取用户信息

var data = {

page: this.page

};

this.$http.get("/api/menus").then(res => {

localStorage.setItem("menuList", JSON.stringify(res.data));

});

this.$http.get("/api/menus?page=" + page).then(res => {

this.listLoading = false;

this.totalNum = res.data.total;

if (this.totalNum != 0) {

delete res.data[0];

console.log("res.data.records = "+JSON.stringify(res.data.records))

this.list = res.data.records;

var data = res.data.records;

function toTree(data) {

data.forEach(function(item) {

delete item.children;

});

var map = {};

data.forEach(function(item) {

map[item.id] = item;

});

// console.log(map);

var val = [];

data.forEach(function(item) {

var parent = map[item.parentId];

if (parent) {

(parent.children || (parent.children = [])).push(item);

} else {

val.push(item);

}

});

return val;

}

this.list = toTree(data);

for (var i = 0; i < this.list.length; i++) {

if (this.list[i].parentId == 0) {

this.list[i].parentId = "无";

}

if (this.list[i].children != null) {

for (var j = 0; j < this.list[i].children.length; j++) {

this.list[i].children[j].parentId = this.list[i].name;

}

}

}

}

});

this.getParentList();

},

getParentList() {

this.$http.get("/api/menus?parent=1").then(res => {

this.listLoading = false;

this.parentList = res.data;

this.parentList.push("无");

});

},

addUser() {

this.reset();

this.dialogFormVisible = true;

this.title = "新增菜单";

},

submitForm(formName) {

var data = {

id: this.menuForm.id,

parent\_name: this.menuForm.parentId,

name: this.menuForm.name,

icon: this.menuForm.icon,

is\_page: this.menuForm.isPage,

url: this.menuForm.url,

is\_visible: this.menuForm.isVisible,

is\_menu: this.menuForm.isMenu,

menu\_order: this.menuForm.menuOrder,

menu\_level: this.menuForm.menuLevel

};

var addData = {

parent\_name: this.menuForm.parentId,

name: this.menuForm.name,

icon: this.menuForm.icon,

is\_page: this.menuForm.isPage,

url: this.menuForm.url,

is\_visible: this.menuForm.isVisible,

is\_menu: this.menuForm.isMenu,

menu\_order: this.menuForm.menuOrder,

menu\_level: this.menuForm.menuLevel

};

this.$refs[formName].validate(valid => {

if (valid) {

this.dialogFormVisible = false;

if (this.title == "新增菜单") {

console.log("this.addData ====="+JSON.stringify(addData))

this.$http.post("/api/menus", addData).then(res => {

if (res.data.respCode == "1") {

this.$alert("菜单新增成功", "成功", {

confirmButtonText: "确定"

}).then(() => {

this.listLoading = true;

this.$http.get("/api/menus").then(res => {

localStorage.setItem("menuList", JSON.stringify(res.data));

});

location.reload();

this.showMenuData(this.page);

});

} else {

this.$alert(res.data.respCode, "失败", {

confirmButtonText: "确定"

});

this.showMenuData(this.page);

}

this.$refs[formName].resetFields();

});

} else {

this.$http.patch("/api/menus", data).then(res => {

if (res.data.respCode == "1") {

this.$http.get("/api/menus").then(res => {

localStorage.setItem("menuList", JSON.stringify(res.data));

});

this.$alert("菜单修改成功", "成功", {

confirmButtonText: "确定"

}).then(() => {

this.listLoading = true;

location.reload();

this.showMenuData(this.page);

});

} else {

this.$alert("菜单修改失败", "失败", {

confirmButtonText: "确定"

});

this.showMenuData(this.page);

}

});

}

}

});

},

resetForm(formName) {

this.dialogFormVisible = false;

this.$refs[formName].resetFields();

this.reset();

this.showMenuData(this.page);

},

handleCurrentChange(val) {

this.page = val;

this.showMenuData(this.page);

},

resetData() {

(this.formInline.menus = ""),

(this.formInline.state = ""),

this.showMenuData(this.page);

}

}

};

</script>

系统参数管理功能代码

<template>

<div class="app-container">

<div class="form-style">

<el-form :inline="true" :model="formInline" class="demo-form-inline">

<el-form-item>

<el-button type="primary" size="small" @click="addData()" icon="el-icon-plus">新增</el-button>

</el-form-item>

<el-form-item>

<el-button type="danger" size="small" @click="deleteData()" icon="el-icon-delete">删除</el-button>

</el-form-item>

</el-form>

<el-table

:data="list"

v-loading="listLoading"

element-loading-text="Loading"

border

fit

highlight-current-row

ref="multipleTable"

tooltip-effect="dark"

style="width: 100%"

@selection-change="handleSelectionChange"

row-key="id"

:tree-props="{children: 'children', hasChildren: 'hasChildren'}"

:header-cell-style="{background:'#eef1f6',color:'#606266'}"

>

<el-table-column type="selection" width="50"></el-table-column>

<!--

<el-table-column

v-for="(item,index) in tableList"

:key="index"

:label="item.label"

:prop="item.prop"

></el-table-column>

-->

<el-table-column label="参数名称" min-width="70" align="center">

<template slot-scope="scope">

<span>{{scope.row.paraName}}</span>

</template>

</el-table-column>

<el-table-column label="关键字" min-width="50" align="center">

<template slot-scope="scope">

<span>{{scope.row.keyName}}</span>

</template>

</el-table-column>

<el-table-column label="值" min-width="80" align="center">

<template slot-scope="scope">

<span>{{scope.row.keyValue}}</span>

</template>

</el-table-column>

<el-table-column align="center" label="操作" min-width="80">

<template slot-scope="scope">

<div>

<el-link type="primary" @click="editData(scope.row)">编辑</el-link>

<el-divider direction="vertical"></el-divider>

<el-link type="danger" @click="deleteUser(scope.row)">删除</el-link>

</div>

</template>

</el-table-column>

</el-table>

<el-pagination

background

@current-change="handleCurrentChange"

layout="total, prev, pager, next"

:total="totalNum"

v-if="totalNum!=0"

:page-size="pageSize"

></el-pagination>

</div>

<el-dialog :title="title" :visible.sync="dialogFormVisible">

<el-form

:model="menuForm"

:rules="menus"

ref="menuForm"

label-width="150px"

class="demo-menuForm"

>

<el-form-item label="参数名称" prop="paraName">

<el-input v-model="menuForm.paraName"></el-input>

</el-form-item>

<el-form-item label="关键字" prop="keyName">

<el-input v-model="menuForm.keyName"></el-input>

</el-form-item>

<el-form-item label="值" prop="keyValue">

<el-input v-model="menuForm.keyValue"></el-input>

</el-form-item>

</el-form>

<div slot="footer" class="dialog-footer">

<el-button type="primary" @click="submitForm('menuForm')">保存</el-button>

<el-button @click="resetForm('menuForm')">取消</el-button>

</div>

</el-dialog>

</div>

</template>

<script>

export default {

data() {

return {

list: [],

tableList: [

{

label: "参数名称",

prop: "paraName"

},

{

label: "关键字",

prop: "keyName"

},

{

label: "值",

prop: "keyValue"

},

],

AllData: [],

listLoading: false,

multipleSelection: [],

dialogFormVisible: false,

parentList: [],

menuForm: {

parentId: "",

paraName: "",

keyValue: "",

keyName: ""

},

menus: {

parentId: [

{ required: true, message: "请选择上级菜单", trigger: "change" }

],

paraName: [{ required: true, message: "参数名称必填", trigger: "blur" }],

keyName: [{ required: true, message: "关键字必填", trigger: "blur" }],

keyValue: [{ required: true, message: "值必填", trigger: "blur" }],

},

totalNum: 0,

title: "新增用户",

pageSize: 10,

page: 1,

formInline: {

menus: "",

state: ""

}

};

},

filters: {

statusFilter(status) {

const statusMap = {

published: "success",

draft: "gray",

deleted: "danger"

};

return statusMap[status];

}

},

created() {

this.showMenuData(this.page);

},

methods: {

reset() {

this.menuForm.parentId = "";

this.menuForm.paraName = "";

this.menuForm.keyValue = "";

this.menuForm.keyName = "";

},

handleSelectionChange(val) {

this.multipleSelection = val;

},

addData() {

var auth = 0;

var authority = JSON.parse(localStorage.getItem("authority"));

for (var i = 0; i < authority.length; i++) {

if (authority[i] == "9") {

auth = 1;

}

}

if (!auth) {

this.reset();

this.dialogFormVisible = true;

this.title = "新增系统参数";

} else {

this.$alert("你没有新增菜单权限", {

confirmButtonText: "确定"

});

}

},

deleteUser(row) {

//单个删除

var auth = 0;

var authority = JSON.parse(localStorage.getItem("authority"));

if (authority) {

for (var i = 0; i < authority.length; i++) {

if (authority[i] == "11") {

auth = 1;

}

}

}

if (!auth) {

// var del\_list = [];

// del\_list.push(row.id);

console.log("this.del\_list = "+ JSON.stringify(row))

this.$confirm("确定要删除该菜单？", "删除菜单", {

confirmButtonText: "确定",

cancelButtonText: "取消",

type: "info"

})

.then(() => {

this.$http.delete("/api/systems?key\_name=" + row.keyName).then(res => {

console.log("res.data.respCode = "+res.data.respCode)

if (res.data.respCode == "1") {

this.$alert("删除成功", "成功", {

confirmButtonText: "确定"

}).then(() => {

this.listLoading = true;

this.$http.get("/api/menus").then(res => {

localStorage.setItem("menuList", JSON.stringify(res.data));

});

location.reload();

this.showMenuData(this.page);

});

}

});

})

.catch(() => {});

} else {

this.$alert("你没有删除菜单权限", {

confirmButtonText: "确定"

});

}

},

deleteData() {

//批量删除

var auth = 0;

var authority = JSON.parse(localStorage.getItem("authority"));

if (authority) {

for (var i = 0; i < authority.length; i++) {

if (authority[i] == "11") {

auth = 1;

}

}

}

if (!auth) {

if (this.multipleSelection.length == 0) {

this.$alert("请至少选中一条数据", "批量删除", {

confirmButtonText: "确定"

});

} else {

var del\_list = [];

for (var i in this.multipleSelection) {

del\_list.push(this.multipleSelection[i].id);

}

this.$confirm("确定要删除所选择的菜单？", "删除菜单", {

confirmButtonText: "确定",

cancelButtonText: "取消",

type: "info"

})

.then(() => {

this.$http.delete("/api/menus?ids=" + del\_list).then(res => {

if (res.data.respCode == "1") {

this.$alert("删除成功", "成功", {

confirmButtonText: "确定"

}).then(() => {

this.listLoading = true;

this.$http.get("/api/menus").then(res => {

localStorage.setItem(

"menuList",

JSON.stringify(res.data)

);

});

location.reload();

this.showMenuData(this.page);

});

}

});

})

.catch(() => {});

}

} else {

this.$alert("你没有删除菜单权限", {

confirmButtonText: "确定"

});

}

},

editData(row) {

var auth = 0;

var authority = JSON.parse(localStorage.getItem("authority"));

if (authority) {

for (var i = 0; i < authority.length; i++) {

if (authority[i] == "10") {

auth = 1;

}

}

}

if (!auth) {

this.menuForm = row;

// parentId: "",

// paraName: "",

// keyValue: "",

// keyName: ""

// this.menuForm.parentId = this.menuForm.parentId.toString();

// this.menuForm.paraName = this.menuForm.paraName.toString();

// this.menuForm.keyValue = this.menuForm.keyValue.toString();

// this.menuForm.keyName = this.menuForm.keyName.toString();

this.title = "编辑系统参数";

this.dialogFormVisible = true;

} else {

this.$alert("你没有编辑系统参数权限", {

confirmButtonText: "确定"

});

}

},

showMenuData(page) {

//获取数据

this.list = [];

this.listLoading = true;

this.page = page;

this.$http.get("/api/systems?page=" + this.page).then(

res => {

// success callback

console.log(res);

this.listLoading = false;

this.totalNum = res.data.total;

if (this.totalNum != 0) {

this.list = res.data.records;

}

},

res => {

// this.$message.error('请求失败! '+res);

this.listLoading = false;

this.$router.push({

path: "/" + res

});

}

);

},

getParentList() {

this.$http.get("/api/menus?parent=1").then(res => {

this.listLoading = false;

this.parentList = res.data;

this.parentList.push("无");

});

},

addUser() {

this.reset();

this.dialogFormVisible = true;

this.title = "新增系统参数";

},

submitForm(formName) {

var data = {

// id: this.menuForm.id,

key\_name: this.menuForm.keyName,

key\_value: this.menuForm.keyValue,

para\_name: this.menuForm.paraName,

};

var addData = {

key\_name: this.menuForm.keyName,

key\_value: this.menuForm.keyValue,

para\_name: this.menuForm.paraName,

};

// if(thsi.menuForm.keyValue == "distance"){

// alert("关键字重复")

// }

this.$refs[formName].validate(valid => {

if (valid) {

this.dialogFormVisible = false;

if (this.title == "新增系统参数") {

console.log("this.title is right//////////////////////ok")

// console.log("this.data2222222222222222222 = "+JSON.stringify(data))

this.showMenuData(data);

this.$http.post("/api/systems", null, {

params: addData

}).then(

res => {

console.log("res.data == "+JSON.stringify(res.data))

// // console.log("res.data.respCode = "+res.data.respCode)

if (res.data.respCode == "1") {

this.$alert("系统参数新增成功", "成功", {

confirmButtonText: "确定"

}).then(() => {

this.listLoading = true;

this.$http.get("/api/menus").then(res => {

localStorage.setItem("menuList", JSON.stringify(res.data));

});

location.reload();

this.showMenuData(this.page);

});

} else {

this.$alert(res.data.respCode, "失败", {

confirmButtonText: "确定"

});

this.showMenuData(this.page);

}

// this.$router.push({path: "/dataDictionary"})

},

// res => {

// this.$router.push({

// path: "/" + res

// });

// }

);

} else {

console.log("修改系统的data = "+JSON.stringify(addData))

this.$http.patch("/api/systems", null, {

params: addData

}).then(res => {

if (res.data.respCode == "1") {

console.log("修改系统参数返回data = "+JSON.stringify(res.data))

this.$http.get("/api/menus").then(res => {

localStorage.setItem("menuList", JSON.stringify(res.data));

});

this.$alert("系统参数修改成功", "成功", {

confirmButtonText: "确定"

}).then(() => {

this.listLoading = true;

location.reload();

this.showMenuData(this.page);

});

} else {

this.$alert("系统参数修改失败", "失败", {

confirmButtonText: "确定"

});

this.showMenuData(this.page);

}

});

}

}

});

},

resetForm(formName) {

this.dialogFormVisible = false;

this.$refs[formName].resetFields();

this.reset();

this.showMenuData(this.page);

},

handleCurrentChange(val) {

this.page = val;

this.showMenuData(this.page);

},

}

};

</script>

# 项目配置文件

'use strict'

// Template version: 1.2.6

// see http://vuejs-templates.github.io/webpack for documentation.

const path = require('path')

module.exports = {

dev: {

// Paths

assetsSubDirectory: 'static',

// assetsPublicPath: './',

assetsPublicPath: '/',

proxyTable: {

'/api': {

// target: 'http://localhost:8080', //设置你调用的接口域名和端口号

target: 'http://120.77.220.113:8080', //设置你调用的接口域名和端口号

changeOrigin: true,

   //跨域

pathRewrite: {

'^/api': '/'     //这里理解成用‘/api’代替target里面的地址，后面组件中我们掉接口时直接用api代替 比如我要调用'http://10.1.5.11:8080/xxx/duty?time=2017-07-07 14:57:22'，直接写‘/api/xxx/duty?time=2017-07-07 14:57:22’即可

}

},

// '/api': { //代理标识

// target: 'http://localhost:8081',

// changeOrigin: true, // 允许跨域

// secure: false,

// pathRewrite: {

// '^.api': '/'

// }

// },

'/apis': { //代理标识

target: 'http://localhost:8081',

changeOrigin: true, // 允许跨域

secure: false,

pathRewrite: {

'^.apis': '/'

}

},

'/apiss': { //代理标识

target: 'http://localhost:8081',

changeOrigin: true, // 允许跨域

secure: false,

pathRewrite: {

'^.apiss': '/'

}

},

},

// Various Dev Server settings

// host: 'localhost', // can be overwritten by process.env.HOST

host: '0.0.0.0',

port: 8081, // can be overwritten by process.env.PORT, if port is in use, a free one will be determined

autoOpenBrowser: true,

errorOverlay: true,

notifyOnErrors: false,

poll: false, // https://webpack.js.org/configuration/dev-server/#devserver-watchoptions-

// Use Eslint Loader?

// If true, your code will be linted during bundling and

// linting errors and warnings will be shown in the console.

useEslint: true,

// If true, eslint errors and warnings will also be shown in the error overlay

// in the browser.

showEslintErrorsInOverlay: false,

/\*\*

\* Source Maps

\*/

// https://webpack.js.org/configuration/devtool/#development

devtool: 'cheap-source-map',

// If you have problems debugging vue-files in devtools,

// set this to false - it \*may\* help

// https://vue-loader.vuejs.org/en/options.html#cachebusting

cacheBusting: true,

// CSS Sourcemaps off by default because relative paths are "buggy"

// with this option, according to the CSS-Loader README

// (https://github.com/webpack/css-loader#sourcemaps)

// In our experience, they generally work as expected,

// just be aware of this issue when enabling this option.

cssSourceMap: false,

},

build: {

// Template for index.html

index: path.resolve(\_\_dirname, '../dist/index.html'),

// Paths

assetsRoot: path.resolve(\_\_dirname, '../dist'),

assetsSubDirectory: 'static',

proxyTable: {

'/api': {

target: 'http://120.77.220.113:8080',

changeOrigin: true,

   //跨域

pathRewrite: {

'^/api': '/'     //这里理解成用‘/api’代替target里面的地址，后面组件中我们掉接口时直接用api代替 比如我要调用'http://10.1.5.11:8080/xxx/duty?time=2017-07-07 14:57:22'，直接写‘/api/xxx/duty?time=2017-07-07 14:57:22’即可

}

}

},

/\*\*

\* You can set by youself according to actual condition

\* You will need to set this if you plan to deploy your site under a sub path,

\* for example GitHub pages. If you plan to deploy your site to https://foo.github.io/bar/,

\* then assetsPublicPath should be set to "/bar/".

\* In most cases please use '/' !!!

\*/

// assetsPublicPath: '/admin/', // If you are deployed on the root path, please use '/'

assetsPublicPath: './',

/\*\*

\* Source Maps

\*/

productionSourceMap: false,

// https://webpack.js.org/configuration/devtool/#production

devtool: '#source-map',

// Gzip off by default as many popular static hosts such as

// Surge or Netlify already gzip all static assets for you.

// Before setting to `true`, make sure to:

// npm install --save-dev compression-webpack-plugin

productionGzip: false,

productionGzipExtensions: ['js', 'css'],

// Run the build command with an extra argument to

// View the bundle analyzer report after build finishes:

// `npm run build --report`

// Set to `true` or `false` to always turn it on or off

bundleAnalyzerReport: process.env.npm\_config\_report

}

}