# 前端代码

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8" />
<meta
name="viewport"
content="width=device-width, user-scalable=no, initial-scale=1.0, maximum-scale=1.0,
minimum-scale=1.0"
/>
<meta http-equiv="X-UA-Compatible" content="ie=edge" />
<title>高拍仪</title>
<link rel="stylesheet" href="css/common.css" />
<link rel="stylesheet" href="css/iconfont.css" />
<link rel="stylesheet" href="css/login.css" />
</head>
<body class="body">
<!-- <div class="header"></div> -->
<!-- <div class="min click" style="right:200px;"><i class="iconfont zuixiaohua"></i>最小
化</div>
<div class="exit click" style="position: absolute; top: 30px; right: 60px;"><i class="iconfont
tuichu1"></i>退出</div> -->
<div class="login-content"><img src="./image/loginleft.png"/>
<div class="login-wrap">
<div class="logo">
<img src="image/OrangeLogo.png" alt="" width="250px" />
</div>
<div class="title">运动营养咨询与指导教学平台</div>
<div class="login">
<input type="text" class="user" placeholder="请输入账号" />
<input type="password" class="password" placeholder="请输入密码" />
</div>
<div class="button click">登录</div>
</div>
</div>

<script src="js/jquery-3.5.1.js"></script>
<script src="js/common.js"></script>
<script src="js/ajax.js"></script>
<script src="js/login.js"></script>
</body>
</html>
$(function () {

// 为所有的 input 元素添加点击事件监听器
$('input').on('click', onInputClick);

if (checkMqttMateCode() == "") {
window.open("mqttMatecode.html", "_self");
return;
```

```
}

//$('.user').on('click', function () {
// console.log('user')
// asynFire.route("Common", "StartKeyBoardFun");
// })
// $('.password').on('click',function () {
// console.log('password')
// asynFire.route("Common", "StartKeyBoardFun");
// })

// 关闭 loading
$.loading(false)

// 点击登录
$('.button').click(function () {
let name = $(".user").val();
let password = $(".password").val();
if (!name || !password) {
$.alert("请输入账号和密码",'shanchushaixuanxiang');
return;
}
let postData = {
username: name,
password: password
};

ajaxHelper('/passport/login', 'POST',postData,
function(res) {
// 登录成功的回调
const data=res.data;
const userInfo={
userId:data.userId,
user_name:data.user_name,
real_name:data.real_name,
is_super:data.is_super
}

var jsonParam = JSON.stringify(userInfo);
window.localStorage.clear();
window.localStorage.setItem('token',data.token);
window.localStorage.setItem('userInfo', jsonParam);
asynFire.route("Login", "SaveUserAccount", jsonParam)
window.open('index.html', '_self')
},
function(xhr, status, error) {
// 登录失败的回调
alert('登录失败: ' + error);
}
);
});
})

function checkMqttMateCode() {
```

```
var matecode = asynFire.route("Common", "GetMqttMateCode");
return matecode;
}

body {
background: url("../image/loginBg.png") no-repeat;
/* background-size: 100%; */
}

.min {
position: absolute;
top: 30px;
right: 60px;
}
.login-content{
margin: auto;
display: flex;
/* height: 600px; */
}
.login-content>img{
width: 800px;
height: 714px;
}

.login-wrap {
width: 752px;
height: 714px;
background: #FFFFFF;
text-align: center
}

.logo {
width: 100%;
margin-top: 80px;
text-align: center;
}

.title {
font-size: 47px;
font-weight: 600;
line-height: 80px;
letter-spacing: 4px;
}

.login {
}

input {
width: 480px;
height: 40px;
border-radius: 8px;
border: 1px solid #CCCCCC;
outline: none;
padding: 10px 20px;
```

```
font-size: 24px;
font-weight: 400;
color: #999999;
line-height: 60px;
letter-spacing: 2px;
margin-top: 40px;
}

.button {
margin: 70px auto;
width: 522px;
height: 60px;
background: linear-gradient(90deg, #4AC5F5 0%, #2493E7 100%);
border-radius: 8px;
font-size: 24px;
font-weight: 400;
color: #FFFFFF;
line-height: 60px;
}
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8" />
<meta name="viewport" content="width=device-width, initial-scale=1.0" />
<title>人脸管理</title>
<link rel="stylesheet" href="./css/common.css" />

<link rel="stylesheet" href="css/iconfont.css" />
<link rel="stylesheet" href="./css/dietary_survey.css" />
<link rel="stylesheet" href="./css/elementui.css" />
<style>
[v-cloak] {
display: none; /* 隐藏未编译的内容 */
}
</style>
</head>
<body>
<div id="app" class="dietary_survey" v-cloak v-loading="isLoading">
<div class="header">
<div class="header_logo">
<img src="./image/svg/logo.svg" alt="" />
</div>
<div class="header_info">
<div class="header_content">
<div>运动营养咨询与指导教学平台</div>
<div>
<span class="wifi"> <i class="iconfont wangluo"></i></span>
<span>欢迎登录：<span class="username">admin</span></span>
</div>
<!-- <div style="cursor: pointer" class="logout">
<i class="iconfont tuichu1"></i>
退出
</div> -->
<div style="cursor: pointer" class="gohome">首页</div>
```

```
</div>
</div>
</div>
<div class="main dietary_survey_body">
<div class="dietary_survey_left">
<div id="identification" style="border-radius: 21px 21px 0 0">
<img src="./image/identification.png" />
<div>身份识别</div>
</div>
<div id="dishRecognition">
<img src="./image/dishRecognition.png" />
<div>菜品识别</div>
</div>
<div id="nutritionalStatistics" class="dietary_survey_checked">
<img src="./image/nutritionalStatistics.png" alt="" />
<div>营养统计</div>
</div>
<div id="nutritionalAnalysis">
<img src="./image/nutritionalAnalysis.png" />
<div>营养分析</div>
</div>
<div
id="evaluationSuggestion"
style="border-radius: 0 0 21px 21px; border-bottom: none"
>
<img src="./image/evaluationSuggestion.png" alt="" />
<div>评估建议</div>
</div>
</div>
<div class="content">
<!-- 人员信息 -->
<div class="basic-info">
<img :src="peopleInfo?.file_url" alt="" class="avatar" />
<div class="basic-content">
<div>
{{peopleInfo?.name}}<img
src="./image/svg/man.svg"
width="20"
style="vertical-align: middle; margin-left: 10px"
/>
</div>
<div class="info">
<div>身高: {{peopleInfo?.height}}cm</div>
<div>体重:{{peopleInfo?.weight}}kg</div>
<div>年龄: {{peopleInfo?.age}}岁</div>
<!-- <div>体重状态: {{peopleInfo?.weight_status}}</div>
<div>健康状态:{{peopleInfo?.health_status}}</div>
<div>运动习惯: {{peopleInfo?.sport_habit}}</div> -->
</div>
</div>
</div>
<el-table
:data="tableData"
max-height="700px"
```

```
style="
width: 100%;
margin-top: 10px;
border-radius: 20px;
color: #333333;
"
stripe
>
<el-table-column prop="index" label="序号" width="80" fixed>
</el-table-column>
<el-table-column
prop="dishes_name"
label="菜品名称"
width="200"
fixed
></el-table-column>
<el-table-column
prop="dishes_weight"
label="菜品重量(g)"
width="200"
fixed
></el-table-column>
<el-table-column
prop="energy"
label="能量(kcal)"
width="150"
fixed
></el-table-column>
<el-table-column
prop="fat"
label="脂肪(g)"
width="100"
fixed
></el-table-column>
<el-table-column
prop="protein"
label="蛋白质(g)"
width="150"
fixed
></el-table-column>
<el-table-column prop="va" label="维生素 A(ugRAE)" width="200">
</el-table-column>
<el-table-column prop="vb1" label="维生素 B1(mg)" width="200">
</el-table-column>
<el-table-column prop="vb2" label="维生素 B2(mg)" width="200">
</el-table-column>
<el-table-column prop="vc" label="维生素 C(mg)" width="200">
</el-table-column>
<el-table-column prop="ve" label="维生素 E(mg)" width="200">
</el-table-column>
<el-table-column prop="vpp" label="烟酸(mg)" width="150">
</el-table-column>
<el-table-column prop="potassium" label="钾(mg)" width="100">
</el-table-column>
```

```
<el-table-column prop="sodium" label="钠(mg)" width="100">
</el-table-column>
<el-table-column prop="calcium" label="钙(mg)" width="100">
</el-table-column>
<el-table-column prop="magnesium" label="镁(mg)" width="100">
</el-table-column>
<el-table-column prop="iron" label="铁(mg)" width="100">
</el-table-column>
<el-table-column prop="zinc" label="锌(mg)" width="100">
</el-table-column>
<el-table-column
prop="dietary_fiber"
label="膳食纤维(g)"
width="150"
>
</el-table-column>
<el-table-column prop="sfa" label="饱和脂肪酸(g)" width="200">
</el-table-column>
<el-table-column prop="mufa" label="单不饱和脂肪酸(g)" width="200">
</el-table-column>
<el-table-column prop="pufa" label="多不饱和脂肪酸(g)" width="200">
</el-table-column>
<el-table-column prop="cholesterol" label="胆固醇(mg)">
</el-table-column>
</el-table>
</div>
</div>
</div>
<script src="js/jquery-3.5.1.js"></script>
<script src="js/common.js"></script>
<script src="js/tab_left.js"></script>
<script src="js/ajax.js"></script>
<script src="./js/vue/vue.js"></script>
<script src="./js/vue/element.js"></script>
<script>
new Vue({
el: "#app",
data() {
return {
tableData: [],
peopleInfo: {},
isLoading:false,
count:0,
};
},
mounted() {
this.peopleInfo = JSON.parse(
window.localStorage.getItem("detail") || "{}"
);
this.getTableData();
},

methods: {
getTableData() {
```

```
const that=this;
that.isLoading = true;
ajaxHelper(
"/meal.meal/infos",
"POST",
{ staff_id: this.peopleInfo.id },
(res) => {
console.log(res);
if (res.data.is_loading == 1 && that.count == 0 ) {
setTimeout(() => {

that.getTableData();
}, 2000);
} else {
clearTimeout(that.count);
that.isLoading = false;
this.tableData = res.data.list.map((item, index) => ({
index: index + 1, // 添加序号
...item,
}));
that.isLoading = false;
}
}
);
},
},
});
</script>
</body>
</html>
.dietary_survey {
background: #0b508f;
}
.dietary_survey_body {
display: flex;
height: calc(100vh - 100px);
padding: 0 10px 10px 10px;
}
.dietary_survey_left {
width: 127px;
text-align: center;
background-color: #ffffff;
/* background: linear-gradient(180deg, #4ac5f5 0%, #2493e7 100%); */
border-radius: 21px 21px 21px 21px;

margin-right: 20px
}
.dietary_survey_left > div {
height: 20%;
display: flex;
flex-direction: column;
justify-content: center;
align-items: center;
font-size: 20px;
```

```
/* border-bottom: 1px solid #B4C2CB; */
}

.dietary_survey_left > div:hover {
cursor: pointer;
}
.dietary_survey_checked {
background: linear-gradient(180deg, #4ac5f5 0%, #2493e7 100%);
color: #ffffff;
}
.dietary_survey_left img {
width: 50px;
height: 50px;
margin-bottom: 10px;
}
.dietary_survey_main {
flex: 1;

background: #ffffff;
border: 1px solid #979797;
opacity: 0.84;
border-radius: 21px;
position: relative;
}
/* .dietary_survey_main::before{
content: "";
width: 10px;
height: 10px;
background: #0b508f;
position: absolute;
top: 0px;
left: 0px;
display: block;
border-radius: 0px 45px 45px 0px;
} */

.dietary_survey_main {
display: flex;
padding-top: 100px;
text-align: center;
}
.dietary_survey_main .btn {
background: linear-gradient(180deg, #4ac5f5 0%, #2493e7 100%), linear-gradient(180deg,
#10e4e5 0%, #4386fc 100%);
border-radius: 60px;
line-height: 60px;
font-weight: 600;
font-size: 38px;
color: #FFFFFF;
background: linear-gradient(180deg, #4ac5f5 0%, #2493e7 100%);
position: fixed;
bottom: 100px;
transform: translateX(-50%);
border: 1px solid #40b7f1;
```

```
}
.dietary_survey_main #confirm_btn {
position: absolute;
top: 45px;
right: 60px;
}
.dietary_survey_main .people{
width: 100%;
position: relative;
}
.dietary_survey_main .people .avatar {
width: 424px;
height: 424px;
border-radius: 50%;
border: 6px solid rgba(148, 184, 233, 1);
}
.dietary_survey_main .people .name {
font-weight: 500;
font-size: 50px;
color: #333333;
margin: 10px 0;
}
.dietary_survey_main .people .info > div {
display: inline-block;
padding: 10px 10px 0 10px;
font-size: 24px;
}
.face_text{
font-weight: 500;
font-size: 50px;
color: #333333;
line-height: 70px;
margin-top: 50px;
}

$(function () {
// $.loading(false)
// 点击左侧菜单

const detail=JSON.parse(window.localStorage.getItem('detail') || '{}');

const dishes=JSON.parse(window.localStorage.getItem('dishes') || '{}');
getDetailClick()
// 身份识别
$('#identification').click(function () {
window.open('dietary_survey.html', '_self')
window.localStorage.removeItem('detail');
window.localStorage.removeItem('dishes');
});

// 点击菜品学习
$('#dishRecognition').click(function () {
Object.keys(detail).length && window.open('dish_recognition.html', '_self')
});
```

```
// 点击营养统计
$('#nutritionalStatistics').click(function () {
Object.keys(detail).length && Object.keys(dishes).length && window.open('face_500.html',
'_self')
});
// 营养分析
$('#nutritionalAnalysis').click(function () {
Object.keys(detail).length && Object.keys(dishes).length &&
window.open('nutritional_analysis.html', '_self')
});
// 点击系统管理
$('#evaluationSuggestion').click(function () {
Object.keys(detail).length && Object.keys(dishes).length &&
window.open('evaluation_suggestion.html', '_self')
});
})
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8" />
<meta
name="viewport"
content="width=device-width, user-scalable=no, initial-scale=1.0, maximum-scale=1.0,
minimum-scale=1.0"
/>
<meta http-equiv="X-UA-Compatible" content="ie=edge" />
<title>高开仪</title>
<link rel="stylesheet" href="css/common.css" />
<link rel="stylesheet" href="css/iconfont.css" />
<link rel="stylesheet" href="css/menuStudy.css" />
</head>
<body class="body">
<div class="top-part">
<div class="logo"><img src="image/logo.png" alt="" /></div>
<div class="top-right">
<div class="wifi"><i class="iconfont wangluo"></i></div>
<div>欢迎登录：<span class="username">admin</span></div>
<div class="buttons-wrap">
<div class="back click"><i class="iconfont fanhui"></i>返回</div>
<div class="min click"><i class="iconfont zuixiaohua"></i>最小化</div>
<div class="exit click"><i class="iconfont tuichu1"></i>退出</div>
</div>
</div>
</div>
<div class="main">
<div class="main-left">
<div class="search">
<i class="iconfont sousuo" onclick="getFoodsData()"></i>
<input class="searchValue" type="text" placeholder="请输入菜品名称" />
<!-- <i class="clear iconfont shanchushaixuanxiang"></i>-->
</div>
<div class="food-class">
<!-- 食物类列表-->
<div class="class-item-wrap"></div>
```

```
<!-- 食物列表-->
<div class="food-list-wrap"></div>
</div>
</div>
<div class="main-right">
<div class="food-img-wrap">
<div
class="select-all select-all_btn click"
onclick="selectAll(this)"
>
全选
</div>
<div
class="select-none select-all click"
style="display: none"
onclick="selectNone(this)"
>
取消全选
</div>
<div class="select-all click" onclick="selectNone()">取消全选</div>
<div class="food-img"></div>
<div class="delete click" onclick="del()">删除</div>
</div>
<div class="menu-study">
<div class="camera"></div>
<div class="study-btn click">学习<br />菜品</div>
</div>
</div>
</div>

<!--删除弹框-->
<div class="delete-wrap">
<div class="delete-pop">
<div class="delete-title">是否删除？</div>
<div class="delete-btn">
<div class="delete-close click" onclick="closeDeleteWrap()">取消</div>
<div class="delete-sure click" onclick="affirmDelete()">确定</div>
</div>
</div>
</div>
<script src="js/jquery-3.5.1.js"></script>
<script src="js/common.js"></script>
<script src="js/menuStudy.js"></script>
</body>
</html>
// 食物分类的数据
var foodsClassData = [
{ name: '全部', value: '', icon: 'quanbu' },
{ name: '主食', value: 1, icon: 'zhushi' },
{name: '素菜', value: 2, icon: 'sucai'},
{name: '荤菜', value: 3, icon: 'huncai'},
{name: '蛋奶坚果', value: 4, icon: 'dannai'},
{name: '水果', value: 5, icon: 'shuiguo'},
{name: '汤饮', value: 6, icon: 'tangyin'}
```

```
]
let foodsData = [
{name: '切片面包', value: 1},
]
let foodsImgData = []

// 食物类别列表渲染
function renderFoodsClass() {
return new Promise((resolve, reject) => {
let li = ''
let len = foodsClassData.length
for (let i = 0; i < len; i++) {
li += '<div onclick="changeFoodsClass(this,\'' + foodsClassData[i].value + '\',' +
foodsClassData.length + ')" data-value="' + foodsClassData[i].value + '" class="class-item
click"><i class="iconfont ' + foodsClassData[i].icon + '"></i><span class="food-name">'
+ foodsClassData[i].name + '</span></div>'
}
$('.class-item-wrap').append(li)
resolve()
})
}

// 切换食物类
function changeFoodsClass(dom, value) {
$('.class-item').removeClass('active')
$(dom).addClass('active')
getFoodsData()
}

// 左侧食物列表渲染
function renderFoodList(data) {
let li = ''
let len = data.length
for (let i = 0; i < len; i++) {
let num = data[i].img_num > 0 ? ' ( ' + data[i].img_num + ' ) ' : "";
li += '<div onclick="changeFood(this, \'' + data[i].uuid + '\',' + data.length + ')" data-id="'
+ data[i].uuid + '" class="food-list click">' + data[i].name + num + '</div>'
}
$('.food-list-wrap').empty().append(li)
$('.food-list').eq(0).click()
}

// 切换食物
function changeFood(dom, uuid) {
$('.food-list').removeClass('active')
$(dom).addClass('active')
renderFoodImg()
getFoodsInfoData(uuid)
}

// 右侧侧食物图片渲染
function renderFoodImg(src, uuid) {
let li = ''
let len = foodsImgData.length
for (let i = 0; i < len; i++) {
```

```
li += '<div onclick="selectFoodImg(this,' + uuid + ')" data-id="' + uuid + '"
class="food-img-item click"><img src="' + src + '" alt=""><div class="img-cover"><i
class="iconfont duigou"></i></div></div>'
}
$('.food-img').empty().append(li)
}

// 选择食物图片
function selectFoodImg(dom, value) {
$(dom).toggleClass('selected')
}

$(function () {
// 关闭 loading
$.loading(false)

renderFoodsClass().then(() => {
// 默认选中第一类
$('.class-item').eq(0).click();
getFoodsData()
})


//学习特征
$(".study-btn").click(function () {
var food = $(".food-list.active");
if (food.length == 0) {
$.alert("请选择菜品");
return;
}

var id = food.attr("data-id");
var result = asynFire.route("MenuStudy", "AddFoodPicture", id);
if (result > 0) {
$.alert("学习成功");
refreshFoodsData(id);
//changeFood(food, id)
//ShowHaveStudyFood();
} else if (result == 0) {
$.alert("学习失败");
} else {
$.alert("学习失败，请调整菜品或光线后再试");
}
})

$('.back').click(function () {
asynFire.route("MenuStudy", "HideDishCamera");
})

setTimeout(() => {
ShowDishCamera($(".camera"));
}, 500)

})
```

```
// 获取食物类别下的食物数据
function getFoodsData() {
$.ajax({
url: baseUrl + 'p/api/select',
async: false,
data: {
category: $('.class-item.active').data('value'),
keyword: $('.searchValue').val()
},
type: 'post',
dataType: 'json',
success: (res) => {
renderFoodList(res.data.rows)
}
})
}

// 获取食物信息数据
function getFoodsInfoData(uuid) {
//$.ajax({
// url: baseUrl + 'p/api/find',
// data: {
// uuid: uuid
// },
// type: 'post',
// dataType: 'json',
// success: (res) => {
// console.log(res);
// renderFoodImg(res.data.file_path)
// // $('.energy').text(res.data.energy)
// // $('.carbohydrate').text(res.data.carbohydrate)
// // $('.protein').text(res.data.protein)
// // $('.fat').text(res.data.fat)
// }
//})
$(".food-img").html("");
var pictures = eval(asynFire.route("MenuLibrary", "GetFoodPicture", uuid));
for (var i = 0; i < pictures.length; i++) {
var p = $('<div onclick="selectFoodImg(this)" class="food-img-item click"><img src=""
alt=""><div class="img-cover"><i class="iconfont duigou"></i></div></div>');
p.find(".img-cover").attr("data-ID", pictures[i].ID).attr("data-FoodID",
pictures[i].FoodID).attr("data-ContSign", pictures[i].ContSign);
p.find("img").attr("src", "../../Image/Dishes/" + pictures[i].FoodID + "/" +
pictures[i].ContSign + ".jpeg");
$(".food-img").append(p);
}
}

// 点击删除
function del() {
if ($(".img-cover:visible").length == 0) {
$.alert("请选择要删除的图片");
return;
}
asynFire.route("MenuStudy", "HideDishCamera");
```

```
$('.delete-wrap').show()
}

// 关闭删除弹框
function closeDeleteWrap() {
$('.delete-wrap').hide()
ShowDishCamera($(".camera"));
}

//删除确认按钮
function affirmDelete() {
var data = new Array()
$(".img-cover:visible").each(function () {
data.push({ ID: $(this).attr("data-ID"), FoodID: $(this).attr("data-FoodID"), ContSign:
$(this).attr("data-ContSign") })
});
var result = asynFire.route("MenuLibrary", "DeleteFoodPicture", JSON.stringify(data));
if (result > 0) {
$.alert("删除成功");
var id = $(".food-list.active").attr("data-id");
refreshFoodsData(id);
// 关闭弹框
closeDeleteWrap()
ShowDishCamera($(".watch"));
} else {
$.alert("删除失败");
}

}

//全选按钮
function selectAll(dom) {
$(".food-img-item").removeClass('selected').addClass('selected')
$(dom).hide()
$('.select-none').show();
}
//取消全选按钮
function selectNone() {
$(".food-img-item").removeClass('selected').addClass('selected')
}

//取消全选按钮
function selectNone(dom) {
$(dom).hide()
$('.select-all_btn').show();
$(".food-img-item").removeClass('selected')
}

function ShowDishCamera(dom) {
var position = dom.offset();
var location = {
X: position.left,
Y: position.top,
H: dom.height(),
```

```
W: dom.width()
};
asynFire.route("Common", "ShowDishCamera", JSON.stringify(location));
}

function refreshFoodsData(uuid) {
let scroll = $(".food-list-wrap").scrollTop();
getFoodsData();
$("[data-id='" + uuid + "']").click();
$(".food-list-wrap").scrollTop(scroll);
}
.main {
padding: 40px 60px 36px;
display: flex;
align-items: center;
justify-content: space-between;
}

.main-left {
width: 500px;
}

/*搜索框*/
.search {
position: relative;
width: 100%;
}

.search input {
width: 395px;
outline: none;
height: 60px;
padding: 0 60px 0 30px;
background: #FFFFFF;
border-radius: 8px;
border: 1px solid #EEEEEE;
font-size: 24px;
font-weight: 400;
color: #999999;
line-height: 60px;
}

.search input:focus {
border: 1px solid #2493E7;
}

.search .iconfont {
position: absolute;
right: 20px;
top: 16px;
color: #2493E7;
}

.food-class {
```

```
width: 100%;
display: flex;
align-items: center;
margin-top: 18px;
}

.class-item-wrap {
width: 150px;
height: 820px;
background: linear-gradient(180deg, #4AC5F5 0%, #2493E7 100%);
border-radius: 8px;
display: flex;
justify-content: center;
align-items: center;
flex-direction: column;
}

.class-item {
width: 144px;
height: 117px;
display: flex;
flex-direction: column;
align-items: center;
justify-content: center;
font-size: 28px;
font-weight: 600;
color: #FFFFFF;
line-height: 40px;
}

.class-item.active {
width: 144px;
/*height: 134px;*/
background: #FFFFFF;
border-radius: 8px;
color: #2493E7;
}

.class-item .iconfont {
font-size: 38px;
}

.class-item.active .iconfont {
color: #2493E7;
}

.food-list-wrap {
flex: 1;
height: 820px;
background: #FFFFFF;
border-radius: 8px;
border: 1px solid #EEEEEE;
overflow: auto;
}
```

```
.food-list {
height: 88px;
padding-left: 26px;
font-size: 26px;
font-weight: 500;
color: #333333;
line-height: 80px;
letter-spacing: 2px;
}

.food-list.active {
background: rgba(22, 144, 255, 0.08);
color: #2493E7;
}

.main-right {
width: 1270px;
display: flex;
flex-direction: column;
align-items: center;
}

.food-img-wrap {
height: 240px;
padding: 0 40px;
background: #FFFFFF;
border-radius: 8px;
border: 1px solid #EEEEEE;
display: flex;
align-items: center;
margin-bottom: 30px;
}

.select-all {
width: 100px;
height: 195px;
background: #EBF5FF;
border-radius: 8px;
font-size: 28px;
font-weight: 500;
color: #2493E7;
text-align: center;
line-height: 100px;
writing-mode: vertical-lr;
}

.food-img {
width: 840px;
display: flex;
align-items: center;
flex-wrap: nowrap;
margin: 0 45px;
overflow: auto;
```

```
}

.food-img-item {
min-width: 260px;
margin-left: 30px;
width: 260px;
height: 195px;
border-radius: 8px;
overflow: hidden;
position: relative;
}

.food-img-item:first-child {
margin-left: 0;
}

.food-img-item img {
width: 100%;
height: 100%;
}

.img-cover {
width: 100%;
height: 100%;
position: absolute;
top: 0;
left: 0;
background: rgba(0, 0, 0, 0.7);
line-height: 195px;
text-align: center;
display: none;
}

.food-img-item.selected .img-cover {
display: block;
}

.img-cover .iconfont {
width: unset;
height: unset;
font-size: 60px !important;
color: #2493E7;
}

.delete {
width: 155px;
height: 195px;
background: linear-gradient(180deg, #FFA8A8 0%, #FF7878 100%);
border-radius: 8px;
font-size: 28px;
font-weight: 500;
color: #FFFFFF;
text-align: center;
line-height: 195px;
```

```
letter-spacing: 1px;
}

.delete:active {
box-shadow: 0px 0px 4px 2px #FFC9C9;
}

.menu-study {
width: 100%;
height: 634px;
display: flex;
align-items: center;
}

.camera {
width: 956px;
height: 634px;
background: #DEDEDE;
border-radius: 8px;
}

.study-btn {
width: 270px;
height: 100%;
border-radius: 8px;
font-size: 50px;
font-weight: 600;
color: #FFFFFF;
display: flex;
align-items: center;
justify-content: center;
letter-spacing: 4px;
margin-left: 40px;
background: linear-gradient(-45deg, #FF96FB, #84FFDF, #55DCFF, #BD8AFF);
background-size: 400% 400%;
animation: gradientBG 10s linear infinite;
}

@keyframes gradientBG {
0% {
background-position: 0 0%;
}
50% {
background-position: 0 100%;
}
100% {
background-position: 0 0%;
}
}

.study-btn:active {
box-shadow: 0px 0px 4px 2px #75C4FF;
}
<!DOCTYPE html>
```

```
<html lang="en">
<head>
<meta charset="UTF-8" />
<meta name="viewport" content="width=device-width, initial-scale=1.0" />
<title>菜品识别中页面</title>

<link rel="stylesheet" href="./css/iconfont.css" />
<link rel="stylesheet" href="./css/dietary_survey.css" />
<link rel="stylesheet" href="./css/menu_identification.css" />
<link rel="stylesheet" href="./css/elementui.css" />
<script src="./js/vue/vue.js"></script>
<script src="./js/vue/element.js"></script>
<link rel="stylesheet" href="./css/common.css" />
</head>
<style>
[v-cloak] {
display: none; /* 隐藏未编译的内容 */
}
.el-radio.is-bordered {
padding: 18px 20px 10px 20px;
height: 60px;
}
.el-radio__label {
font-size: 18px;
}
.el-dialog--center .el-dialog__body {
text-align: center;
}
</style>
<body>
<div id="app" class="dietary_survey" v-cloak>
<div class="header">
<div class="header_logo">
<img src="./image/svg/logo.svg" alt="" />
</div>
<div class="header_info">
<div class="header_content">
<div>运动营养咨询与指导教学平台</div>
<div>
<span class="wifi"> <i class="iconfont wangluo"></i></span>
<span>欢迎登录：<span class="username">admin</span></span>
</div>
<!-- <div style="cursor: pointer" class="logout">
<i class="iconfont tuichu1"></i>
退出
</div> -->
<div style="cursor: pointer" class="gohome">首页</div>
</div>
</div>
</div>
<div class="dietary_survey_body">
<div class="dietary_survey_left">
<div id="identification" style="border-radius: 21px 21px 0 0">
<img src="./image/identification.png" />
```

```
<div>身份识别</div>
</div>
<div id="dishRecognition" class="dietary_survey_checked">
<img src="./image/dishRecognition.png" />
<div>菜品识别</div>
</div>
<div id="nutritionalStatistics">
<img src="./image/nutritionalStatistics.png" alt="" />
<div>营养统计</div>
</div>
<div id="nutritionalAnalysis">
<img src="./image/nutritionalAnalysis.png" />
<div>营养分析</div>
</div>
<div
id="evaluationSuggestion"
style="border-radius: 0 0 21px 21px; border-bottom: none"
>
<img src="./image/evaluationSuggestion.png" alt="" />
<div>评估建议</div>
</div>
</div>
<div class="content">
<!-- 人员信息 -->
<div class="basic-info" style="justify-content: space-between">
<div style="display: flex; align-items: center">
<img :src="peopleInfo?.file_url" alt="" class="avatar" />
<div class="basic-content">
<div>
{{peopleInfo?.name}}
<img
src="./image/svg/man.svg"
width="20"
style="vertical-align: middle; margin-left: 10px"
/>
</div>
<div class="info">
<div>身高: {{peopleInfo?.height}}cm</div>
<div>体重:{{peopleInfo?.weight}}kg</div>
<div>年龄: {{peopleInfo?.age}}岁</div>
<!-- <div>体重状态: {{peopleInfo?.weight_status}}</div>
<div>健康状态:{{peopleInfo?.health_status}}</div>
<div>运动习惯: {{peopleInfo?.sport_habit}}</div> -->
</div>
</div>
</div>
<!-- :loading="saveLoading" -->
<el-button
type="primary"
round
class="date_save_btn"
@click="handleSave"
style="padding: 15px 40px; border-radius: 35px"
>保存</el-button
```

```
>
</div>
<!-- 菜品识别 -->
<div class="menu_recognition_content">
<div class="menu_identification">
<img :src="filepath" />
</div>
<!-- 操作流程 -->
<div class="menu_operation">
<div class="menu_opera_list menu_opera_list_title">
<span>序号</span>
<span>食物名称</span>
<span>食物重量(g)</span>
<span>操作</span>
</div>
<div id="menuContainer">
<div
class="menu_opera_list"
v-for="(item,index) in dishes"
:key="index"
>
<span>{{ index +1}}</span>
<el-select
v-model="item.dishes_id"
style="width: 150px; margin: 0 5px"
@change="(e)=>handleChange(e,item)"
@focus="handleInputClick"
filterable
>
<!-- <el-option-group
v-for="group in categoryList"
:key="group.label"
:label="group.label"
> -->
<el-option
v-for="item in categoryList"
:key="item.id"
:label="item.name"
:value="item.id"
>
</el-option>
<!-- </el-option-group> -->
</el-select>
<!-- <span>{{item.dishes_weight}}</span> -->
<el-input
v-model="item.dishes_weight"
style="width: 150px; margin: 0 5px"
@focus="handleInputClick"
></el-input>
<span
><el-button
class="food_delete"
@click="handleDelete(item,index)"
plain
```

```
type="danger"
round
style="
font-size: 20px;
padding: 15px 0;
border-radius: 30px;
"
>删除</el-button
></span
>
</div>
</div>
</div>
</div>
</div>
</div>

<el-dialog title="识别结果保存" :visible="saveVisible" center :show-close="false">
<el-radio v-model="saveRadio" :label="1" border
>清除历史菜单,仅保存此次结果</el-radio
>
<el-radio v-model="saveRadio" :label="2" border
>累加到历史就餐结果</el-radio
>
<span slot="footer" class="dialog-footer">
<el-button @click="saveVisible = false">取 消</el-button>
<el-button type="primary" @click="handleConfirm">确 定</el-button>
</span>
</el-dialog>
</div>
</body>
<script src="js/jquery-3.5.1.js"></script>
<script src="js/common.js"></script>

<script src="js/tab_left.js"></script>
<script src="js/ajax.js"></script>
<script>
new Vue({
el: "#app",
data() {
return {
tableData: [],
peopleInfo: {},
filepath: "",
dishes: [],
saveLoading: false,
allDishes: [],
categoryList: [],
saveVisible: false,
saveRadio: 1,
};
},
mounted() {
this.peopleInfo = JSON.parse(
```

```
window.localStorage.getItem("detail") || "{}"
);
const datainfo = JSON.parse(
window.localStorage.getItem("dishes") || "{}"
);

datainfo.dishes.map((item) => {
if (item.dishes_id == 0 && !item.dishes_name) {
item.dishes_id = null;
}
});
this.dishes = datainfo.dishes;
this.filepath = datainfo.filepath;

this.getAllDishes();
},
methods: {
handleInputClick() {
asynFire.route("Common", "StartKeyBoardFun");
},
handleChange(e, item) {
item.dishes_id = e;
},
handleConfirm() {
this.saveVisible = false;
const url =
this.saveRadio == 1 ? "/meal.meal/addMeal" : "/meal.meal/add";
this.saveLoading = true;
var data = asynFire.route(
"Common",
"GetBase64FromImage",
this.filepath
);
this.dishes.forEach((obj) => {
delete obj.dishes_name;
});
ajaxHelper("/upload/imageBase64", "POST", { img: data }, (result) => {
if (result.status == 200) {
const file_id = Number(result.data.fileInfo.file_id);
const params = {
form: {
staff_id: this.peopleInfo.id,
file_id,
dishes: this.dishes,
},
};
console.log(params, "params");
ajaxHelper(url, "POST", params, (res) => {
if (res.status == 200) {
$.alert(res.message);
} else {
$.alert(res.message, "shanchushaixuanxiang");
}
this.saveLoading = false;
```

```
});
} else {
$.alert(res.message, "shanchushaixuanxiang");
this.saveLoading = false;
}
});
},
getAllDishes() {
const that = this;
// ajaxHelper("/dishes.dishes/cate", "POST", {}, (res) => {
// const foodList = res.data.list;
// foodList.map((category) => {
// ajaxHelper(
// "/dishes.dishes/all",
// "POST",
// { category: category.id },
// (res) => {
// that.categoryList.unshift({
// label: category.name,
// options: [...res.data.list],
// });
// }
// );
// });
// console.log(that.categoryList, "----");
// });

ajaxHelper("/dishes.dishes/all", "POST", {}, (res) => {
that.categoryList = res.data.list;
});
},
handleSave() {
this.saveVisible = true;
console.log(this.saveVisible, "----");
// this.saveLoading = true;
// var data = asynFire.route(
// "Common",
// "GetBase64FromImage",
// this.filepath
// );
// this.dishes.forEach(obj=>{
// delete obj.dishes_name;
// })
// ajaxHelper("/upload/imageBase64", "POST", { img: data }, (result) => {
// if (result.status == 200) {
// const file_id = Number(result.data.fileInfo.file_id);
// const params = {
// form: {
// staff_id: this.peopleInfo.id,
// file_id,
// dishes: this.dishes,
// },
// };
// console.log(params, "params");
```

```
// ajaxHelper("/meal.meal/add", "POST", params, (res) => {
// if (res.status == 200) {
// $.alert(res.message);
// } else {
// $.alert(res.message, "shanchushaixuanxiang");
// }
// this.saveLoading = false;
// });
// } else {
// $.alert(res.message, "shanchushaixuanxiang");
// this.saveLoading = false;
// }
// });
},
handleDelete(data,index) {
this.dishes.splice(index,1)
},
},
});
</script>
<script>
// 获取 base64
function showFoodPicture(base64) {
// asynFire.route("Face", "HideFaceCamera")
$(".menuCameraBg").attr("src", "" + base64);
setTimeout(() => {
asynFire.route("MenuStudy", "HideDishCamera");
}, 200);
}
$(document).ready(function () {
const datainfo = JSON.parse(
window.localStorage.getItem("dishes") || "{}"
);
let dishes = datainfo.dishes;

// 遍历数据并动态生成 HTML
// $.each(dishes, function (index, item) {
// let foodRow = `
// <div class="menu_opera_list">
// <span>${index + 1}</span>
// <span>${item.dishes_name}</span>
// <span>${item.dishes_weight}</span>
// <span><button class="food_delete" data-id="${
// item.dishes_id
// }">删除</button></span>
// </div>
// `;
// $("#menuContainer").append(foodRow);
// });

// 删除按钮的点击事件
// $(document).on("click", ".food_delete", function () {
// let id = $(this).data("dishes_id");
// $(this).closest(".menu_opera_list").remove();
```

```
// dishes = dishes.filter((item) => id !== item.dishes_id);
// console.log("删除食物 ID:", dishes);
// });
// $(document).on("click", ".dish_save_btn", function () {
// // let dishImgSrc = $("#dish_img").attr("src");
// // const img = $(".menuCameraBg").attr("src");
// // const formData = new FormData();
// // formData.append('img', img);
// let form = {
// staff_id: 1,
// file_id: 1,
// dishes,
// };
// const ContentType = "application/json";
// ajaxHelper(
// "/meal.meal/add",
// "POST",
// { form },
// (result) => {
// $.alert(result.message);
// },
// (error) => {
// console.error("error", error);
// }
// );
// });
});
</script>
</html>
.menu_recognition_content {
margin-top: 10px;
display: flex;
justify-content: space-between;
margin-right: 10px;
}
.menu_identification {
width: 60%;
min-height: 700px;
background: #4c4c4c;
border-radius: 8px;
text-align: center;
margin-right: 20px;
}
.menu_identification > img {
object-fit: scale-down;
width: 100%;
}

.menu_operation {
flex: 1;
background-color: #ffffff;
padding: 0 10px 10px 0px;
}
```

```
.menu_opera_list {
display: flex;
line-height: 44px;
padding: 10px 0;
font-size: 20px
}
.menu_opera_list > span {
flex: 1;
color: #333333;
text-align: center;
}

.menu_opera_list_title {
font-weight: bold;
}

.food_delete {
border-radius: 35px;
border: 1px solid #ed3c25;
width: 80%;
line-height: 30px;
background-color: #ffffff;
color: #ed3c25;
padding: 10px 0;
}
.food_delete:hover{
box-shadow: 0px 0px 5px 0px #ed3c25;
}
.dish_save_btn{
width: 100px;
height: 40px;
background: linear-gradient( 180deg, #4AC5F5 0%, #2493E7 100%);
box-shadow: 0px 0px 10px 0px #42BAF2;
border-radius: 35px;
border: 1px solid #40B7F1;
color: #ffffff;

}
```

# 后端代码

```
<?php
declare (strict_types=1);
namespace app\admin\controller\admin;
use think\response\Json;
use app\admin\controller\Controller;
use app\admin\model\admin\User as AdminUserModel;
class User extends Controller
{
    protected array $methodRules = [
        'detail' => 'GET',
        'renew' => 'POST',
    ];
    public function detail(): Json
    {
        $userInfo = AdminUserModel::detail($this->admin['user']['admin_user_id']);
        return $this->renderSuccess(['userInfo' => $userInfo]);
    }
    public function renew(): Json
    {
        $model = AdminUserModel::detail($this->admin['user']['admin_user_id']);
        if ($model->renew($this->postForm())) {
            return $this->renderSuccess('更新成功');
        }
        return $this->renderError($model->getError() ?: '更新失败');
    }
}
<?php
declare (strict_types=1);
namespace app\admin\controller\setting;
use think\response\Json;
use app\admin\controller\Controller;
use app\admin\service\Cache as CacheService;
class Cache extends Controller
{
    public function clear(): Json
    {
        $CacheService = new CacheService;
        if (!$CacheService->rmCache($this->postForm())) {
            return $this->renderError($CacheService->getError() ?: '操作失败');
        }
        return $this->renderSuccess('操作成功');
    }
}
<?php
declare (strict_types=1);
namespace app\admin\controller\setting;
use think\response\Json;
use app\common\library\helper;
use app\admin\controller\Controller;
class Science extends Controller
```

```
{
    public function info(): Json
    {
        return $this->renderSuccess([
            'scienceInfo' => [
                'server' => $this->server(),
                'phpinfo' => $this->phpinfo(),
                'writeable' => $this->writeable(),
            ]
        ]);
    }
    private function server(): array
    {
        return [
            [
                'name' => '服务器操作系统',
                'key' => 'system',
                'value' => PHP_OS,
                'status' => PHP_SHLIB_SUFFIX === 'dll' ? 'warning' : 'normal',
                'remark' => '建议使用 Linux 系统以提升程序性能'
            ],
            [
                'name' => 'Web 服务器环境',
                'key' => 'webserver',
                'value' => $this->request->server('SERVER_SOFTWARE'),
                'status' => PHP_SAPI === 'isapi' ? 'warning' : 'normal',
                'remark' => '建议使用 Apache 或 Nginx 以提升程序性能'
            ],
            [
                'name' => 'PHP 版本',
                'key' => 'php',
                'value' => PHP_VERSION,
                'status' => version_compare(PHP_VERSION, '7.4.0') === -1 ? 'danger' : 'normal',
                'remark' => 'PHP 版本必须为 7.4.0 及以上'
            ],
            [
                'name' => 'PHP 运行位数',
                'key' => 'architecture',
                'value' => (PHP_INT_SIZE === 4 ? '32' : '64') . '位',
                'status' => PHP_INT_SIZE === 4 ? 'warning' : 'normal',
                'remark' => '建议使用 64 位 PHP 以提升程序性能'
            ],
            [
                'name' => '文件上传最大值',
                'key' => 'upload_max_filesize',
                'value' => ini_get('upload_max_filesize'),
                'status' => $this->compareBytes(ini_get('upload_max_filesize'), '30m') ?
'danger' : 'normal',
                'remark' => '不能小于 30MB；请修改 php.ini 中 upload_max_filesize'
            ],
            [
                'name' => 'POST 数据最大值',
                'key' => 'post_max_size',
                'value' => ini_get('post_max_size'),
```

```
            'status' => $this->compareBytes(ini_get('post_max_size'), '30m') ? 'danger' :
'normal',
            'remark' => '不能小于 30MB，请修改 php.ini 中 post_max_size'
        ],
        [
            'name' => '程序运行目录',
            'key' => 'root_path',
            'value' => str_replace('\\', '/', root_path()),
            'status' => 'normal',
            'remark' => ''
        ],
    ];
  }
  private function phpinfo(): array
  {
      return [
        [
            'name' => 'PHP 版本',
            'key' => 'php_version',
            'value' => '7.4.0',
            'status' => version_compare(PHP_VERSION, '7.4.0') === -1 ? 'danger' : 'normal',
            'remark' => 'PHP 版本必须为 7.4.0'
        ],
        [
            'name' => 'Mysqlnd',
            'key' => 'mysqlnd',
            'value' => '支持',
            'status' => extension_loaded('mysqlnd') ? 'normal' : 'danger',
            'remark' => '您的 PHP 环境不支持 mysqlnd, 系统无法正常运行'
        ],
        [
            'name' => 'ZIP',
            'key' => 'zip',
            'value' => '支持',
            'status' => extension_loaded('zip') ? 'normal' : 'warning',
            'remark' => '您的 PHP 环境不支持 zip, 系统无法使用 zip 压缩文件'
        ],
        [
            'name' => 'CURL',
            'key' => 'curl',
            'value' => '支持',
            'status' => extension_loaded('curl') && function_exists('curl_init') ? 'normal' :
'danger',
            'remark' => '您的 PHP 环境不支持 CURL, 系统无法正常运行'
        ],
        [
            'name' => 'JSON',
            'key' => 'json',
            'value' => '支持',
            'status' => extension_loaded('json') ? 'normal' : 'danger',
            'remark' => '您的 PHP 环境不支持 JSON, 系统无法正常运行'
        ],
        [
            'name' => 'Fileinfo',
```

```
            'key' => 'fileinfo',
            'value' => '支持',
            'status' => extension_loaded('fileinfo') ? 'normal' : 'danger',
            'remark' => '您的 PHP 环境不支持 fileinfo, 系统无法上传文件'
        ],
        [
            'name' => 'OpenSSL',
            'key' => 'openssl',
            'value' => '支持',
            'status' => extension_loaded('openssl') ? 'normal' : 'danger',
            'remark' => '没有启用 OpenSSL, 将无法访问微信平台的接口'
        ],
        [
            'name' => 'PDO',
            'key' => 'pdo',
            'value' => '支持',
            'status' => extension_loaded('PDO') && extension_loaded('pdo_mysql') ?
'normal' : 'danger',
            'remark' => '您的 PHP 环境不支持 PDO, 系统无法正常运行'
        ],
        [
            'name' => 'GD',
            'key' => 'gd',
            'value' => '支持',
            'status' => extension_loaded('gd') ? 'normal' : 'danger',
            'remark' => '您的 PHP 环境不支持 GD, 系统无法正常生成图片'
        ],
        [
            'name' => 'BCMath',
            'key' => 'bcmath',
            'value' => '支持',
            'status' => extension_loaded('bcmath') ? 'normal' : 'danger',
            'remark' => '您的 PHP 环境不支持 BCMath, 系统无法正常运行'
        ],
        [
            'name' => 'Mbstring',
            'key' => 'mbstring',
            'value' => '支持',
            'status' => extension_loaded('mbstring') ? 'normal' : 'danger',
            'remark' => '您的 PHP 环境不支持 mbstring, 系统无法正常运行'
        ],
        [
            'name' => 'SimpleXML',
            'key' => 'simplexML',
            'value' => '支持',
            'status' => extension_loaded('SimpleXML') ? 'normal' : 'danger',
            'remark' => '您的 PHP 环境不支持 SimpleXML, 系统无法解析 xml 无法使用微信支付'
        ],
    ];
}
private function writeable(): array
{
    $paths = [
        'data' => realpath(data_path()) . '/',
```

```
            'uploads' => realpath(web_path()) . '/uploads/',
            'downloads' => realpath(web_path()) . '/downloads/',
            'temp' => realpath(web_path()) . '/temp/',
        ];
        return [
            [
                'name' => '系统数据目录',
                'key' => 'data',
                'value' => str_replace('\\', '/', $paths['data']),
                'status' => helper::checkWriteable($paths['data']) ? 'normal' : 'danger',
                'remark' => '目录不可写，系统将无法正常写入文件'
            ],
            [
                'name' => '文件上传目录',
                'key' => 'uploads',
                'value' => str_replace('\\', '/', $paths['uploads']),
                'status' => helper::checkWriteable($paths['uploads']) ? 'normal' : 'danger',
                'remark' => '目录不可写，系统将无法正常上传文件'
            ],
            [
                'name' => '文件下载目录',
                'key' => 'downloads',
                'value' => str_replace('\\', '/', $paths['downloads']),
                'status' => helper::checkWriteable($paths['downloads']) ? 'normal' : 'danger',
                'remark' => '目录不可写，系统将无法正常上传文件'
            ],
            [
                'name' => '临时文件目录',
                'key' => 'temp',
                'value' => str_replace('\\', '/', $paths['temp']),
                'status' => helper::checkWriteable($paths['temp']) ? 'normal' : 'danger',
                'remark' => '目录不可写，系统将无法正常写入文件'
            ],
        ];
    }
    private function compareBytes(string $size1, string $size2): bool
    {
        $size1 = helper::convertToBytes($size1);
        $size2 = helper::convertToBytes($size2);
        return $size1 < $size2;
    }
}
<?php
declare (strict_types=1);
namespace app\admin\controller\setting;

use think\response\Json;
use app\admin\controller\Controller;
use app\common\service\system\Process as SystemProcessService;
class Timer extends Controller
{
    public function test(): Json
    {
        sleep(3);
```

```
        if (\time() - SystemProcessService::getLastWorkingTime('timer') <= 10) {
            return $this->renderSuccess('恭喜您，定时任务已开启');
        }
        return $this->renderError('很抱歉，定时任务未开启');
    }
}
<?php
declare (strict_types=1);
namespace app\admin\controller\store;
use think\response\Json;
use app\admin\controller\Controller;
use app\admin\model\store\Api as ApiModel;
class Api extends Controller
{
    public function index(): Json
    {
        $model = new ApiModel;
        $list = $model->getList();
        return $this->renderSuccess(compact('list'));
    }
    public function add(): Json
    {
        $model = new ApiModel;
        if ($model->add($this->postForm())) {
            return $this->renderSuccess('添加成功');
        }
        return $this->renderError($model->getError() ?: '添加失败');
    }
    public function edit(int $apiId): Json
    {
        $model = ApiModel::detail($apiId);
        if ($model->edit($this->postForm())) {
            return $this->renderSuccess('更新成功');
        }
        return $this->renderError($model->getError() ?: '更新失败');
    }
    public function delete(int $apiId): Json
    {
        $model = ApiModel::detail($apiId);
        if (!$model->remove()) {
            return $this->renderError($model->getError() ?: '删除失败');
        }
        return $this->renderSuccess('删除成功');
    }
}
<?php
declare (strict_types=1);
namespace app\admin\controller\store;
use app\common\enum\equipment\EquipmentType;
use think\response\Json;
use app\admin\controller\Controller;
use app\admin\service\store\Equipment as EquipmentService;
class Equipment extends Controller
{
```

```
public function add(): Json
{
    $server = new EquipmentService();
    if ($server->add($this->request->param())) {
        $id  = $server->id;
        return $this->renderSuccess(compact('id'), '添加成功');
    }
    return $this->renderError($server->getError() ?: '添加失败');
}
public function detail(int $id): Json
{
    $server = new EquipmentService();
    $detail = $server->detail($id);
    return $this->renderSuccess(compact('detail'));
}
public function lists()
{
    $server = new EquipmentService;
    $list = $server->getList($this->request->param());
    return $this->renderSuccess(compact('list'));

public function getCode(int $id): Json
{
    $server = new EquipmentService();
    $code = $server->getCode($id);
    return $this->renderSuccess($code);
}
public function delete(int $id): Json
{
    $server = new EquipmentService;
    if ($server->setDelete($id)) {
        return $this->renderSuccess('删除成功');
    }
    return $this->renderError($server->getError() ?: '删除成功');
}
public function edit()
{
    $server = new EquipmentService;
    if ($server->edit($this->request->param())) {
        return $this->renderSuccess('更新成功');
    }
    return $this->renderError($server->getError() ?: '更新失败');
}
public function type()
{
    $detail = EquipmentType::data();
    return $this->renderSuccess(array_values($detail));
}
public function getStoreList()
{
    $server = new EquipmentService;
    $lists = $server->getStoreList();
    return $this->renderSuccess($lists);
}
```

```
}
<?php
declare (strict_types=1);
namespace app\admin\controller\store;
use think\response\Json;
use app\admin\controller\Controller;
use app\admin\model\store\Menu as MenuModel;
class Menu extends Controller
{
    public function index(): Json
    {
        $model = new MenuModel;
        $list = $model->getList();
        return $this->renderSuccess(compact('list'));
    }
    public function info(int $menuId): Json
    {
        $model = MenuModel::detail($menuId);
        return $this->renderSuccess(['info' => $model]);
    }
    public function add(): Json
    {
        $model = new MenuModel;
        if ($model->add($this->postForm())) {
            return $this->renderSuccess('添加成功');
        }
        return $this->renderError($model->getError() ?: '添加失败');
    }
    public function edit(int $menuId): Json
    {
        $model = MenuModel::detail($menuId);
        if ($model->edit($this->postForm())) {
            return $this->renderSuccess('更新成功');
        }
        return $this->renderError($model->getError() ?: '更新失败');
    }
    public function setApis(int $menuId): Json
    {
        $model = MenuModel::detail($menuId);
        if ($model->setApis($this->postForm())) {
            return $this->renderSuccess('操作成功');
        }
        return $this->renderError($model->getError() ?: '操作失败');
    }
    public function delete(int $menuId): Json
    {
        $model = MenuModel::detail($menuId);
        if (!$model->remove()) {
            return $this->renderError($model->getError() ?: '删除失败');
        }
        return $this->renderSuccess('删除成功');
    }
}
<?php
```

```php
declare (strict_types=1);
namespace app\admin\controller;
use cores\BaseController;
use app\admin\service\admin\User as AdminUserService;
use cores\exception\BaseException;
class Controller extends BaseController
    protected $admin;
    protected string $controller = '';
    protected string $action = '';
    protected string $routeUri = '';
    protected string $group = '';
    protected array $allowAllAction = [
        'passport/login',
    ];
    protected array $methodRules = [];
    public function initialize()
    {
        $this->setAdminInfo();
        $this->getRouteinfo();
        $this->checkLogin();
        $this->checkMethodRules();
    }
    private function setAdminInfo()
    {
        $this->admin = AdminUserService::getLoginInfo();
    }
    protected function getRouteinfo()
    {
        $this->controller = uncamelize($this->request->controller());
        $this->action = $this->request->action();
        $groupstr = strstr($this->controller, '.', true);
        $this->group = $groupstr !== false ? $groupstr : $this->controller;
        $this->routeUri = "{$this->controller}/$this->action";
    }
    private function checkLogin(): void
    {
        if (in_array($this->routeUri, $this->allowAllAction)) {
            return;
        }
        if (empty($this->admin) || (int)$this->admin['is_login'] !== 1) {
            throwError('请先登录后再访问', config('status.not_logged'));
        }
    }
    private function checkMethodRules(): void
    {
        if (!isset($this->methodRules[$this->action])) {
            return;
        }
        $methodRule = $this->methodRules[$this->action];
        $currentMethod = $this->request->method();
        if (empty($methodRule)) {
            return;
        }
        if (is_array($methodRule) && in_array($currentMethod, $methodRule)) {
```

```
            return;
        }
        if (is_string($methodRule) && $methodRule == $currentMethod) {
            return;
        }
        throwError('illegal request method');
    }
}
<?php
declare (strict_types=1);
namespace app\admin\controller;
use think\response\Json;
use app\admin\service\admin\User as AdminUserService;
use app\admin\model\admin\User as UserModel;
class Passport extends Controller
{
    protected array $methodRules = [
        'login' => 'POST',
    ];
    public function login(): Json
    {
        $model = new UserModel;
        if (($userInfo = $model->login($this->postData())) === false) {
            return $this->renderError($model->getError() ?: '登录失败');
        }
        return $this->renderSuccess([
            'userId' => $userInfo['admin_user_id'],
            'token' => $model->getToken()
        ], '登录成功');
    }
    public function logout(): Json
    {
        AdminUserService::logout();
        return $this->renderSuccess('操作成功');
    }
}
<?php
declare (strict_types=1);
namespace app\admin\controller;
use think\response\Json;
use app\admin\model\Store as StoreModel;
use app\admin\service\Store as StoreService;
class Store extends Controller
{
    protected array $methodRules = [
        'index' => 'GET',
        'recycle' => 'GET',
        'add' => 'POST',
        'move' => 'POST',
        'delete' => 'POST',
    ];
    public function index(): Json
    {
        $model = new StoreModel;
```

```
        $list = $model->getList();
        return $this->renderSuccess(compact('list'));
    }
    public function add(): Json
    {
        $service = new StoreService;
        if ($service->add($this->postForm())) {
            return $this->renderSuccess('添加成功');
        }
        return $this->renderError($service->getError() ?: '添加失败');
    }
    public function recycle(): Json
    {
        $model = new StoreModel;
        $list = $model->getList(true);
        return $this->renderSuccess(compact('list'));
    }
    public function recovery(int $storeId): Json
    {
        $model = StoreModel::detail($storeId);
        if (!$model->recycle()) {
            return $this->renderError($model->getError() ?: '操作失败');
        }
        return $this->renderSuccess('操作成功');
    }
    public function move(int $storeId): Json
    {
        $model = StoreModel::detail($storeId);
        if (!$model->recycle(false)) {
            return $this->renderError($model->getError() ?: '操作失败');
        }
        return $this->renderSuccess('操作成功');
    }
}
<?php
declare (strict_types=1);
namespace app\admin\model\admin;
use app\common\model\admin\User as UserModel;
use app\admin\service\admin\User as AdminUserService;
class User extends UserModel
{
    protected $hidden = [
        'password',
    ];
    private string $token;

    public function login(array $data)
    {
        if (!$user = $this->getUserInfoByLogin($data)) {
            $this->error = '登录失败, 用户名或密码错误';
            return false;
        }
        $this->token = AdminUserService::login($user->toArray());
        return $user;
```

```
    }
    public function getToken(): string
    {
        return $this->token;
    }
    private function getUserInfoByLogin(array $data)
    {
        $useInfo = static::withoutGlobalScope()
            ->where(['user_name' => $data['username']])
            ->find();
        if (empty($useInfo)) return false;
        if (!password_verify($data['password'], $useInfo['password'])) {
            return false;
        }
        return $useInfo;
    }
    public function renew(array $data): bool
    {
        if ($data['password'] !== $data['password_confirm']) {
            $this->error = '确认密码不正确';
            return false;
        }
        if ($this->save([
                'user_name' => $data['user_name'],
                'password' => encryption_hash($data['password']),
            ]) === false) {
            return false;
        }
        AdminUserService::update($this->toArray());
        return true;
    }
}
<?php
declare (strict_types=1);
namespace app\admin\model\store;
use app\admin\model\store\MenuApi as MenuApiModel;
use app\common\model\store\Api as ApiModel;
class Api extends ApiModel
{
    public function getList(): array
    {
        $all = static::getAll();
        return $this->getTreeData($all);
    }
    public function add(array $data): bool
    {
        return $this->allowField(['name', 'parent_id', 'url', 'sort'])->save($data);
    }
    public function edit(array $data): bool
    {
        if ($data['parent_id'] > 0) {
            $parentIds = $this->getTopApiIds($data['parent_id']);
            if (in_array($this['api_id'], $parentIds)) {
                $this->error = '上级权限不允许设置为当前子权限';
```

```
                return false;
            }
        }
        return $this->allowField(['name', 'parent_id', 'url', 'sort'])->save($data);
    }
    public function remove(): bool
    {
            if ($item['parent_id'] == $parentId) {
                $children = $this->getTreeData($list, (int)$item['api_id']);
                !empty($children) && $item['children'] = $children;
                $data[] = $item;
                unset($list[$key]);
            }
        }
        return $data;
    }
}
<?php
declare (strict_types=1);
namespace app\admin\model\store;
use app\common\model\store\Menu as MenuModel;
use app\admin\model\store\MenuApi as MenuApiModel;

class Menu extends MenuModel
{
    public function getList(): array
    {
        return $this->getTreeData(static::getAll());
    }
    public function add(array $data): bool
    {
        return $this->save($data);
    }
    public function edit(array $data): bool
    {
        if (isset($data['parent_id']) && $data['parent_id'] > 0) {
            $parentIds = $this->getTopMenuIds($data['parent_id']);
            if (in_array($this['menu_id'], $parentIds)) {
                $this->error = '上级菜单不允许设置为当前子菜单';
                return false;
            }
        }
        return $this->save($data);
    }
    public function setApis(array $data): bool
    {
        if (empty($data['apiIds'])) {
            $this->error = 'API 权限不能为空';
            return false;
        }
        return (new MenuApiModel)->updateByMenuId($this['menu_id'], $data['apiIds']);
    }
    public function remove(): bool
    {
```

```
        if (self::detail(['parent_id' => $this['menu_id']])) {
            $this->error = '当前菜单下存在子菜单或操作，请先删除';
            return false;
        }
        MenuApiModel::deleteAll(['menu_id' => $this['menu_id']]);
        return $this->delete();
    }
    private function getTopMenuIds(int $menuId, $menuList = null): array
    {
        static $ids = [];
        is_null($menuList) && $menuList = $this->getAll();
        foreach ($menuList as $item) {
            if ($item['menu_id'] == $menuId && $item['parent_id'] > 0) {
                $ids[] = $item['parent_id'];
                $this->getTopMenuIds($item['parent_id'], $menuList);
            }
        }
        return $ids;
    }
}
<?php
declare (strict_types=1);
namespace app\admin\model\store;

use app\common\model\store\MenuApi as MenuApiModel;
class MenuApi extends MenuApiModel
{
    public function updateByMenuId(int $menuId, array $apiIds): bool
    {
        return $this->transaction(function () use ($menuId, $apiIds) {
            $this->removeByMenuId($menuId);
            return $this->insertByMenuId($menuId, $apiIds);
        });
    }
    private function removeByMenuId(int $menuId): void
    {
        static::deleteAll(['menu_id' => $menuId]);
    }
    private function insertByMenuId(int $menuId, array $apiIds): bool
    {
        $data = [];
        foreach ($apiIds as $api) {
            $data[] = ['menu_id' => $menuId, 'api_id' => $api];
        }
        return (bool)$this->addAll($data);
    }
}
<?php
declare (strict_types=1)
namespace app\admin\model\store;
use app\common\model\store\User as StoreUserModel;
class User extends StoreUserModel
{
    public function add(int $storeId, array $data): bool
```

```
    {
        return $this->save([
            'user_name' => $data['user_name'],
            'password' => encryption_hash($data['password']),
            'is_super' => 1,
            'store_id' => $storeId,
        ]);
    }
    public function login(int $storeId)
    {
        $userInfo = $this->getSuperStoreUser($storeId);
        if (empty($userInfo)) {
            throwError('超级管理员用户信息不存在');
        }
    }
    private function getSuperStoreUser(int $storeId): ?User
    {
        return static::detail(['store_id' => $storeId, 'is_super' => 1], ['wxapp']);
    }
    public static function setDelete(int $storeId): bool
    {
        static::update(['is_delete' => '1'], ['store_id' => $storeId]);
        return true;
    }
}
<?php
declare (strict_types=1);
namespace app\admin\model;
use app\common\model\Help as HelpModel;
class Help extends HelpModel
{
}
<?php
declare (strict_types=1);
namespace app\admin\model;
use app\common\library\helper;
use app\common\model\store\Setting as SettingModel;
class Setting extends SettingModel
{
    public function insertDefault(int $storeId): bool
    {
        $data = [];
        foreach ($this->defaultData() as $key => $item) {
            $item['values'] = helper::jsonEncode($item['values']);
            $item['store_id'] = $storeId;
            $data[] = $item;
        }
        return $this->addAll($data) !== false;
    }
}
<?php
declare (strict_types=1);
namespace app\admin\model;
use app\common\model\Store as StoreModel;
```

```
use app\common\model\store\User as StoreUserModel;
class Store extends StoreModel
{
    public function getList(bool $isRecycle = false): \think\Paginator
    {
        return $this->where('is_recycle', '=', (int)$isRecycle)
            ->where('is_delete', '=', 0)
            ->order(['sort' => 'asc', 'create_time' => 'desc'])
            ->paginate(15);
    }
    public function recycle(bool $isRecycle = true): bool
    {
        return $this->save(['is_recycle' => (int)$isRecycle]);
    }
    public function addStore(array $storeData,array $storeUserData): bool
    {
        $this->transaction(function () use ($storeData, $storeUserData) {
            $this->save($storeData);
            $storeUserData['store_id'] = $this['store_id'];
            (new StoreUserModel)->save($storeUserData);
        });
        return true;
    }
    public static function checkExist(string $storeName): bool
    {
        return (bool)static::withoutGlobalScope()
            ->where('store_name', '=', $storeName)
            ->where('is_delete', '=', 0)
            ->value('store_id');
    }
}
<?php
declare (strict_types=1);
namespace app\admin\model;
use app\common\model\UploadFile as UploadFileModel;
class UploadFile extends UploadFileModel
{
}
<?php
declare (strict_types=1);
namespace app\admin\service\admin;
use think\facade\Cache;
use app\common\service\BaseService;
class User extends BaseService
{
    const TOKEN_SALT = '_admin_user_salt_';
    public static function getLoginInfo()
    {
        if (($token = self::getToken()) !== false) {
            return Cache::get($token);
        }
        return false;
    }
    public static function login(array $userInfo): string
```

```
    {
        $token = self::makeToken((int)$userInfo['admin_user_id']);
        Cache::set($token, [
            'user' => [
                'admin_user_id' => (int)$userInfo['admin_user_id'],
                'user_name' => $userInfo['user_name'],
            ],
            'is_login' => true,
        ], 86400 * 7);
        return $token;
    }
    public static function logout(): bool
    {
        return Cache::delete(self::getToken());
    }
    public static function update(array $userInfo): bool
    {
        return Cache::set(self::getToken(), [
            'user' => [
                'admin_user_id' => $userInfo['admin_user_id'],
                'user_name' => $userInfo['user_name'],
            ],
            'is_login' => true,
        ], 86400 * 7);
    }
    private static function makeToken(int $userId): string
    {
        $guid = get_guid_v4();
        $timeStamp = microtime(true);
        $salt = self::TOKEN_SALT;
        return md5("{$timeStamp}_{$userId}_{$guid}_{$salt}");
    }
    private static function getToken()
    {
        $token = request()->header('Access-Token');
        if (empty($token) && is_debug()) {
            $token = request()->param('Access-Token');
        }
        if (empty($token)) {
            return false;
        }
        return $token;
    }
}
<?ph
declare (strict_types=1);
namespace app\admin\service\store;
use app\admin\model\Store;
use app\admin\service\admin\User as AdminUserService;
use app\common\enum\equipment\EquipmentType;
use app\common\service\BaseService;
use app\common\model\Equipment as EquipmentModel;
use app\admin\validate\Equipment as EquipmentValidate;
class Equipment extends BaseService
```

```
{
    public $id;
    public function add(array $param)
    {
        $this->validate($param);
        if (EquipmentModel::checkExist('name',$param['name'])) {
            $this->error = '设备名称已存在';
            return false;
        }
        if (EquipmentModel::checkExist('code',$param['code'])) {
            $this->error = '设备号已存在';
            return false;
        }
        $loginInfo = AdminUserService::getLoginInfo();
        $param['create_by'] = $loginInfo['user']['user_name'];
        $model = new EquipmentModel();
        $res = $model->add($param);
        $this->id = $model['id'] ?: 0;
        return $res;
    }
    private function validate(array $data): void
    {
        $validate = new EquipmentValidate();
        if (!$validate->check($data)) {
            throwError($validate->getError());
        }
    }
    public function detail(int $id): array
    {
        $infos = EquipmentModel::detail($id);
        if (empty($infos)){
            return [];
        }
        $type = EquipmentType::data();
        $store_all = Store::getAllData(['is_delete' => 0],'store_id,store_name');
        $store_infos = array_column($store_all,'store_name','store_id');
        $infos = $infos->toArray();
        $infos['type_name'] = $type[$infos['type']]['name'] ?? '';
        $infos['store_name'] = $store_infos[$infos['store_id']] ?? '';
        return $infos;
    }
    public function getList($param): array
    {
        $where = [['is_delete','=',0]];
        if (!empty($param['type'])){
            $where[] = ['type','=',$param['type']];
        }
        $type = EquipmentType::data();
        $lists = EquipmentModel::getList($where);
        $lists['data'] = array_map(function ($item) use ($type) {
            $item['type_name'] = $type[$item['type']]['name'] ?? '';
            return $item;
        },$lists['data']);
        return $lists;
```

```
    }
    public function getCode($id): array
    {
        $matecode = '';
        $flag = true;
        while($flag) {
            $matecode = $this->random(6, 3);
            $matecode_exist = EquipmentModel::checkExist('matecode',$matecode);
            if(!$matecode_exist) {
                $flag = false;
            }
        }
        $update = EquipmentModel::update(['matecode' => $matecode, 'mate_flag' =>
0],['id' => $id]);
        if ($update === false){
            throwError('设备编号生成失败');
        }
        return ['code' => $matecode];
    }
    public function random(int $length, int $type = 1, string $chars =
'0123456789abcdefghijklmnopqrstuvwxyz') {
        if($type == 1) {
            $hash = sprintf('%0'.$length.'d', mt_rand(0, pow(10, $length) - 1));
        } else {
            $hash = '';
            if($type == 3) $chars .= 'ABCDEFGHIJKLMNOPQRSTUVWXYZ';
            $max = strlen($chars) - 1;
            for($i = 0; $i < $length; $i++) $hash .= $chars[mt_rand(0, $max)];
        }
        return $hash;
    }
    public function setDelete($id): bool
    {
        $del = EquipmentModel::setDelete($id);
        if ($del === false){
            throwError('设备删除失败');
        }
        return true;
    }
    public function edit(array $param): bool
    {
        $detail = EquipmentModel::detail($param['id']);
        if (empty($detail)){
            throwError('设备不存在');
        }
        if
(EquipmentModel::checkExist('name',$param['name'],[['is_delete','=',0],['id','<>',$param['id
']]])) {
            $this->error = '设备名称已存在';
            return false;
        }
        if
(EquipmentModel::checkExist('code',$param['code'],[['is_delete','=',0],['id','<>',$param['id']]
])) {
            $this->error = '设备号已存在';
```

```
            return false;
        }
        $id = $param['id'];
        unset($param['id']);
        $update = EquipmentModel::update($param,['id' => $id]);
        if ($update === false){
            throwError('设备编辑失败');
        }
        return true;
    }
    public function getStoreList()
    {
        $lists = Store::getAllData(['is_delete' => 0],'store_id,store_name');
        return $lists;
    }
}
<?php
declare (strict_types=1);
namespace app\admin\service\store;
use app\common\service\store\User as Basics;
class User extends Basics
{
}
<?php
declare (strict_types=1);
namespace app\admin\service;
use app\common\service\BaseService;
use think\facade\Cache as CacheDriver;
class Cache extends BaseService
{
    private $cache;
    public function initialize()
    {
        $this->cache = CacheDriver::instance();
    }
    public function rmCache($data): bool
    {
        if (in_array('data', $data['item'])) {
            $isForce = isset($data['isForce']) && (bool)$data['isForce'];
            $isForce ? $this->cache->clear() : $this->cache->tag('cache')->clear();
        }
        if (in_array('temp', $data['item'])) {
            $paths = [
                'temp' => web_path() . 'temp/',
                'runtime-image' => runtime_root_path() . 'image/',
                'runtime-local' => runtime_root_path() . 'local/',
            ];
            foreach ($paths as $path) {
                $this->deleteFolder($path);
            }
        }
        return true;
    }
    private function deleteFolder($path): void
```

```
    {
        if (!is_dir($path)) {
            return;
        }
        foreach (scandir($path) as $val) {
            if (!in_array($val, ['.', '..', '.gitignore'])) {
                if (is_dir($path . $val)) {
                    $this->deleteFolder($path . $val . '/');
                    rmdir($path . $val . '/');
                } else {
                    unlink($path . $val);
                }
            }
        }
    }
}
<?php
declare (strict_types=1);
namespace app\admin\service;
use app\admin\model\Store as StoreModel;
use app\admin\validate\Store as StoreValidate;
use app\admin\service\admin\User as AdminUserService;
use app\common\service\BaseService;
use cores\exception\BaseException;
class Store extends BaseService
{
    public function add(array $data): bool
    {
        $this->validate($data);
        if (StoreModel::checkExist($data['store_name'])) {
            $this->error = '商户名称已存在';
            return false;
        }
        if ($data['password'] !== $data['password_confirm']) {
            $this->error = '确认密码不正确';
            return false;
        }
        $loginInfo = AdminUserService::getLoginInfo();
        $storeData = [
            'store_name' => $data['store_name'],
            'sort' => $data['sort'],
            'host' => $data['host'],
        ];
        $storeUserData = [
            'user_name' => $data['user_name'],
            'password' => encryption_hash($data['password']),
            'real_name' => '系统管理员',
            'is_super' => 1,
            'create_by' => $loginInfo['user']['user_name'],
            'store_id' => 0
        ];
        $model = new StoreModel;
        $res = $model->addStore($storeData, $storeUserData);
        return $res;
```

```
    }
    private function validate(array $data): void
    {
        $validate = new StoreValidate;
        if (!$validate->check($data)) {
            throwError($validate->getError());
        }
    }
}
<?php
declare (strict_types=1);
namespace app\admin\validate;
use think\Validate;
class Equipment extends Validate
{
    protected $rule = [
        'name'        =>  'require',
        'code'        =>  'require',
        'type'        =>  'require',
        'store_id'    =>  'require',
    ];
    protected $message = [
        'name.require'        => '设备名称不能为空',
        'code.require'        => '设备编号不能为空',
        'type.require'        => '设备类型不能为空',
        'store_id.require'    => '所属商户不能为空',
    ];
}
<?php
declare (strict_types=1);
namespace app\admin\validate;
use think\Validate;
class Store extends Validate
{
    protected $rule = [
        'store_name'        =>  'require',
        'host'              =>  'require',
        'user_name'         =>  'require',
        'password'          =>  'require',
        'password_confirm'  =>  'require'
    ];
    protected $message = [
        'store_name.require'        => '商户名称不能为空',
        'host.require'              => '商户域名不能为空',
        'user_name.require'         => '用户名不能为空',
        'password.require'          => '用户密码不能为空',
        'password_confirm.require'  => '确认密码不能为空'
    ];
}
<?php
declare (strict_types=1);
return [
    'imageW' => 0,
    'imageH' => 0,
```

```
    'length' => 4,
    'useCurve' => false,
    'useNoise' => true,
    'fontSize' => 26,
    'codeSet' => '23456ACEFHJKLMNPRTUVWXY',
    'checkTimes' => 5,
];
<?php
declare (strict_types=1);
namespace app\api\controller\article;
use think\response\Json;
use app\api\controller\Controller;
use app\api\model\article\Category as CategoryModel;
class Category extends Controller
{
    public function list(): Json
    {
        $model = new CategoryModel;
        $list = $model->getShowList();
        return $this->renderSuccess(compact('list'));
    }
}
<?php
declare (strict_types=1)
namespace app\api\controller\balance;
use think\response\Json;
use app\api\controller\Controller;
use app\api\model\user\BalanceLog as BalanceLogModel;
class Log extends Controller
{
    public function list(): Json
    {
        $model = new BalanceLogModel;
        $list = $model->getList();
        return $this->renderSuccess(compact('list'));
    }
}
<?php
declare (strict_types=1);
namespace app\terminal\controller\analysis;
use think\response\Json;
use app\terminal\controller\Controller;
use app\terminal\service\analysis\Analysis as AnalysisService;
class Analysis extends Controller
{
    public function supply(int $staff_id): Json
    {
        $server = new AnalysisService;
        $data = $server->supply($staff_id);
        return $this->renderSuccess($data);
    }
    public function dishesType(int $staff_id): Json
    {
        $server = new AnalysisService;
```

```
        $data = $server->dishesType($staff_id);
        return $this->renderSuccess($data);
    }
    public function dietary(int $staff_id): Json
    {
        $server = new AnalysisService;
        $data = $server->dietary($staff_id);
        return $this->renderSuccess($data);
    }
    public function vitamin(int $staff_id, int $type): Json
    {
        $server = new AnalysisService;
        $data = $server->vitamin($staff_id,$type);
        return $this->renderSuccess($data);
    }
    public function dietaryFiber(int $staff_id): Json
    {
        $server = new AnalysisService;
        $data = $server->dietaryFiber($staff_id);
        return $this->renderSuccess($data);
    }

    public function cookingMethod(int $staff_id): Json
    {
        $server = new AnalysisService;
        $data = $server->cookingMethod($staff_id);
        return $this->renderSuccess($data);
    }
    public function score(int $staff_id,int $group,array $type,string $evaluate): Json
    {
        $server = new AnalysisService;
        if ($server->score($staff_id, $group, $type, $evaluate)) {
            return $this->renderSuccess('添加成功');
        }
        return $this->renderError($server->getError() ?: '添加失败');
    }
}
<?php
declare (strict_types=1);
namespace app\terminal\controller\client\h5;
use think\response\Json;
use app\terminal\controller\Controller;
use app\terminal\model\h5\Setting as SettingModel;
class Setting extends Controller
{
    public function detail(string $key): Json
    {
        $detail = SettingModel::getItem($key);
        return $this->renderSuccess(compact('detail'));
    }
    public function update(string $key): Json
    {
        $model = new SettingModel;
        if ($model->edit($key, $this->postForm())) {
```

```
            return $this->renderSuccess('更新成功');
        }
        return $this->renderError($model->getError() ?: '更新失败');
    }
}
<?php
declare (strict_types=1);
namespace app\terminal\controller\client\wxapp;
use app\terminal\controller\Controller;
use app\terminal\model\wxapp\Setting as SettingModel;
use think\response\Json;
class Setting extends Controller
{
    public function detail(string $key): Json
    {
        $detail = SettingModel::getItem($key);
        $domain = $this->request->host(true);
        return $this->renderSuccess(compact('detail', 'domain'));
    }
    public function update(string $key): Json
    {
        $model = new SettingModel;
        if ($model->edit($key, $this->postForm())) {
            return $this->renderSuccess('更新成功');
        }
        return $this->renderError($model->getError() ?: '更新失败');
    }
}
<?php
declare (strict_types=1);
namespace app\terminal\controller\content\article;
use think\response\Json;
use app\terminal\controller\Controller;
use app\terminal\model\article\Category as CategoryModel;
class Category extends Controller
{
    public function list(): Json
    {
        $model = new CategoryModel;
        $list = $model->getList();
        return $this->renderSuccess(compact('list'));
    }
    public function add(): Json
    {
        $model = new CategoryModel;
        if ($model->add($this->postForm())) {
            return $this->renderSuccess('添加成功');
        }
        return $this->renderError($model->getError() ?: '添加失败');
    }
    public function edit(int $categoryId): Json
    {
        $model = CategoryModel::detail($categoryId);
        if ($model->edit($this->postForm())) {
```

```
            return $this->renderSuccess('更新成功');
        }
        return $this->renderError($model->getError() ?: '更新失败');
    }
    public function delete(int $categoryId): Json
    {
        $model = CategoryModel::detail($categoryId);
        if (!$model->remove($categoryId)) {
            return $this->renderError($model->getError() ?: '删除失败');
        }
        return $this->renderSuccess('删除成功');
    }
}
<?php
declare (strict_types = 1);
namespace app\terminal\service;
use app\terminal\model\SignAction as SignActionModel;
use app\terminal\model\UploadFile as UploadFileModel;
use app\terminal\service\store\User as StoreUserService;
use app\terminal\validate\SignAction as SignActionValidate;
use app\common\service\BaseService;
use cores\exception\BaseException;
class Sign extends BaseService
{
    public $signId;
    public function getList(array $param = []): \think\Paginator
    {
        $model = new SignActionModel;
        return $model->getList($param);
    }
    public function add(array $data): bool
    {
        $this->validate($data);
        $userId = StoreUserService::getLoginUserId();
        $model = new SignActionModel;
        $res = $model->add($data, $userId);
        $this->signId = $model['id'] ?: 0;
        return $res;
    }
    public function detail(int $signId)
    {
        $with = ['addressList'];
        $signInfo = SignActionModel::detail($signId,  $with);
        if (empty($signInfo)) {
            throwError('打卡活动不存在');
        }
        $signInfo = $signInfo->toArray();
        $file_ids = [$signInfo['cover_img_id'], $signInfo['rule_img_id']];
        if (!empty($signInfo['addressList'])) {
            $qrurl_ids = array_column($signInfo['addressList'], 'qrurl_id');
            $receive_url_ids = array_column($signInfo['addressList'], 'receive_url_id');
            $file_ids = array_merge($file_ids, $qrurl_ids, $receive_url_ids);
        }
        $file_ids = array_filter(array_unique($file_ids));
```

```
        if (!empty($file_ids)) {
            $uploadFileModel = new UploadFileModel();
            $files = $uploadFileModel->getAll(['file_ids' => $file_ids]);
            if (!empty($files)) {
                $files = array_column($files, null, 'file_id');
            }
        }
        $cover_img =  $files[$signInfo['cover_img_id']] ?? [];
        $rule_img =  $files[$signInfo['rule_img_id']] ?? [];
        $signInfo['cover_img'] = $cover_img;
        $signInfo['cover_img_url'] = $cover_img['external_url'] ?? '';
        $signInfo['rule_img'] = $rule_img;
        $signInfo['rule_img_url'] = $rule_img['external_url'] ?? '';
        if (!empty($signInfo['addressList'])) {
            $signInfo['addressList'] = array_map(function ($item) use ($files) {
                $item['qrurl'] = $files[$item['qrurl_id']]['external_url'] ?? '';
                $item['receive_url'] = $files[$item['receive_url_id']]['external_url'] ?? '';
                return $item;
            }, $signInfo['addressList']);
        }
        return $signInfo;
    }
    public function edit(int $signId, array $data): bool
    {
        $this->validate($data);
        $userId = StoreUserService::getLoginUserId();
        $with = ['addressList'];
        $signInfo = SignActionModel::detail($signId,  $with);
        if (empty($signInfo)) {
            throwError('打卡活动不存在');
        }
        return $signInfo->edit($data, $userId);
    }
    public function setDelete(int $signId): bool
    {
        $model = SignActionModel::detail($signId);
        if (empty($model)) {
            throwError('打卡活动不存在');
        }
        return $model->setDelete();
    }
    private function validate(array $data): void
    {
        $validate = new SignActionValidate;
        if (!$validate->check($data)) {
            throwError($validate->getError());
        }
    }
}
<?php
declare (strict_types = 1);
namespace app\terminal\service\staff;
use app\terminal\model\Equipment as EquipmentModel;
use app\terminal\model\staff\Staff as StaffModel;
```

```php
use app\terminal\model\staff\StaffFace as StaffFaceModel;
use app\terminal\model\UploadFile as UploadFileModel;
use app\terminal\model\message\MqMessage as MqMessageModel;
use app\terminal\service\store\User as StoreUserService;
use app\terminal\validate\StaffFace as StaffFaceValidate;
use app\common\service\BaseService;
use app\common\service\Face as FaceService;
use cores\exception\BaseException;
class Face extends BaseService
{
    public $id;
    public function add(array $data): bool
    {
        $this->validate($data);
        $staff = StaffModel::detail($data['staff_id']);
        if (empty($staff)) {
            throwError('用户不存在');
        }
        $faceStatus = 2;
        $model = StaffFaceModel::detail($data['staff_id']);
        if (empty($model)) {
            $model = new StaffFaceModel;
            $data['create_by'] = StoreUserService::getLoginUserName();
            $faceStatus = 1;
        }
        $data['store_id'] = $this->getStoreId();
        $res = $model->save($data);
        $this->id = $model['id'] ?: 0;
        if ($res) {
            (new FaceService)->pushMessage($data['staff_id'], $faceStatus);
        }
        return $res;
    }
    public function setDelete(int $staff_id): bool
    {
        $model = StaffFaceModel::detail($staff_id);
        if (empty($model)) {
            return true;
        }
        $res = $model->setDelete();
        if ($res) {
            (new FaceService)->pushMessage($staff_id, 3);
        }
        return $res;
    }
    private function validate(array $data): void
    {
        $validate = new StaffFaceValidate;
        if (!$validate->check($data)) {
            throwError($validate->getError());
        }
    }
    public function facedata(string $code, int $type = 0): array
    {
```

```
        $model = EquipmentModel::withoutGlobalScopeGetOne(['code' => $code, 'is_delete'
=> 0]);
        if (empty($model)) {
            throwError('设备不存在');
        }
        $equipment = $model->toArray();
        if ($equipment['mate_flag'] != 1 || $equipment['store_id'] != self::getStoreId()) {
            throwError('设备未配对');
        }
        if ($equipment['store_id'] != self::getStoreId()) {
            throwError('设备无权限获取人脸数据');
        }
        $mqMessageModel = new MqMessageModel;
        $mq_message_where = array(
            'type' => 1,
            'receive_device_id' => $code,
            'status' => 1,
            'is_delete' => 0,
        );
        $face_data = [];
        if ($type == 1) {
            $staffFaces = StaffFaceModel::getAllData(['is_delete' => 0]);
            if (!empty($staffFaces)) {
                $staff_ids = array_column($staffFaces, 'staff_id');
                $staffs = StaffModel::getAllData(['id' => $staff_ids, 'is_delete' => 0]);
                $staffs = array_column($staffs, null, 'id');
                $file_ids = array_column($staffFaces, 'file_id');
                $faceFiles = UploadFileModel::getAllData(['file_id' => $file_ids]);
                $faceFiles = array_column($faceFiles, null, 'file_id');
                foreach ($staffFaces as $staffFace) {
                    $staff = $staffs[$staffFace['staff_id']] ?? null;
                    if (empty($staff)) {
                        continue;
                    }
                    $faceFile = $faceFiles[$staffFace['file_id']] ?? null;
                    if (empty($faceFile['preview_url'])) {
                        continue;
                    }
                    $face_data[] = [
                        'type'         => 1,
                        'name'         => $staff['name'],
                        'staff_id'     => $staffFace['staff_id'],
                        'head_img_url' => $faceFile['preview_url'] ?? '',
                        'file_id'      => $staffFace['file_id'] ?? '',
                        'face_token'   => $staffFace['face_token'] ?? '',
                        'status'       => 1,
                        'msg_id'       => 0
                    ];
                }
            }
            $mqMessages = MqMessageModel::getAllData($mq_message_where);
            if (!empty($mqMessages)) {
                $mqMessage_ids = array_column($mqMessages, 'id');
                $mqMessageModel->where('id', 'in', $mqMessage_ids)
```

```
                ->where('status', 1)
                ->update(['feedback_time' => date('Y-m-d H:i:s'), 'status' => 3]);
            }
        } else {
            $mqMessages = MqMessageModel::getAllData($mq_message_where);
            if (!empty($mqMessages)) {
                array_multisort(array_column($mqMessages, 'publish_time'), SORT_DESC,
$mqMessages);
                $face_data_key = [];
                foreach ($mqMessages as $mqMessage) {
                    if (empty($face_data_key[$mqMessage['staff_id']])) {
                        $face_data_key[$mqMessage['staff_id']] =
json_decode($mqMessage['message'], true);
                    }
                }
                $mqMessage_ids = array_column($mqMessages, 'id');
                $mqMessageModel->where('id', 'in', $mqMessage_ids)
                    ->where('status', 1)
                    ->update(['feedback_time' => date('Y-m-d H:i:s'), 'status' => 3]);
                $face_data = array_values($face_data_key);
            }
        }
        return $face_data;
    }
    public function feedback(string $code, int $msg_id): bool
    {
        $mqMessageModel = new MqMessageModel;
        $mqMessage = $mqMessageModel::getOne(['msg_id' => $msg_id,
'receive_device_id' => $code, 'status' => 1, 'is_delete' => 0]);
        if (!empty($mqMessage)) {
            $mqMessage->save(['status' => 2]);
        }
        return true;
    }
}
```