

目录

ChangeLog	1.1
AIOT应用计算一体机简介	1.2
版本说明	1.2.1
一体机对接指南	1.3
接口鉴权规则	1.3.1
错误码定义	1.3.2
数据字典	1.3.3
帮助工具	1.3.4
图片上传Demo	1.3.4.1
接口使用流程	1.3.5
API接口文档	1.4
人员管理	1.4.1
上传人员图片	1.4.1.1
上传人员图片（人脸质量判断按照最低标准进行判断）	1.4.1.2
入库质量检测	1.4.1.3
批量添加人员	1.4.1.4
批量删除人员	1.4.1.5
修改人员	1.4.1.6
搜索人员列表	1.4.1.7
批量查询人员详情信息	1.4.1.8
查询人员详情信息	1.4.1.9
获取员工、访客的二维码	1.4.1.10
人员uuid自定义开关	1.4.1.11
组织管理	1.4.2
添加组织	1.4.2.1
修改组织信息	1.4.2.2
查询组织列表	1.4.2.3
删除组织	1.4.2.4
获取最上级组织信息	1.4.2.5
人员组管理	1.4.3
批量添加人员组	1.4.3.1
删除人员组	1.4.3.2
修改人员组	1.4.3.3
人员组列表	1.4.3.4
查询人员组详情	1.4.3.5

批量移除组内人员	1.4.3.6
查询组内人员详情	1.4.3.7
编辑组内人员	1.4.3.8
获取默认组信息（不建议使用）	1.4.3.9
设备管理	1.4.4
设备添加	1.4.4.1
发现设备列表	1.4.4.2
面板机设备导入	1.4.4.3
删除设备预校验	1.4.4.4
删除设备	1.4.4.5
修改设备信息	1.4.4.6
查询设备列表	1.4.4.7
查询设备信息（不建议使用）	1.4.4.8
查询设备详情	1.4.4.9
查询算力设备可绑定的设备列表	1.4.4.10
按类型分组设备列表（不建议使用）	1.4.4.11
查询设备型号关系（不建议使用）	1.4.4.12
设备绑定	1.4.4.13
设备解绑	1.4.4.14
时间计划管理	1.4.5
添加时间计划集合	1.4.5.1
修改时间计划	1.4.5.2
批量删除时间计划	1.4.5.3
时间计划列表	1.4.5.4
查询时间计划详情列表	1.4.5.5
门禁点管理	1.4.6
查询绑定B3R的端设备列表	1.4.6.1
查询门禁点设备列表	1.4.6.2
查询同步失败人员信息	1.4.6.3
查询绑定设备类型	1.4.6.4
查询设备配置信息	1.4.6.5
批量添加和移除门禁点	1.4.6.6
移除门禁点	1.4.6.7
绑定开关	1.4.6.8
远程常开	1.4.6.9
远程常闭	1.4.6.10
远程开门	1.4.6.11
恢复正常	1.4.6.12

配置设备	1.4.6.13
查询同步列表（废弃）	1.4.6.14
同步失败详情（废弃）	1.4.6.15
门禁设备告警	1.4.7
查询门禁设备告警列表	1.4.7.1
通行权限管理	1.4.8
添加通行权限	1.4.8.1
删除通行权限	1.4.8.2
编辑通行权限	1.4.8.3
通行权限列表	1.4.8.4
通行记录	1.4.9
查询通行记录列表	1.4.9.1
考勤设备管理	1.4.10
批量添加考勤设备	1.4.10.1
批量删除考勤设备	1.4.10.2
考勤设备列表查询	1.4.10.3
考勤组管理	1.4.11
创建考勤组	1.4.11.1
批量删除考勤组	1.4.11.2
更新考勤组	1.4.11.3
考勤员工分页列表查询	1.4.11.4
考勤组分页列表查询	1.4.11.5
考勤组详情查询	1.4.11.6
考勤记录	1.4.12
员工月考勤记录分页列表查询	1.4.12.1
员工考勤记录日历模式查询	1.4.12.2
考勤记录分页列表查询	1.4.12.3
考勤记录详情分页列表查询	1.4.12.4
考勤记录调整	1.4.12.5
考勤统计	1.4.13
考勤统计记录分页列表查询	1.4.13.1
出勤调整	1.4.14
出勤调整	1.4.14.1
出勤调整记录分页列表查询	1.4.14.2
更新出勤调整记录	1.4.14.3
功能配置	1.4.15
获取一脸通配置	1.4.15.1
重置一脸通配置	1.4.15.2

一脸通配置保存	1.4.15.3
获取二维码配置	1.4.15.4
保存二维码配置	1.4.15.5
查看入库标准	1.4.15.6
保存入库标准	1.4.15.7
重置入库标准	1.4.15.8
测温、口罩、健康码开关查询	1.4.15.9
鸿图页面回调数据	1.4.16
事件记录回调	1.4.16.1
事件记录回调说明	1.4.16.1.1
事件记录回调数据结构	1.4.16.1.2
事件回调数据示例	1.4.16.1.3
通行记录回调	1.4.16.2
考勤记录回调	1.4.16.3
业务数据推送	1.4.17
业务数据记录订阅	1.4.17.1
查询业务数据记录订阅	1.4.17.2
解除业务数据记录订阅	1.4.17.3
业务推送数据结构	1.4.17.4
通行识别记录推送	1.4.17.4.1
通用模块	1.4.18
上传文件（可用于人脸比对）	1.4.18.1
运维管理	1.4.19
查询设备信息	1.4.19.1
人脸比对	1.4.20
1：N列表查询	1.4.20.1
1:1查询	1.4.20.2

- [1. 1.0.6](#)
 - [1.1. 接口变更](#)
 - [1.2. 数据字典变更](#)

1. 1.0.6

1.1. 接口变更

模块	接口	操作	描述
组织管理	添加组织	新增	添加新的组织
组织管理	修改组织信息	新增	修改组织信息
组织管理	查询组织列表	新增	查询组织信息
组织管理	删除组织	新增	删除组织信息
组织管理	获取最上级组织信息	新增	获取最上级组织信息
通用模块	上传文件（可用于人脸比对）	新增	进行文件上传
人脸对比	人脸对比	新增	上传两张人脸图片进行人脸相似度比分
人脸对比	1：N列表查询	新增	上传图片进行比分获取底库最相似人
门禁点管理	远程常闭	新增	远程控制设备，将面板机设备远程常闭，不支持非面板机设备
门禁点管理	查询门禁点设备列表	变更	返回值中增加"常开常闭状态"

模块	接口	操作	描述
门禁点管理	查询同步失败人员信息	变更	查询同步失败人员信息接口请求参数"uuid"变更为"noticeCode", 返回值新增"reasonStr"
门禁点管理	查询设备配置信息	变更	查询设备配置信息, 返回值"测温参数"相关字段补充
门禁点管理	远程常开	新增	远程控制设备, 将面板机设备远程常开, 不支持非面板机设备
门禁点管理	恢复正常	新增	远程控制设备, 将远程常开、远程常闭的面板机设备恢复正常, 不支持非面板机设备
门禁点管理	配置设备	变更	配置设备新增参数测温对象"temperatureConfig"、健康码参数"healthkitConfig"
门禁点管理	同步失败详情 (废弃)	变更	同步失败详情 (废弃), 请求参数"uuid"变更为"noticeCode", 返回值新增"reasonStr"
设备管理	设备添加	变更	请求参数"deviceMode"更新注释、"rtspUrl"更新注释, 新增"protocol"字段
设备管理	删除设备预校验	变更	返回值"tag"注释更新
设备管理	修改设备信息	变更	新增参数"activeMode"

模块	接口	操作	描述
设备管理	查询设备列表	变更	请求参数"deviceMode"注释更新, 返回值"deviceType"、"deviceMode"注释更新
设备管理	查询设备信息 (不建议使用)	变更	1.返回值新增字段 协议类型 protocol 2.返回值新增字段 是否支持主动模式 activeMode 3.返回值新增字段 是否支持activeSdk isSupportActiveSdk 4.返回值新增字段 常开常闭状态码 cmdStatus 5.返回值新增字段 1-SDK, 2-1400 joinProtocol
设备管理	查询设备详情	变更	1.请求参数"deviceMode"注释更新 2.返回值新增字段 协议类型 protocol 3.返回值新增字段 是否支持主动模式 activeMode
考勤组管理	创建考勤组	变更	1.请求参数中date由毫秒时间戳改为yyyy-MM-dd格式 2.返回值中date由毫秒时间戳改为yyyy-MM-dd格式
考勤组管理	更新考勤组	变更	1.请求参数中date由毫秒时间戳改为yyyy-MM-dd格式 2.返回值中date由毫秒时间戳改为yyyy-MM-dd格式
考勤组管理	考勤组分页列表查询	变更	1.返回值中date由毫秒时间戳改为yyyy-MM-dd格式 2.返回值中weekPlan修改注释
考勤组管理	考勤组详情查询	变更	1.返回值中date由毫秒时间戳改为yyyy-MM-dd格式 2.返回值中weekPlan修改注释
规则管理	添加规则	变更	1.请求参数中类型type由必须改为非必须 2.请求参数中增加typeList 3.返回值在successes和failures里面类型type由必须改为非必须 4.返回值在successes和failures里面增加typeList属性

模块	接口	操作	描述
规则管理	修改规则	变更	1.请求参数中groupList修改注释 2.请求参数中增加typeList
规则管理	规则分页查询	变更	1.请求参数中type修改注释 2.返回值中type修改注释 3.返回值中groupList修改注释 4.返回值中增加typeList
功能配置	获取二维码配置	新增	获取人员配置中员工、访客二维码有效时长配置
功能配置	保存二维码配置	新增	保存人员配置中员工、访客二维码有效时长配置
功能配置	重置一脸通配置	变更	增加健康码检测开关配置参数
功能配置	一脸通配置保存	变更	增加口罩提示开关、健康码检测开关配置参数
功能配置	测温、口罩、健康码开关查询	变更	增加口罩提示开关、健康码检测开关参数
通行记录	查询通行记录列表	变更	增加健康码、人员类型、个性签名参数
鸿图页面回调数据	通行记录回调	变更	推送数据中recognitionType状态4由陌生人修改为无权限人员、增加个性签名字段

模块	接口	操作	描述
业务数据推送	通行识别记录推送	变更	推送数据中recognitionType状态4由陌生人修改为无权限人员、增加个性签名字段
人员管理	入库质量检测	新增	上传人脸图片文件质量检测
人员管理	上传人员图片 (人脸质量判断按照最低标准进行判断)	新增	上传人员图片(使用非系统入库参数,适用于1:N图片上传等场景,不建议入库)
人员管理	入库质量检测	新增	上传人脸图片文件质量检测, 接口内部包含了人脸质量判断, 不做上传, 只是个质量判断。
人员管理	批量添加人员	变更	请求参数phone字段修改为访客类型手机号必填, 其他人员类型非必填 请求参数、返回值中增加【autograph】个性签名 请求参数、返回值中增加【orgUuid】组织uuid
人员管理	修改人员	变更	请求参数phone字段修改为访客类型手机号必填, 其他人员类型非必填 请求参数中增加【autograph】个性签名字段 请求参数中增加【orgUuid】组织uuid字段
人员管理	搜索人员列表	变更	请求参数增加【orgUuid】组织uuid字段 返回值增加【autograph】个性签名字段 返回值增加【orgUuid】组织uuid字段 返回值增加【orgName】组织名称字段
人员管理	批量查询人员详情信息	变更	返回值增加【autograph】个性签名字段 返回值增加【orgUuid】组织uuid字段 返回值增加【orgName】组织名称字段
人员管理	查询人员详情信息	变更	请求参数中的uuid备注修改为人员uuid 返回值增加【autograph】个性签名字段 返回值增加【orgUuid】组织uuid字段 返回值增加【orgName】组织名称字段

模块	接口	操作	描述
人员管理	获取员工、访客的二维码	变更	接口描述修改为【员工、访客获取二维码使用】 请求参数中uuid备注修改为员工、访客uuid 返回值中qrCode备注修改为员工、访客二维码图片(base64编码的字符串) 返回值增加【expireTime】二维码图片的过期分钟数字段
时间计划管理	添加时间计划集合	变更	请求参数、返回值holiday备注修改为节假日(事件时间计划不支持假日) 请求参数、返回值type备注修改为类型, 1 通行, 3 事件
时间计划管理	修改时间计划	变更	请求参数holiday备注修改为节假日(事件时间计划不支持假日)
时间计划管理	时间计划列表	变更	返回值type备注修改为类型, 1 通行, 3 事件
时间计划管理	查询时间计划详情列表	变更	返回值type备注修改为类型, 1 通行, 3 事件
设备管理	设备添加	变更	1.请求参数和返回值新增字段 协议类型 protocol 2.请求参数和返回值新增字段 是否支持主动模式 activeMode 3.请求参数和返回值新增字段 设备sn码, PDNS接入时必填 snCode 4.请求参数新增字段 接入协议 joinProtocol
设备管理	修改设备信息	变更	1.请求参数新增字段 是否支持主动模式 activeMode

模块	接口	操作	描述
设备管理	查询设备详情	变更	1.返回值新增字段 服务器模式 serverMode 2.返回值新增字段 协议类型 protocol 3.返回值新增字段 设备编码 snCode 4.返回值新增字段 是否支持主动模式 activeMode 5.返回值新增字段 是否支持activeSdk isSupportActiveSdk 6.返回值新增字段 常开常闭状态码 cmdStatus 7.返回值新增字段 接入协议 1-SDK, 2-1400 joinProtocol
设备管理	查询可布控设备列表	变更	1.返回值新增字段 是否是all-in-one模式 (type=0)下面的警戒模式还是 sensor+compute的警戒模式(type=1),其他模式(type=2) type 2.返回值新增字段 是否支持主动模式 1 主动模式 activeMode 3.返回值新增字段 设备sn码 snCode
门禁点管理	查询门禁点设备列表	变更	1.返回值新增字段 常开常闭状态 cmdStatus
门禁点管理	查询同步失败人员信息	变更	1.请求参数新增字段 通知类型: 1-人员照片下发失败, 6-人脸特征抽取失败 noticeCode 2.返回值新增字段 失败原因描述 reasonStr

模块	接口	操作	描述
门禁点管理	查询设备配置信息	变更	1.返回值新增字段 健康码参数 healthKitConfig 2.返回值新增字段 是否开启健康码功能 0 关闭 1 开启 healthKitMode 3.返回值新增字段 健康码省份列表 healthKitProvinceList 4.返回值新增字段 城市编码 cityCode 5.返回值新增字段 城市名字 cityName 6.返回值新增字段 健康码使用的省份 healthKitProvince 7.返回值新增字段 坐标上传开启 0:关闭 1: 开启 healthKitCoordinateMode 8.返回值新增字段 坐标经度 healthKitLongitude 9.返回值新增字段 健康码绿码 healthKitGreen 10.返回值新增字段 健康码红码 healthKitRed 11.返回值新增字段 健康码黄码 healthKitYellow 12.返回值新增字段 健康码未知 healthKitUnknown 13.返回值新增字段 健康码超时 healthKitTimeout 14.返回值新增字段 二维码图片uri healthKitImageUrl 15.返回值新增字段 身份证模块开关 healthKitIdcardVerify 16.返回值新增字段 自动保存身份证号 码开关 healthKitIdcardAutosave 17.返回值新增字段 通行时间权限开关 healthKitScheduleVerify 18.返回值新增字段 高温禁行开关 healthKitHighTemperatureNoEntry 19.返回值新增字段 高温阈值 healthKitHighTemperatureThreshold 20.返回值新增字段 健康码平台地址 healthKitServerAddress 21.返回值新增字段 健康码测温灵敏度 healthKitTemperatureSensitivity

模块	接口	操作	描述
门禁点管理	配置设备	变更	1.返回值新增字段 健康码参数 healthKitConfig 2.返回值新增字段 是否开启健康码功能 0 关闭 1 开启 healthKitMode 3.返回值新增字段 健康码省份列表 healthKitProvinceList 4.返回值新增字段 城市编码 cityCode 5.返回值新增字段 城市名字 cityName 6.返回值新增字段 健康码使用的省份 healthKitProvince 7.返回值新增字段 坐标上传开启 0:关闭 1: 开启 healthKitCoordinateMode 8.返回值新增字段 坐标经度 healthKitLongitude 9.返回值新增字段 健康码绿码 healthKitGreen 10.返回值新增字段 健康码红码 healthKitRed 11.返回值新增字段 健康码黄码 healthKitYellow 12.返回值新增字段 健康码未知 healthKitUnknown 13.返回值新增字段 健康码超时 healthKitTimeout 14.返回值新增字段 二维码图片uri healthKitImageUrl 15.返回值新增字段 身份证模块开关 healthKitIdcardVerify 16.返回值新增字段 自动保存身份证号开关 healthKitIdcardAutosave 17.返回值新增字段 通行时间权限开关 healthKitScheduleVerify 18.返回值新增字段 高温禁行开关 healthKitHighTemperatureNoEntry 19.返回值新增字段 高温阈值 healthKitHighTemperatureThreshold 20.返回值新增字段 健康码平台地址 healthKitServerAddress 21.返回值新增字段 健康码测温灵敏度 healthKitTemperatureSensitivity
功能配置	获取一脸通配置	变更	1.返回值新增字段 健康码检测开关, 0: 关闭, 1: 打开 healthKitSwitch
功能配置	重置一脸通配置	变更	1.返回值新增字段 健康码检测开关, 0: 关闭, 1: 打开 healthKitSwitch

模块	接口	操作	描述
功能配置	一脸通配置保存	变更	1.返回值和请求参数新增字段 健康码检测开关, 0: 关闭, 1: 打开 healthKitSwitch
功能配置	测温、口罩、健康码开关查询	变更	1.返回值新增字段 健康码检测开关, 0: 关闭, 1: 打开 healthKitSwitch
通行记录	查询通行记录列表	变更	1.请求参数新增字段 健康码状态 healthyState 2.返回值新增字段 访问人员的类型 visitPersonType 3.返回值新增字段 健康码状态 healthyState 4.返回值新增字段 健康码通行状态 healthkitPasstype 5.返回值新增字段 个性签名 autograph
通行记录回调	通行记录回调	变更	1.返回值新增字段 个性签名 autograph

1.2. 数据字典变更

模块	数据字典	操作	描述
系统安全参数配置	人脸下发方式	新增	新增的枚举类型
设备	设备模式	新增	新增12表示全模式
设备	设备类型	变更	1.智能摄像头增加Smart IPC 2.智能面板机增加MegEye-W4K-E82和SmartEye-W4K-E7
设备	设备型号	新增	新增13表示Smart IPC, 44表示MegEye-W4K-E82, 45表示SmartEye-W4K-E7
设备	设备型号	删除	MegCube-B4H16-311由39改为57
事件	人脸下发方式	新增	新增的枚举类型

- 1. AIOT应用计算一体机简介
 - 1.1. 产品概述

1. AIOT应用计算一体机简介

1.1. 产品概述

旷视鸿图2000系列AIoT应用计算一体机（MegCube-HAL2002-4S1）是面向中小型楼宇园区，主要提供出入口场景方案的自研核心产品。。

鸿图基于低功耗X86平台+平台软件+高性能算力芯片+出厂自带授权的一体化形态，可采用内置算力芯片进行视频流AI解析，并可配合门禁一体机、智能分析盒和智能相机等智能边端产品，为用户提供高性价比的智慧通行、考勤、访客、迎宾等智能应用业务。

同时，鸿图提供丰富的OpenAPI接口，支持将AI数据和业务数据上传给行业业务平台，为AI赋能百业提供AI数据汇聚和分析能力。

通行记录回调

鸿图2000.V106版本

- 1. 接口鉴权
 - 1.1. 鉴权方式
 - 1.2. 签名计算规则
 - 1.3. 签名示例
 - 1.4. 可用签名及秘钥
 - 1.5. 签名SDK
 - 1.5.1. Java版本

1. 接口鉴权

1.1. 鉴权方式

OpenAPI请求采用接口签名方式，需要在HTTP HEADER中加入以下请求头：

序号	header名	描述
1	ctimestamp	客户端本地unix毫秒时间戳
2	cnonce	6位随机字符串，可含字母，客户端自行生成
3	cappkey	标识第三方系统身份，由鸿图分配
4	csign	按签名规则计算的签名

1.2. 签名计算规则

签名算法

- 签名算法：MD5，32位小写
- sign计算规则：csign=MD5(uri-method-queryParamStr-requestBodyMD5-secretKey-ctimestamp-cnonce-cappkey)
 - 数据解释：
 - uri：必须，请求URI，以"/"开头，如："/v1/api/person/list"
 - method：必须，请求方式，全大写，如："POST"
 - queryParamStr：非必须，query域传值，有值传形如"name=张三&age=28&..."字符串；无值传""
 - requestBodyMD5：非必须，仅文件上传接口可为空；否则为JSON数据或"{}"的MD5加密串(为空会报JSON解析异常)
 - secretKey：必须，鸿图平台提供的签名私钥
 - ctimestamp：必须，当前时间戳
 - cnonce：必须，随机码
 - cappkey：必须，鸿图平台提供的签名公钥
 - csign：生成的最终MD5签名串
 - 示例：
 - uri="/v1/api/person/list"
 - method="POST"
 - queryParamStr=""
 - requestBodyMD5="dcb8b5bfe22ef26821b9b34e650c534b"

- secretKey="sdfajk3242324falldjq7"
- ctimestamp="1627300328673"
- cnonce="1234344"
- cappkey="appkey1"

1.3. 签名示例

以人员列表查询为例，通过在线Demo计算签名，Demo地址：<https://cloud9-package.oss-cn-beijing.aliyuncs.com/hongtusignutil.html>

说明：本工具帮助您在开发中快速生成OpenAPI的签名，替代您手动签名。
注意：1. 支持文件上传接口签名；2. 支持JSON字符串；3. 不支持二进制数据；4. 不支持请求体为空；5. 不支持请求体为null；6. 不支持JSON字符串。

uri: /v1/api/person/list

method: POST

queryParamStr(改研): {"pageNum": 1, "pageSize": 10}

requestBody(JSON或{}): {"pageNum": 1, "pageSize": 10}

secretKey: sdfajk3242324falldjq7

ctimestamp: 1627300328673

cnonce: 1234344

cappkey: appkey1

计算签名

uri=/v1/api/person/list
requestBodyMD5=md5({"pageNum": 1, "pageSize": 10})=675c299bcd88edf31c09c48f9c7fba4b
secretKey=sdfajk3242324falldjq7
uri-method-queryParamStr-requestBodyMD5-secretKey-ctimestamp-cnonce-cappkey=/v1/api/person/list-POST-675c299bcd88edf31c09c48f9c7fba4b-sdfajk3242324falldjq7-1627300328673-1234344-appkey1
csign=md5(uri-method-queryParamStr-requestBodyMD5-secretKey-ctimestamp-cnonce-cappkey)=52141def7a1e346ceee2f22c43d5fb97

请求体MD5

签名串

请求参数如下：

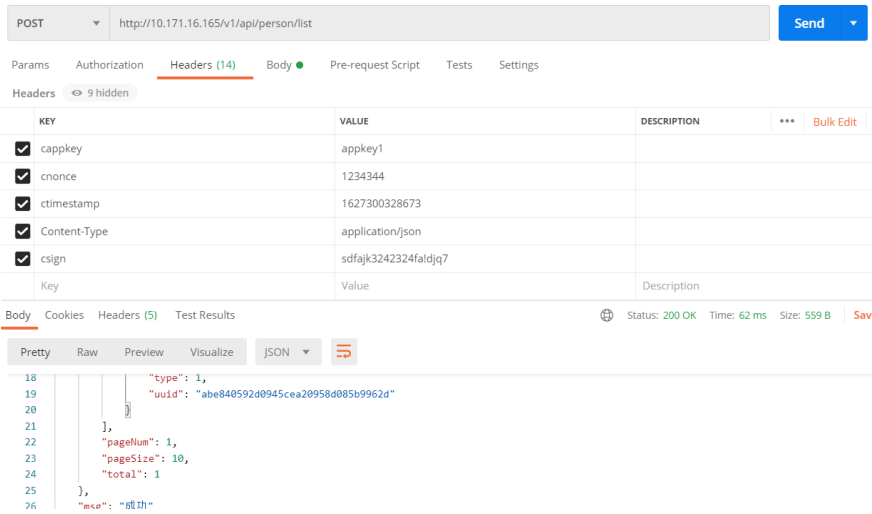
- uri="/v1/api/person/list"
- method="POST"
- requestBody（请求体）：

```
{"pageNum": 1, "pageSize": 10}
```

则requestBodyMD5为：675c299bcd88edf31c09c48f9c7fba4b

- secretKey="sdfajk3242324falldjq7"
- ctimestamp="1627300328673"
- cnonce="1234344"
- cappkey="appkey1"
- 则csign=MD5("/v1/api/person/list-POST-675c299bcd88edf31c09c48f9c7fba4b-sdfajk3242324falldjq7-1627300328673-1234344-appkey1"), 结果为：52141def7a1e346ceee2f22c43d5fb97

那么，我们可以如下请求：



1.4. 可用签名及密钥

目前，鸿图里面有两组cappkey和密钥，如下：

cappkey	secret
appkey1	sdfajk3242324fa!djQ7
appkey2	zxcv&jlkfd234adf#xi9

1.5. 签名SDK

1.5.1. Java版本

```

package com.megvii.sign.sdk;
import com.alibaba.fastjson.JSONObject;
import org.apache.commons.lang3.StringUtils;
/**
描述:
Author: liuxing
Date: 2020-09-11
*/
public class SignUtils {
    private static final String SIGN_FORMAT = "%s-%s-%s-%s-%s-%s-%s-%s";
    /**
签名算法生成卡密昂
@param ctimestamp
@param cnonce
@param requestBody 请求体对象
@param requestParam
@param url
@param method
@param capKey
@param secret
@return
*/
    public static String sign(String ctimestamp, String cnonce, Object requestBody, String requestParam, String url, String method, String capKey, String secret) {
        String requestBodyMd5 = "";
        if (requestBody != null) {
            String requestBodyJson = JSONObject.toJSONString(requestBody);
            if (StringUtils.isNotEmpty(requestBodyJson)) {
                requestBodyMd5 = DigestUtil.encryptMd5(requestBodyJson);
            }
        }
        String signStr = String.format(SIGN_FORMAT, url, method, requestParam, requestBodyMd5, ctimestamp, cnonce, capKey, secret);
        String sign = DigestUtil.encryptMd5(signStr);
        return sign;
    }
}

package com.megvii.sign.sdk;
import javax.crypto.Cipher;
import javax.crypto.SecretKey;
import javax.crypto.SecretKeyFactory;
import javax.crypto.spec.DESedeKeySpec;
import java.security.MessageDigest;
import java.security.NoSuchAlgorithmException;
public class DigestUtil {
    /** 算法名称 */
    private static final String ALGORITHM = "DESede";
    /** 16进制字母 */
    private static final char[] HEX_DIGITS={'0','1','2','3','4','5','6','7','8','9','a','b','c','d','e','f'};
    /**
    * @param decrypt 要加密的字符串
    * @return 加密的字符串
    * SHA1加密
    */
    public final static String SHA1(String decrypt) {
        try {
            MessageDigest digest = java.security.MessageDigest
                .getInstance("SHA-1");
            digest.update(decrypt.getBytes());
            byte messageDigest[] = digest.digest();
            // Create Hex String
            StringBuffer hexString = new StringBuffer();
            // 字节数组转换为 十六进制 数
            for (int i = 0; i < messageDigest.length; i++) {
                String shaHex = Integer.toHexString(messageDigest[i] & 0xFF);
                if (shaHex.length() < 2) {
                    hexString.append(0);
                }
            }
        } catch (NoSuchAlgorithmException e) {
            e.printStackTrace();
        }
        return hexString.toString();
    }
}

```

```

        }
        hexString.append(shaHex);
    }
    return hexString.toString();
}
catch (NoSuchAlgorithmException e) {
    e.printStackTrace();
}
return "";
}
}
/**
 * 3DES加密,key必须是长度大于等于 3*8 = 24 位
 * @param src
 * @param key
 * @return
 * @throws Exception
 */
public static String encrypt3DES(String src, String key) {
    byte[] keyBytes = key.getBytes();
    try {
        DESedeKeySpec dks = new DESedeKeySpec(keyBytes);
        SecretKeyFactory keyFactory = SecretKeyFactory.getInstance(ALGORITHM);
        SecretKey securekey = keyFactory.generateSecret(dks);
        Cipher cipher = Cipher.getInstance("DESede/ECB/PKCS5Padding");
        cipher.init(Cipher.ENCRYPT_MODE, securekey);
        byte[] b=cipher.doFinal(src.getBytes());
        return StringUtil.byte2hex(b);
    }
    catch (Exception e) {
        return null;
    }
}
}
/**
 * 3DESECB解密,key必须是长度大于等于 3*8 = 24 位
 * @param src
 * @param key
 * @return
 * @throws Exception
 */
public static String decrypt3DES(String src, String key) {
    byte[] keyBytes = key.getBytes();
    try {
        //---通过base64,将字符串转成byte数组
        byte[] bytesrc = StringUtil.hex2byte(src);
        //---解密的key
        DESedeKeySpec dks = new DESedeKeySpec(keyBytes);
        SecretKeyFactory keyFactory = SecretKeyFactory.getInstance(ALGORITHM);
        SecretKey securekey = keyFactory.generateSecret(dks);
        //---Cipher对象解密
        Cipher cipher = Cipher.getInstance("DESede/ECB/PKCS5Padding");
        cipher.init(Cipher.DECRYPT_MODE, securekey);
        byte[] retbyte = cipher.doFinal(bytesrc);
        return new String(retbyte);
    }
    catch (Exception e) {
        return null;
    }
}
}
/**
 *
 * Description:md5加密
 * @Create_by:JH
 * @Create_date:2014-9-9
 * @Last_Edit_By:
 * @Edit_Description
 * @Create_Version:exinhua 1.0

```

```

    */
    public static String encryptMd5(String src) {
        if(StringUtil.isEmpty(src)) {
            return src;
        }
        try {
            byte[] btInput = src.getBytes();
            // 获得MD5摘要算法的 MessageDigest 对象
            MessageDigest mdInst = MessageDigest.getInstance("MD5");
            // 使用指定的字节更新摘要
            mdInst.update(btInput);
            // 获得密文
            byte[] md = mdInst.digest();
            // 把密文转换成十六进制的字符串形式
            return byteToHexString(md);
        }
        catch (Exception e) {
            throw new RuntimeException("error occurred when encrypt", e);
        }
    }
}
/**
 * 对输入进行sha1加密后, 在进行16进制转换
 * @param value
 * @return
 */
public static String hexSHA1(String value) {
    try {
        MessageDigest md = MessageDigest.getInstance("SHA-1");
        md.update(value.getBytes("utf-8"));
        byte[] digest = md.digest();
        return byteToHexString(digest);
    }
    catch (Exception ex) {
        throw new RuntimeException(ex);
    }
}
}
/**
 * 字节数组转换成16进制表示
 * @param bytes
 * @return
 */
public static String byteToHexString(byte[] bytes) {
    int j = bytes.length;
    char[] str = new char[j * 2];
    int k = 0;
    for (int i = 0; i < j; i++) {
        byte byte0 = bytes[i];
        str[k++] = HEX_DIGITS[byte0 >>> 4 & 0xf];
        str[k++] = HEX_DIGITS[byte0 & 0xf];
    }
    return new String(str);
}
}
package com.megvii.sign.sdk;
/**
 * 字符串工具类
 */
public class StringUtil {
    public static final String[] HEX_ARRAYS = { "0", "1", "2", "3", "4", "5",
        "6", "7", "8", "9", "a", "b", "c", "d", "e", "f" };
}
/**
 * 判断value是否是null或者其length为0
 */
 * @param value
 * @return

```



```

    */
    public static Boolean isEmpty(String value) {
        return value == null || value.length() == 0;
    }
    /**
     * 判断value是否不是null且其length>0
     *
     * @param value
     * @return
     */
    public static Boolean isEmpty(String value) {
        return !isEmpty(value);
    }
    /**
     * trim操作
     *
     * @param value
     * @return
     */
    public static String trim(String value) {
        if (value == null) {
            return null;
        }
        return value.trim();
    }
    /**
     * byte string to hex string
     *
     * @param b
     * @return
     */
    public static String byte2hex(byte[] b) {
        if (b == null) {
            return null;
        }
        if (b.length == 0) {
            return "";
        }
        StringBuilder retBuilder = new StringBuilder();
        for (int n = 0; n < b.length; ++n) {
            retBuilder.append(HEX_ARRAYS[(b[n] & 0xF0) >> 4]);
            retBuilder.append(HEX_ARRAYS[b[n] & 0x0F]);
        }
        return retBuilder.toString();
    }
    /**
     * hex string to byte string
     *
     * @param str
     * @return
     */
    public static byte[] hex2byte(String str) {
        if (str == null)
            return null;
        str = str.trim();
        int len = str.length();
        if ((len == 0) || (len % 2 == 1))
            return null;
        byte[] b = new byte[len / 2];
        byte tmp = 0;
        char[] strChar = str.toCharArray();
        for (int i = 0; i < len; i++) {
            byte t = 0;
            if (strChar[i] >= '0' && strChar[i] <= '9') {
                t = (byte) (strChar[i] - '0');
            } else if (strChar[i] >= 'A' && strChar[i] <= 'F') {

```

```

        t = (byte) (strs[i] - 'A' + 10);
    } else if (strs[i] >= 'a' && strs[i] <= 'f') {
        t = (byte) (strs[i] - 'a' + 10);
    }
    if ((i & 0x1) == 1) {
        tmp <= 4;
        tmp += t;
        b[i / 2] = tmp;
        tmp = 0;
    } else {
        tmp = t;
    }
}
return b;
}
/**
 * 判断一个字符串是否只包含数字(10进制)字符
 *
 * @param str
 * @return
 */
public static Boolean isNumeric(String str) {
    if (isEmpty(str)) {
        return false;
    }
    char[] arr = str.toCharArray();
    for (char c : arr) {
        if (!Character.isDigit(c)) {
            return false;
        }
    }
    return true;
}
/**
 * 混淆字符串(显示前pn个字符和后tn个字符, 其余全部用*填充)
 *
 * @param source
 * @param pn
 * @param tn
 * @return
 */
public static String mix(String source, int pn, int tn) {
    return mix(source, pn, tn, '*');
}
/**
 * 混淆字符串(显示前pn个字符和后tn个字符, 其余前部用mixChar填充)
 *
 * @param source
 * @param pn
 * @param tn
 * @param mixChar
 * @return
 */
public static String mix(String source, int pn, int tn, char mixchar) {
    if (source == null || source.length() <= pn + tn) {
        return source;
    }
    int len = source.length();
    StringBuilder tmp = new StringBuilder(source.length());
    char[] mobileAs = source.toCharArray();
    for (int i = 0; i < pn; i++) {
        tmp.append(mobileAs[i]);
    }
    for (int i = 0; i < len - (pn + tn); i++) {
        tmp.append(mixchar);
    }
}

```

```

        for (int i = len - tn; i < len; i++) {
            tmp.append(mobileAs[i]);
        }
        return tmp.toString();
    }
}
/**
 * 获取字符串长度(全角2,半角1)
 * @param value
 * @return
 */
public static final int getLength(String value) {
    if(StringUtil.isEmpty(value)) {
        return 0;
    }
    int len = 0;
    for (char c : value.toCharArray()) {
        len++;
        if(isSbcCase(c)) {
            len++;
        }
    }
    return len;
}
/**
 * 判断字符是否是半角
 * @param c
 * @return
 */
public static final Boolean isDbcCase(char c) {
    int k = 0x80;
    return c / k == 0 ? true : false;
}
/**
 * 判断是否是全角
 * @param c
 * @return
 */
public static final Boolean isSbcCase(char c) {
    return isDbcCase(c);
}
/**
 * 判断是否都是数字
 * @param value
 * @return
 */
public static final Boolean isAllDigits(String value) {
    if(StringUtil.isEmpty(value)) {
        return true;
    }
    for (char c : value.toCharArray()) {
        if(c < '0' || c > '9') {
            return false;
        }
    }
    return true;
}
/**
 * 在指定的字符串前面填充0, 直到字符串的长度达到指定的长度
 * 示例: 输入6, 2 ----> 返回06
 * @param data 需要被填充的字符串
 * @param fillStr 需要填充得字符串
 * @param length 目标字符串的长度
 * @return 返回填充后的字符串
 */
public static String frontFillStr(String data, String fillStr, int length)
    StringBuilder sb = new StringBuilder();

```

通行记录回调

```
        for (int i = 0; i < length-data.length(); i++) {  
            sb.append(fillStr);  
        }  
        return sb.append(data).toString();  
    }  
}
```

请求示例

```

package test;
import lombok.AllArgsConstructor;
import lombok.Builder;
import lombok.Data;
import lombok.NoArgsConstructor;
/**
 * 描述:
 * Author: liuxing
 * Date: 2020-09-14
 */
@Data
@AllArgsConstructor
@NoArgsConstructor
@Builder
public class DeviceReq {
    private String name;
    private Integer pageNum;
    private Integer deviceType;
    private Integer pageSize;
}

package test;
import com.alibaba.fastjson.JSONObject;
import com.megvii.sign.sdk.SignUtilis;
import okhttp3.*;
import java.util.concurrent.TimeUnit;
/**
 * 描述: 模拟一次请求的过程
 * Author: liuxing
 * Date: 2020-09-14
 */
public class SignUtilTest {
    public static void main(String[] args) {
        String ctimestamp = "1599648837833";
        // 时间戳, 用户自己生成
        String cnonce = "123456";
        // 6位随机码, 用户自己生成
        String requestParam = "";
        // 请求参数, form表单方式, 鸿图大部分接口都是body传参, 故这里大部分都是"", 如果有参
        String url = "/v1/api/device/list";
        // 要请求的url, 看要请求的接口说明
        String method = "POST";
        // 请求方式, 看要请求的接口说明
        String cappKey = "appkey1";
        // appkey, 固定,
        String secret = "sdfajk3242324fa!djQ7";
        // 密钥, 固定
        DeviceReq deviceReq = DeviceReq.builder().name("core").build();
        // 构建请求体, 看要请求的接口说明
        // 计算签名
        String csign = SignUtilis.sign(ctimestamp, cnonce, deviceReq, requestPa
        // 发起请求
        String realUrl = "http://10.122.101.181:18082" + url;
        postJson(ctimestamp, cnonce, cappKey, csign, deviceReq, realUrl);
    }

    private static void postJson(String ctimestamp, String cnonce, String capp
        String bodyJson = JSONObject.toJSONString(reqeustData);
        OkHttpClient httpClient = new OkHttpClient.Builder()
            .connectTimeout(3000, TimeUnit.SECONDS)
            .readTimeout(30000, TimeUnit.SECONDS)
            .writeTimeout(30000, TimeUnit.SECONDS)
            .build();
        MediaType mediaType = MediaType.parse("application/json;charset=UTF-8")
        RequestBody requestBody = RequestBody.create(mediaType, bodyJson);
        Request request = new Request.Builder()
            .url(url)

```

```
        .addHeader("cnonce", cnonce)
        .addHeader("ctimestamp", ctimestamp)
        .addHeader("cappKey", cappKey)
        .addHeader("csign", csign)
        .post(requestBody)
        .build();
    Call call = httpClient.newCall(request);
    try {
        Response response = call.execute();
        System.out.println("请求结果: " + response.body().string());
        System.out.println(response);
    }
    catch (Exception e) {
        e.printStackTrace();
    }
}
```

- 1. 错误码定义
 - 1.1. 公共错误码
 - 1.2. 人员模块错误码
 - 1.3. 设备模块错误码
 - 1.4. 通行权限模块错误码
 - 1.5. 考勤业务错误码
 - 1.6. 事件业务错误码

1. 错误码定义

1.1. 公共错误码

错误码	描述	备注
100003	参数不合法	
	自定义时, customConfig必传	
	自定义时, blur模糊度为0-1之间的数	
	自定义时, blur模糊度为0-1之间的数	
	自定义时, pitch垂直角度为1-100之间的数	
	自定义时, pitch垂直角度为1-100之间的数	
	自定义时, roll旋转角度为1-100之间的数	
	自定义时, roll旋转角度为1-100之间的数	
	自定义时, yaw水平角度为1-100之间的数	
	自定义时, yaw水平角度为1-100之间的数	
	自定义时, brightnessMin最小平均亮度为1-1000之间的整数	
	自定义时, brightnessMin最小平均亮度为1-1000之间的整数	
	自定义时, brightnessMin最小平均亮度为1-1000之间的整数	
	自定义时, size最小人脸为1-500之间的整数	
	自定义时, size最小人脸为1-500之间的整数	
	自定义时, size最小人脸为1-500之间的整数	
	自定义时, sdtDeviation亮度标准差为1-1000之间的整数	
	自定义时, sdtDeviation最小人脸为1-500之间的整数	
	自定义时, adtDeviation亮度标准差为1-1000之间的整数	
	自定义时, brightnessMax最大平均亮度为1-1000之间的整数	
	自定义时, brightnessMax最大平均亮度为1-1000之间的整数	
	自定义时, brightnessMax最大平均亮度为1-1000之间的整数	
100008	签名错误	
100012	获取配置信息失败	

1.2. 人员模块错误码

错误码	描述	备注
101003	feature删除失败	
101005	机构不存在	
101006	数据插入失败	
101007	手机号已存在	
101008	证件号已存在	
101009	邮箱已存在	
101010	卡号已存在	
101011	员工编号已存在	
101012	该人员不存在	
101013	删除人员失败	
101019	生成二维码失败	
101020	访客访问时间不存在	
101027	邮箱格式错误	
101039	读取文件出错误	
101040	imageuri为空	
101050	模糊度不通过	
101051	人脸大小不通过	
101052	垂直角度不通过	
101053	水平角度不通过	
101054	旋转角度不通过	
101101	面部存在遮挡	
101055	亮度不通过	
101056	标准差亮度不通过	
101066	图片格式或名称错误	
101067	人员数量超过最大值	
101069	访客被拜访员工不能为空	
101070	人脸分析错误，可能是图片人脸质量较差或者没有人脸	
101084	区域不存在	
101081	该设备已进行布控	

错误码	描述	备注
101082	图片大小超过10MB	
101090	唯一标识已存在	
101091	规则名称已存在	
101093	人员关系增加失败	
101096	规则不存在	
101099	UUID已存在	
101104	人脸分析错误，可能是图片中存在多个人脸	
101105	人员uuid类型不符	

1.3. 设备模块错误码

错误码	描述	备注
102001	设备添加失败	
102002	设备修改失败	
102003	不可删除	
102003	设备强制删除失败	
102003	离线删除失败	
102003	设备删除失败	
102005	根据主键id查询记录失败	
102007	门禁点添加失败	
102008	deviceSet删除失败	
102010	根据条件查询deviceSet记录失败	
102011	根据条件关联查询设备记录失败	
102011	根据条件关联查询设备记录失败	
102013	当前请求设备不存在	
102018	deviceSetGroupScheduleRelation关系表查询集合失败	
102023	删除设备配置失败	
102024	修改设备配置失败	
102026	根据deviceid进行批量修改失败	
102027	根据deviceid进行批量新增失败	
102030	根据companyId查询设备数量失败	
102040	获取设备版本详情失败	
102042	设备安装包不存在	
102044	发送MQ消息失败	
102044	下发mq消息失败	
102044	操作命令不存在	
102047	根据snCode查询设备信息失败	
102048	设备唯一标识deviceToken不存在	
102059	设备型号错误	
102060	该设备类型无法进行绑定	

错误码	描述	备注
102061	设备有布控，警戒布控/结构化布控/门禁点布控/事件联动布控/重点人员布控/车辆布控/陌生人员布控	
102062	设备分组查询失败	
102063	无设备可绑	
102065	关联查询存储设备信息失败	
102066	设备解绑失败	
102070	结构化设备添加布控失败	
102076	设备类型和型号不匹配	
102077	设备模式非空	
102077	设备模式和型号不匹配	
102082	门禁点移除失败	
102083	查询设备绑定状态失败	
102085	设备删除预校验失败	
102087	设备名称已存在	
102088	IP地址已存在	
102089	Mac地址已存在	
102090	视频流地址已存在	
102091	设备离线状态	
102092	设备布控查询类型失败	
102098	监控视频流总数超限	
102099	智能相机布控总数超限	
102104	设备apeNum已存在	
102105	sn码无效	
102107	超过企业最大设备接入数,添加设备失败	
102108	设备接入必须在线	
102113	设备不存在	
102114	core能绑定人脸识别的总数超过限制	
102115	core能绑定的人脸抓拍机总数超过限制	
102116	core能绑定的结构化总数超过限制	

错误码	描述	备注
102117	core能绑定的警戒总数超过限制	
102118	设备绑定部分失败	
102119	设备已布控不能修改	
102124	设备管控查询失败	
102125	设备管控移除失败	
102136	ip地址不能为空	
102137	IP规则0.0.0.0~255.255.255.255之间	
102138	用户名由中文、字母和数字组成	
102139	用户名长度1-50位	
102141	云台控制调用失败	
102147	中心算力设备不存在	
102148	中心算力设备离线	
102149	添加人员管控的设备中，存在类型不合法	
102150	默认中心算力不允许修改	
102152	当前设备不存在或者没有进行绑定	

1.4. 通行权限模块错误码

错误码	描述	备注
103001	新增Group失败	
103003	新增GroupPersonRelation失败	
103004	参数不能为空	
103100	数据错误	
103101	参数错误	
103106	时间计划不能超过128条	
103107	通行时间段不能超过3条	
103108	节假日不能超过32条	
103110	组信息异常，核实数据是否被异常删除	
103112	工作日的周计划最多设置7天	
103113	工作日的周计划不允许为空	
103115	该时间计划不允许被删除	
103116	时间计划名称不允许重复	
103117	存在员工没有分配部门	
103118	存在访客没有设置访问对象	
103123	无法删除默认组	
103124	组名重复	
103125	默认组不允许修改	
103126	组不存在	
103133	本类组数目达到最大值，无法添加	
103134	组名称长度应为2~32	
103136	时间计划不存在	
103137	默认时间计划名称不允许修改	
103142	非重点人员组	
103143	B3R通行权限最多支持64个组	
103144	绑定B3R的设备通行权限最多支持20个组	
103145	面板机通行权限最多支持128个组	
103146	人组关系删除发送mq消息异常	
103149	时间计划uuid不允许重复	

错误码	描述	备注
103150	组uuid不允许重复	
103151	B3R最多支持人数是10万	
103152	CORE最多支持人数是100万	
103153	W4K面板机最多支持人数是5万	
103154	W5K面板机最多支持人数是10万	
103156	设备不在人员管控设备中	
103157	设备不在门禁设备中	
103162	添加失败，未指定中心设备	

1.5. 考勤业务错误码

错误码	描述	备注
104034	参数无效	
104001	请选择要添加的考勤设备	
104007	考勤记录不存在	
104008	考勤时段不存在	
104010	人员信息不存在	
104011	考勤组不存在	
104013	未设置考勤时段	
104014	考勤时段编号设置错误	
104015	最多允许设置3个考勤时段	
104016	跨天时段不能超过次日12时	
104017	考勤时段累计不能超过24小时	
104018	考勤时段间隔必须大于1分钟	
104019	考勤时段的起始时间不能晚于结束时间	
104020	考勤时段不支持重合	
104021	最多可设置32个特殊日期	
104022	特殊日期不能重复	
104023	特殊日期不能为空	
104024	不存在跨天时段，不允许设置跨天统计	
104025	只允许一个时间段跨天	
104026	考勤周计划重复	
104027	设备不存在	
104029	出勤调整失败，不符合调整规定	
104033	权限不足	
104036	重复添加！	
104037	本月5号前只能调整上一个自然月及其以后的考勤记录	
104038	本月5号以后只能调整本月及本月以后的考勤记录	
104039	不支持服务器类设备的添加	
104040	名字包含非法字符	
104044	考勤周期枚举类型设置错误，请检查！	

错误码	描述	备注
104045	非门禁点设备	
104047	考勤调整参数无区域	
104048	参数isPunch非法且仅为0或者1	

1.6. 事件业务错误码

错误码	描述	备注
106002	参数错误:s%	

- 1. 数据字典
 - 1.1. 系统安全参数配置
 - 1.1.1. 人脸下发方式(faceConfigStatus)
 - 1.2. 人员
 - 1.2.1. 人员类型(personType)
 - 1.2.2. 性别(sex,gender)
 - 1.2.3. 访客类型(visitType)
 - 1.2.4. 访客状态(visitedStatus)
 - 1.2.5. 人脸入库登记等级(level)
 - 1.3. 组管理
 - 1.3.1. 组类型(type)
 - 1.4. 设备
 - 1.4.1. 设备模式(deviceMode)
 - 1.4.2. 设备类型(deviceType)
 - 1.4.3. 设备型号(deviceModel)
 - 1.4.4. 协议类型
 - 1.4.5. 接入模式(discoveryMethod)
 - 1.5. 门禁点管理
 - 1.5.1. 绑定类型(bindType)
 - 1.5.2. 照片显示类型(pictureType)
 - 1.5.3. 报警输入(alarmInput1,alarmInput2)
 - 1.5.4. 报警输出(alarmOutput1)
 - 1.5.5. 通行认证方式(verificationMode,vistorRecordType)
 - 1.5.6. 使用模式(recognitionMode)
 - 1.6. 考勤
 - 1.6.1. 打卡状态(isPunch)
 - 1.6.2. 考勤结果(result)
 - 1.6.3. 出勤调整类型(adjustType)
 - 1.6.4. 考勤状态(status)
 - 1.6.5. 考勤时段编号(order)
 - 1.6.6. 跨天打卡考勤统计标识(daySpan)
 - 1.7. 事件
 - 1.7.1. 事件等级(eventLevelId)
 - 1.7.2. 事件状态(status)
 - 1.7.3. 订阅业务类型(busiType)
 - 1.7.4. 人脸下发方式(faceConfigStatus)

1. 数据字典

1.1. 系统安全参数配置

1.1.1. 人脸下发方式(faceConfigStatus)

枚举值	枚举描述	备注
0	图片下发	默认为“0”
1	特征值下发	

1.2. 人员

1.2.1. 人员类型(personType)

枚举值	枚举描述	备注
1	员工	
2	访客	
3	重点人员	

1.2.2. 性别(sex,gender)

枚举值	枚举描述	备注
0	未知	
1	男	
2	女	

1.2.3. 访客类型(visitType)

枚举值	枚举描述	备注
1	普通访客	
2	VIP访客	

1.2.4. 访客状态(visitedStatus)

枚举值	枚举描述	备注
true	有效	
false	无效	

1.2.5. 人脸入库登记等级(level)

枚举值	枚举描述	备注
1	低	
3	高	
4	自定义	

1.3. 组管理

1.3.1. 组类型(type)

枚举值	枚举描述	备注
1	员工组	
2	访客组	
3	重点人员组	

1.4. 设备

1.4.1. 设备模式(deviceMode)

枚举值	枚举描述	备注		
1	人脸识别模式			
4	人脸抓拍模式			
5	混合模式	B3R类型设备		
12	全模式	B4h类型设备		

1.4.2. 设备类型(deviceType)

枚举值	枚举描述	备注
1	智能摄像头	设备列表: MegEye-C3S-123 MegEye-C4H-141 MegEye-C4H-241 MegEye-C4N-888 MegEye-C4S-222 MegEye-C4S-322 MegEye-C4R-122 MegEye-C4R-222 MegEye-C4R-322 Smart IPC 海康摄像头 大华摄像头 宇视摄像头 科达摄像头 华为摄像头
2	智能面板机	设备列表: MegEye-W5K-I8 MegEye-W4K-E82 MegEye-W4K-E7 MegEye-W4K-I51 MegEye-W4K-I52 MegEye-W4K-I31 SmartEye-W4K-E7 MegEye-W5K-EA MegEye-W5K-EA-D MegEye-W5K-EA-T MegEye-W5K-EA-DT
3	智能服务器	设备列表: MegBox-B3R-412 MegBox-B3R-1012 MegBox-B3R-1012(H1) MegCube-B4H16-311
4	网络摄像机	设备列表: MegEye-C3S-123 MegEye-C4H-141 MegEye-C4H-241 MegEye-C4N-888 MegEye-C4S-222 MegEye-C4S-322 MegEye-C4R-122 MegEye-C4R-222 MegEye-C4R-322 IPC

1.4.3. 设备型号(deviceModel)

枚举值	枚举描述	备注
1	MegEye-W5K-I8	
2	MegEye-W4K-E7	
3	MegEye-W4K-I51	
4	MegEye-C3S-123	
5	MegEye-C4H-141	
6	MegEye-C4H-241	
7	MegEye-C4N-888	
8	MegEye-C4S-222	
9	MegEye-C4S-322	
10	MegEye-C4R-122	
11	MegEye-C4R-222	
12	MegEye-C4R-322	
13	Smart IPC	
21	IPC	
22	MegBox-B3R-412	
23	MegBox-B3R-1012	
24	MegNVR-N4K-001	
25	MegNVR-N4M-001	
26	MegNVR-N8K-281	
27	MegNVR-N8M-281	
28	MegNVR-N8P-281	
29	海康摄像头	
30	大华摄像头	
31	宇视摄像头	
32	科达摄像头	
33	华为摄像头	
34	MegEye-W4K-I52	
36	MegEye-W4K-I31	
37	MegBox-B3R-1012(H1)	
40	MegEye-W5K-EA	
41	MegEye-W5K-EA-D	
42	MegEye-W5K-EA-T	

枚举值	枚举描述	备注
43	MegEye-W5K-EA-DT	
44	MegEye-W4K-E82	
45	SmartEye-W4K-E7	
57	MegCube-B4H16-311	

1.4.4. 协议类型

枚举值	枚举描述	备注
1	rtsp	添加设备时需传递rtsp流地址

1.4.5. 接入模式(discoveryMethod)

枚举值	枚举描述	备注
1	主动上报	
2	页面添加	

1.5. 门禁点管理

1.5.1. 绑定类型(bindType)

枚举值	枚举描述	备注
1	B3R	
3	面板机	

1.5.2. 照片显示类型(pictureType)

枚举值	枚举描述	备注
0	底库照片	
1	抓拍照片	
2	不显示照片	

1.5.3. 报警输入(alarmInput1,alarmInput2)

枚举值	枚举描述	备注
0	不启用	
1	消防信号	
2	报警信号	

1.5.4. 报警输出(alarmOutput1)

枚举值	枚举描述	备注
0	联动门铃提示	
1	黑名单报警联动报警	
2	门磁超时联动报警	
3	报警输入1联动报警	
4	报警输入2联动报警	

1.5.5. 通行认证方式 (verificationMode,vistorRecordType)

枚举值	枚举描述	备注
0	人脸	
1	人脸或刷卡	
2	人脸及刷卡	
3	人脸及密码	

1.5.6. 使用模式(recognitionMode)

枚举值	枚举描述	备注
0	单人	
1	多人	

1.6. 考勤

1.6.1. 打卡状态(isPunch)

枚举值	枚举描述	备注
0	不打卡	考勤组-考勤规则，设置特殊日期时使用。默认为0
1	打卡	考勤组-考勤规则，设置特殊日期时使用。

1.6.2. 考勤结果(result)

枚举值	枚举描述	备注
0	--	--"表示暂无结果
1	正常	
2	缺勤	
3	迟到	
4	早退	
5	迟到早退	
6	漏打卡	
7	出勤调整	

1.6.3. 出勤调整类型(adjustType)

枚举值	枚举描述	备注
1	补卡	
2	请假	
3	调休	
4	调班	

1.6.4. 考勤状态(status)

枚举值	枚举描述	备注
0	--	--"表示暂无结果
1	正常	
2	旷工	
3	出勤调整	
4	异常	

1.6.5. 考勤时段编号(order)

枚举值	枚举描述	备注
1	考勤时段1	考勤组-考勤规则-设置考勤时段使用。
2	考勤时段2	
3	考勤时段3	最多3个考勤时段

1.6.6. 跨天打卡考勤统计标识(daySpan)

枚举值	枚举描述	备注
0	正常打卡统计	考勤打卡跨天的，跨天的考勤记录按跨天的第一天计算，示例：6号跨7号打卡，此时考勤记录的日期为6号，按6号统计考勤
1	跨天打卡统计	考勤打卡跨天时，跨天的考勤记录按跨天的第二天计算，示例：6号跨7号打卡，此时考勤记录的日期为7号，按7号统计考勤

1.7. 事件

1.7.1. 事件等级(eventLevelId)

枚举值	枚举描述	备注
1	高	
2	中	
3	低	

1.7.2. 事件状态(status)

枚举值	枚举描述	备注
1	已处理	
0	未处理	

1.7.3. 订阅业务类型(busiType)

枚举值	枚举描述	备注
5	通行记录	

1.7.4. 人脸下发方式(faceConfigStatus)

通行记录回调

枚举值	枚举描述	备注	
0	图片		
1	特征值		

- [1. 帮助工具](#)
 - [1.1. 图片上传Demo](#)
 - [1.1.1. Java版本](#)

1. 帮助工具

1.1. 图片上传Demo

业务中提供了图片文件上传接口，为了方便第三方业务集成图片文件上传接口，因此提供Demo示例：

1.1.1. Java版本

```

import com.megvii.bbu.common.util.bone.MegviiCommonErrorCode;
import com.megvii.bbu.common.util.bone.MegviiException;
import lombok.extern.slf4j.Slf4j;
import okhttp3.*;
import org.apache.commons.lang3.StringUtils;

import javax.net.ssl.*;
import java.io.File;
import java.net.URLEncoder;
import java.security.cert.CertificateException;
import java.security.cert.X509Certificate;
import java.util.HashMap;
import java.util.Map;
import java.util.concurrent.TimeUnit;

;

@Slf4j
public class OkHttpUtil {

    private static final byte[] LOCKER = new byte[0];
    private static OkHttpUtil instance;
    private static OkHttpUtil httpsInstance;
    private OkHttpClient okHttpClient;

    private OkHttpClient getUnsafeOkHttpClient() {
        try {
            // Create a trust manager that does not validate certificate chain
            TrustAllManager trustAllManager = new TrustAllManager();
            TrustManager[] trustManagers = {trustAllManager};
            // Install the all-trusting trust manager
            final SSLContext sslContext = SSLContext.getInstance("SSL");
            sslContext.init(null, trustManagers, new java.security.SecureRandom());
            // Create an ssl socket factory with our all-trusting manager
            final SSLSocketFactory sslSocketFactory = sslContext.getSocketFactory();

            OkHttpClient.Builder builder = new OkHttpClient.Builder();
            builder.sslSocketFactory(sslSocketFactory, trustAllManager);
            builder.hostnameVerifier(new HostnameVerifier() {

                @Override
                public boolean verify(String hostname, SSLSession session) {
                    return true;
                }
            });

            OkHttpClient okHttpClient = builder.build();
            return okHttpClient;
        } catch (Exception e) {
            throw new RuntimeException(e);
        }
    }

    private OkHttpUtil(boolean https) {
        OkHttpClient.Builder builder;
        if (https) {
            builder = getUnsafeOkHttpClient().newBuilder();
        } else {
            builder = new OkHttpClient.Builder();
        }

        okHttpClient = builder
            // 10秒连接超时
            .connectTimeout(10, TimeUnit.SECONDS)
            // 写入超时

```

```

        .writeTimeout(900, TimeUnit.SECONDS)
        // 读取超时
        .readTimeout(900, TimeUnit.SECONDS).build();
    }

    public static OkHttpUtil getInstance() {
        if (instance == null) {
            synchronized (LOCKER) {
                if (instance == null) {
                    instance = new OkHttpUtil(false);
                }
            }
        }
        return instance;
    }

    /**
     * true -https
     * false -http
     *
     * @param https
     * @return
     */
    public static OkHttpUtil getInstance(boolean https) {
        if (!https) {
            return getInstance();
        }
        if (httpsInstance == null) {
            synchronized (LOCKER) {
                if (httpsInstance == null) {
                    httpsInstance = new OkHttpUtil(https);
                }
            }
        }
        return httpsInstance;
    }

    public String doPostFormMultipart(String targetUrl, String fileUrl, Map<String, String> params) {
        if (isBlankUrl(targetUrl) || isBlankUrl(fileUrl)) {
            return null;
        }
        Request request = getRequestForPostFormMultipart(targetUrl, fileUrl, params);
        return commonRequest(request);
    }

    private Boolean isBlankUrl(String url) {
        if (StringUtils.isBlank(url)) {
            log.info("url is not blank");
            return true;
        } else {
            return false;
        }
    }

    private String commonRequest(Request request) {
        String re = "";
        try {
            Response response = okHttpClient.newCall(request).execute();
            if (response.isSuccessful()) {
                re = response.body().string();
                log.info("request url: {}; response: {}", request.url().toString(), re);
            } else {
                log.info("request failure url: {}; message: {}", request.url().toString(), response.message());
                throw new Exception("code:" + response.code() + "request failure" + "message:" + response.message());
            }
        } catch (Exception e) {
            log.error("request failure url: {}; message: {}", request.url().toString(), e.getMessage());
        }
    }

```

```

        e.printStackTrace();
        log.error("request execute failure", e);
        throw new MegviiException(MegviiCommonErrorCode.SYSTEM_ERROR);
    }
    return re;
}

private Request getRequestForPostFormMultipart(String serverUrl, String fi
    if (headers == null) {
        headers = new HashMap<>();
    }
    File file = new File(fileUrl); // 文件路径(本地)
    String filename = file.getName();
    MultipartBody.Builder builder = new MultipartBody.Builder().setType(Mu
    builder.addFormDataPart("file", filename, RequestBody.create(MediaType
    Request request = new Request.Builder().url(serverUrl).post(builder.bu
        .addHeader("File-Length", String.valueOf(file.length())) .addHe
        .header("Connection", "close").header("Accept-Encoding", "iden
        // 签名
        .header("cappkey", headers.get("cappkey"))
        .header("ctimestamp", headers.get("ctimestamp"))
        .header("cnonce", headers.get("cnonce"))
        .header("csign", headers.get("csign"))
        .build();
    return request;
}

private Request getRequestForGet(String url, HashMap<String, String> param
    Request request = new Request.Builder().url(getUrlStringForGet(url, pa
    return request;
}

private String getUrlStringForGet(String url, HashMap<String, String> para
    StringBuilder urlBuilder = new StringBuilder();
    urlBuilder.append(url);
    urlBuilder.append("?");
    if (params != null && params.size() > 0) {
        for (Map.Entry<String, String> entry : params.entrySet()) {
            try {
                urlBuilder.append("&").append(entry.getKey()).append("=")
                    .append(URLEncoder.encode(entry.getValue(), "UTF-8
            } catch (Exception e) {
                urlBuilder.append("&").append(entry.getKey()).append("=").
            }
        }
    }
    return urlBuilder.toString();
}

public static class TrustAllManager implements X509TrustManager {
    @Override
    public void checkClientTrusted(X509Certificate[] chain, String authTyp
    }

    @Override
    public void checkServerTrusted(X509Certificate[] chain, String authTyp
    }

    @Override
    public X509Certificate[] getAcceptedIssuers() {
        return new X509Certificate[0];
    }
}

public static void main(String[] args) {

```



```
//服务端上传接口，假设鸿图服务器地址为：10.171.16.76，以图片上传接口为例，如下：
String targetUrl = "http://10.171.16.76/v1/api/person/uploadImage";
//本地要上传的文件
String fileUrl = "C:\\Users\\work\\Desktop\\2.jpg";
//签名信息
Map<String, String> files = new HashMap<>();
files.put("cappkey", "appkey1");
files.put("cnonce", "1234344");
files.put("ctimestamp", "1234344");
files.put("csign", "974228f6c7c56bb1320f8501c01aa991");
//http 地址请求；false 表示
String ret = OkHttpUtil.getInstance(false).doPostFormMultipart(targetU
System.out.println(ret);
//https 地址请求；true 表示
String rets = OkHttpUtil.getInstance(true).doPostFormMultipart(targetU
System.out.println(rets);
    }
}
```

- 1. 接口使用流程
 - 1.1. 通行业务
 - 1.2. 考勤业务
 - 1.3. 事件记录回调
 - 1.4. 通行记录回调
 - 1.5. 考勤记录回调
 - 1.6. 业务数据推送

1. 接口使用流程

1.1. 通行业务

序号	步骤	接口	备注
1	新建员工、访客、重点人员	批量添加人员： <code>/v1/api/person/batchAdd</code>	添加人员需要头像的时候通过“上传人员图片”进行上传。
2	新建员工组、访客组、重点人员组	批量添加组： <code>/v1/api/pass/group/batchAdd</code>	
3	添加员工到员工组、访客到访客组、重点人员到重点人员组	编辑组内人员： <code>/v1/api/pass/group/batchUpdatePerson</code>	这个接口只会保留最后一次调用的 <code>personId</code> ，所以如果是新加入到组的话，需要先查询一遍已经存在的 <code>person</code>
4	添加边设备（接入算力服务器）	设备添加： <code>/v1/api/device/add</code>	
5	添加端设备，添加端设备有以下两种类型：		
	1. 视频流、抓拍机的添加	设备添加： <code>/v1/api/device/add</code>	
	2. 面板机的接入	发现设备查询： <code>/v1/api/device/queryReportDevice</code> 设备导入： <code>/v1/api/device/importDevice</code>	
6	绑定端设备到边设备	设备绑定： <code>/v1/api/device/bind</code>	

序号	步骤	接口	备注
7	添加设备到门禁点	添加门禁点: /v1/api/door/add	可以添加的设备, 通过“分页查询设备列表信息”查询
8	添加通行时间	时间计划管理: 可添加, 查询列表等	
9	添加门禁权限	添加通行权限: /v1/api/pass/authority/add	时间计划通过“时间计划列表”查询出来
	运行... 产生记录.....		
10	查看通行记录	分页查询通行记录: /v1/api/pass_record/list	

1.2. 考勤业务

序号	步骤	接口	备注
1	新建员工	批量添加人员： <code>/v1/api/person/batchAdd</code>	
2	新建员工组	批量添加组： <code>/v1/api/pass/group/batchAdd</code>	
3	添加员工到员工组	编辑组内人员： <code>/v1/api/pass/group/batchUpdatePerson</code>	这个接口只会保留最后一次调用的 <code>personId</code> ，所以如果是新加入到组的话，需要先查询一遍已经存在的 <code>person</code>
4	添加边设备 (接入算力服务器)	设备添加： <code>/v1/api/device/add</code>	
5	添加端设备， 添加端设备有以下两种类型：		
	1. 视频流、抓拍机的添加	设备添加： <code>/v1/api/device/add</code>	
	2. 面板机的接入	发现设备查询： <code>/v1/api/device/queryReportDevice</code> 设备导入： <code>/v1/api/device/importDevice</code>	
6	绑定端设备到边设备	设备绑定： <code>/v1/api/device/bind</code>	
7	添加设备到门禁点	添加门禁点： <code>/v1/api/door/add</code>	
8	添加通行时间	时间计划管理：可添加，查询列表等	

序号	步骤	接口	备注
9	添加门禁权限	添加通行权限： <code>/v1/api/pass/authority/add</code>	
10	添加考勤设备	批量添加考勤设备： <code>/v1/api/attendance/device/batchAdd</code>	
11	添加考勤组	创建考勤组： <code>/v1/api/attendance/group/add</code>	
12	选择人员到考勤组	考勤员工分页列表查询： <code>/v1/api/attendance/group/person/list</code>	
	运行... 产生考勤记录...		
13	查看考勤记录	考勤记录分页列表查询： <code>/v1/api/attendance/record/list</code>	

对于产生的考勤结果，我们可以做很多的操作，如：查看考勤详情、考勤调整等，通过“考勤统计管理”和“考勤记录管理”进行处理，具体不做赘述。

1.3. 事件记录回调

序号	步骤	备注
1	用户在【系统管理】-【功能配置】-【事件配置】上配置事件订阅地址	
2	【事件管理】-【事件联动】页面添加规则	联动反馈一栏，必须选中‘联动事件订阅’
3	事件产生	
4	订阅地址接收到数据	参看api接口“事件回调的数据结构”部分

1.4. 通行记录回调

序号	步骤	接口	备注
1	客户自行开发接受数据的接口		例如： http://xxx 或者 https://xxx
2	在一脸通配置接口添加识别记录回调地址	/v1/api/sys/config/saveAFaceConfig	<pre>{ "identifyRecordCallback " http://xxx " } 或者 { "identifyRecordCallback " https://xxx " }</pre>
3	用户接受到的数据类型		参看"通行记录"->"通行记录数据结构"部分

1.5. 考勤记录回调

序号	步骤	接口	备注
1	客户自行开发接受数据的接口		例如： http://xxx 或者 https://xxx
2	在一脸通配置接口添加考勤回调地址	/v1/api/sys/config/saveAFaceConfig	<pre>{ "attendanceRecordCallb " http://xxx " } 或者 { "attendanceRecordCallb " https://xxx " }</pre>
3	用户接受到的数据类型		参看"考勤记录管理"-"考勤回调数据结构"部分

1.6. 业务数据推送

序号	步骤	备注
1	用户调【订阅接口】，订阅想要的	参看api接口“业务数据订阅”部分
2	事件产生	
3	订阅地址接收到数据	参看api接口“业务数据推送的数据结构”部分

- 1. 人员管理
 - 1.1. 上传人员图片
 - 1.1.1. 基本信息
 - 1.1.2. 请求参数
 - 1.1.3. 返回数据
 - 1.1.4. HTTP请求示例
 - 1.1.5. HTTP响应示例
 - 1.2. 上传人员图片（人脸质量判断按照最低标准进行判断）
 - 1.2.1. 基本信息
 - 1.2.2. 请求参数
 - 1.2.3. 返回数据
 - 1.2.4. HTTP请求示例
 - 1.2.5. HTTP响应示例
 - 1.3. 入库质量检测
 - 1.3.1. 基本信息
 - 1.3.2. 请求参数
 - 1.3.3. 返回数据
 - 1.3.4. 质量不过关相关错误码
 - 1.3.5. HTTP请求示例
 - 1.3.6. HTTP响应示例
 - 1.4. 批量添加人员
 - 1.4.1. 基本信息
 - 1.4.2. 请求参数
 - 1.4.3. 返回数据
 - 1.4.4. HTTP请求示例
 - 1.4.5. HTTP响应示例
 - 1.5. 批量删除人员
 - 1.5.1. 基本信息
 - 1.5.2. 请求参数
 - 1.5.3. 返回数据
 - 1.5.4. HTTP请求示例
 - 1.5.5. HTTP响应示例
 - 1.6. 修改人员
 - 1.6.1. 基本信息
 - 1.6.2. 请求参数
 - 1.6.3. 返回数据
 - 1.6.4. HTTP请求示例
 - 1.6.5. HTTP响应示例
 - 1.7. 搜索人员列表
 - 1.7.1. 基本信息
 - 1.7.2. 请求参数
 - 1.7.3. 返回数据
 - 1.7.4. HTTP请求示例
 - 1.7.5. HTTP响应示例
 - 1.8. 批量查询人员详情信息
 - 1.8.1. 基本信息
 - 1.8.2. 请求参数
 - 1.8.3. 返回数据

- 1.8.4. HTTP请求示例
 - 1.8.5. HTTP响应示例
- 1.9. 查询人员详情信息
 - 1.9.1. 基本信息
 - 1.9.2. 请求参数
 - 1.9.3. 返回数据
 - 1.9.4. HTTP请求示例
 - 1.9.5. HTTP响应示例
- 1.10. 获取员工、访客的二维码
 - 1.10.1. 基本信息
 - 1.10.2. 请求参数
 - 1.10.3. 返回数据
 - 1.10.4. HTTP请求示例
 - 1.10.5. HTTP响应示例
- 1.11. 人员uuid自定义开关
 - 1.11.1. 基本信息
 - 1.11.2. 请求参数
 - 1.11.3. 返回数据
 - 1.11.4. HTTP请求示例
 - 1.11.5. HTTP响应示例

1. 人员管理

1.1. 上传人员图片

1.1.1. 基本信息

Path: /v1/api/person/uploadImage

Method: POST

接口描述: 上传人脸图片文件，接口内部包含了人脸质量判断，图片压缩等处理，适用于需要上传人脸的所有场景，使用参考“帮助工具-图片上传”Demo

1.1.2. 请求参数

Headers

参数名称	参数值	是否必须	示例	备注
Content-Type	multipart/form-data	是		

Body

参数名称	参数类型	是否必须	备注
file	file	是	上传人脸图片文件，接口内部包含了人脸质量判断，图片压缩等处理，适用于需要上传人脸的所有场景，最大10m,支持格式:jpg/jpeg/bmp/png/JPG/JPEG/BMP/PNG

1.1.3. 返回数据

名称	类型	是否必须	备注
uri	string	非必须	上传返回的uri

1.1.4. HTTP请求示例

请求 path

```
/v1/api/person/uploadImage
```

请求 body

```
upload file stream (OMITTED)
```

1.1.5. HTTP响应示例

响应 200

```
{
  "code": 0,
  "data": {
    "uri": "_ZzEwMF9mb3JldmVyQnVja2V0_9a00bafda96b4829a9ff6cd470b1c675"
  },
  "msg": "成功"
}
```

1.2. 上传人员图片（人脸质量判断按照最低标准进行判断）

1.2.1. 基本信息

Path: /v1/api/person/config/uploadImage

Method: POST

接口描述：上传人员图片(使用非系统入库参数,适用于1：N图片上传等场景，不建议入库)

1.2.2. 请求参数

Headers

参数名称	参数值	是否必须	示例	备注
Content-Type	multipart/form-data	是		

Body

参数名称	参数类型	是否必须	示例	备注
file	file	是		上传人脸图片文件，接口内部包含了人脸质量判断(按照最低标准进行判断)，图片压缩等处理，适用于需要上传人脸等场景，不建议入库使用, 最大10m,支持格式:jpg/jpeg/bmp/png/JPG/JPEG/BMP/PNG

1.2.3. 返回数据

名称	类型	是否必须	备注
uri	string	非必须	上传返回的uri

1.2.4. HTTP请求示例

请求 path

```
/v1/api/person/config/uploadImage
```

请求 body

```
upload file stream (OMITTED)
```

1.2.5. HTTP响应示例

响应 200

```
{
  "code": 0,
  "data": {
    "uri": "_ZzEwMF9mb3JldmVyQnVja2V0_9a00bafda96b4829a9ff6cd470b1c675"
  },
  "msg": "成功"
}
```

1.3. 入库质量检测

1.3.1. 基本信息

Path: /v1/api/misc/file/qualityJudge

Method: POST

接口描述: 上传人脸图片文件质量检测，接口内部包含了人脸质量判断，不做上传，只是个质量判断。

1.3.2. 请求参数

Headers

参数名称	参数值	是否必须	示例	备注
Content-Type	multipart/form-data	是		

Body

参数名称	参数类型	是否必须	示例	备注
file	file	是		上传人脸图片文件质量检测，接口内部包含了人脸质量判断，不做上传，只是个质量判断，最大10m,支持格式:jpg/jpeg/bmp/png/JPG/JPEG/BMP/PNG

1.3.3. 返回数据

OK

1.3.4. 质量不过关相关错误码

code	msg
101082	图片大小超过10MB
101066	图片格式或名称错误
101050	模糊度不通过
101052	垂直角度不通过
101053	水平角度不通过
101054	旋转角度不通过
101055	亮度不通过
101056	标准差亮度不通过
101051	人脸大小不通过
101101	面部存在遮挡

1.3.5. HTTP请求示例

请求 path

```
/v1/api/misc/file/qualityJudge
```

请求 body

```
upload file stream (OMITTED)
```

1.3.6. HTTP响应示例

响应 200

```
{
  "code": 0,
  "msg": "成功"
}
```

1.4. 批量添加人员

1.4.1. 基本信息

Path: /v1/api/person/batchAdd

Method: POST

接口描述：一次添加多个人员，注意：人员uuid字段默认不支持第三方自定义（默认系统内容生成），若要开启自定义人员uuid需要调用接口“人员uuid自定义开关”开启（‘系统初始化’或‘系统重置’都会变成默认状态-关闭自定义人员uuid）。

1.4.2. 请求参数

Headers

参数名称	参数值	是否必须	示例	备注
Content-Type	application/json	是		

Body

名称	类型	是否必须	备注
personList	object []	必须	用户列表
└─ name	string	必须	用户名, 长度:[1,40]
└─ type	integer	必须	人员类型, 详情请查看数据字典“人员类型”
└─ sex	integer	非必须	性别, 详情请查看数据字典“性别”
└─ uuid	string	非必须	人员唯一标识uuid, 若需自定义uuid需调用uuid自定义接口开启(默认系统内部生成), uuid需保证历史唯一且最大长度32位
└─ imageUri	string	非必须	用户识别照片的uri
└─ code	string	非必须	员工编码,员工可填,若填写需保证唯一, 格式: 允许大小写英文字母,数字, 长度:[1,32]
└─ identifyNum	string	非必须	身份证号, 格式:允许大小写英文字母,数字, 长度:[1,32]
└─ uniqueIdentify	string	非必须	唯一标识, 若填写需要保证唯一, 格式:最大40位数字、字母和ascii码的字符
└─ visitFirm	string	非必须	访客所属单位, 访客可填, 格式: 汉字,大小写英文字母,数字, 长度: [1,40]
└─ visitStartTimeStamp	integer	非必须	拜访起始时间(时间戳, 毫秒), 访客必填
└─ visitEndTimeStamp	integer	非必须	拜访结束时间(时间戳, 毫秒), 访客必填
└─ visitReason	string	非必须	拜访原因, 访客可填,格式: 任意字符, 长度: [1, 255]

名称	类型	是否必须	备注
└─ visitedUuid	string	非必须	受访人的uuid, 访客必填
└─ visitType	integer	非必须	访客类型, 访客可填, 1 普通访客, 2 VIP
└─ email	string	非必须	邮箱, 格式: 符合一般邮箱格式即可
└─ phone	string	非必须	手机号, 格式: 数字, 长度: [6, 18], 访客类型手机号必填, 其他人员类型非必填
└─ cardNum	string	非必须	卡号, 若填写需保证唯一, 格式: 数字, 长度: [1, 20]
└─ password	string	非必须	密码, 员工可填, 格式: 数字, 长度: [4, 6]
└─ entryTimeStamp	integer	非必须	入职时间(时间戳, 毫秒), 员工可填
└─ postion	string	非必须	职位, 员工可填, 格式: 任意字符, 长度: [0, 64]
└─ birthdayStamp	integer	非必须	生日(时间戳, 毫秒)
└─ ext	string	非必须	备注, 格式: 任意字符, 长度: [0, 255]
└─ groupList	string []	非必须	加入的组uuid集合
└─		非必须	
└─ autograph	string	非必须	个性签名

名称	类型	是否必须	备注
└orgUuid	string	非必须	组织uuid

1.4.3. 返回数据

名称	类型	是否必须	备注
successes	object []	非必须	成功列表
└─name	string	非必须	用户名
└─code	integer	非必须	结果状态码
└─type	integer	非必须	人员类型, 详情请查看数据字典“人员类型”
└─msg	string	非必须	结果信息提示
└─sex	integer	非必须	性别, 详情请查看数据字典“性别”
└─uuid	string	非必须	人员唯一标识uuid, 自定义时返回自定义uuid(默认返回系统内部生成的uuid)
└─imageUri	string	非必须	用户识别照片的uri
└─staffCode	string	非必须	员工编码, 员工可填, 若填写需保证唯一, 格式: 允许大小写英文字母, 数字, 长度: [1, 32]
└─identifyNum	string	非必须	身份证号, 格式: 允许大小写英文字母, 数字, 长度: [1, 32]
└─visitStartTimeStamp	integer	非必须	拜访起始时间(时间戳, 毫秒), 访客必填
└─visitEndTimeStamp	integer	非必须	拜访结束时间(时间戳, 毫秒), 访客必填
└─visitReason	string	非必须	拜访原因, 访客可填, 格式: 任意字符, 长度: [1, 255]

名称	类型	是否必须	备注
└─visitedUuid	string	非必须	受访人的uuid, 访客必填
└─visitType	integer	非必须	访客类型, 访客可填, 1 普通访客, 2 VIP
└─email	string	非必须	邮箱, 格式: 符合一般邮箱格式即可
└─phone	string	非必须	手机号, 格式: 数字, 长度: [6,18]
└─cardNum	string	非必须	卡号, 若填写需保证唯一, 格式: 数字, 长度: [1,16]
└─password	string	非必须	密码, 员工可填, 格式: 数字, 长度: [4,6]
└─entryTimeStamp	integer	非必须	入职时间(时间戳, 毫秒), 员工可填
└─postion	string	非必须	职位, 员工可填, 格式: 任意字符, 长度: [1,64]
└─birthdayStamp	integer	非必须	生日(时间戳, 毫秒)
└─ext	string	非必须	备注, 格式: 任意字符, 长度: [1,255]
└─orgUuid	string	非必须	组织uuid
└─groupList	string []	非必须	加入的组uuid集合
└─		非必须	

名称	类型	是否必须	备注
failures	object []	非必须	失败列表
└─name	string	非必须	用户名
└─code	integer	非必须	结果状态码
└─type	integer	非必须	人员类型, 详情请查看数据字典“人员类型”
└─msg	string	非必须	结果信息提示
└─sex	integer	非必须	性别, 详情请查看数据字典“性别”
└─uuid	string	非必须	uuid, 若填写需保证唯一, 最大长度32位
└─imageUri	string	非必须	用户识别照片的uri
└─staffCode	string	非必须	员工编码, 员工可填, 若填写需保证唯一, 格式: 允许大小写英文字母, 数字, 长度: [1, 32]
└─identifyNum	string	非必须	身份证号, 格式: 允许大小写英文字母, 数字, 长度: [1, 32]
└─visitStartTimeStamp	integer	非必须	拜访起始时间(时间戳, 毫秒), 访客必填
└─visitEndTimeStamp	integer	非必须	拜访结束时间(时间戳, 毫秒), 访客必填
└─visitReason	string	非必须	拜访原因, 访客可填, 格式: 任意字符, 长度: [1, 255]

名称	类型	是否必须	备注
└─visitedUuid	string	非必须	受访人的uuid, 访客必填
└─visitType	integer	非必须	访客类型, 访客可填, 1 普通访客, 2 VIP
└─email	string	非必须	邮箱, 格式: 符合一般邮箱格式即可
└─phone	string	非必须	手机号, 格式: 数字, 长度: [6,18]
└─cardNum	string	非必须	卡号, 若填写需保证唯一, 格式: 数字, 长度: [1,16]
└─password	string	非必须	密码, 员工可填, 格式: 数字, 长度: [4,6]
└─entryTimeStamp	integer	非必须	入职时间(时间戳, 毫秒), 员工可填
└─postion	string	非必须	职位, 员工可填, 格式: 任意字符, 长度: [1,64]
└─birthdayStamp	integer	非必须	生日(时间戳, 毫秒)
└─ext	string	非必须	备注, 格式: 任意字符, 长度: [1,255]
└─orgUuid	string	非必须	组织uuid
└─groupList	string []	非必须	加入的组uuid集合

名称	类型	是否必须	备注
—autograph	string	非必须	个性签名

1.4.4. HTTP请求示例

请求 path

```
/v1/api/person/batchAdd
```

请求 body

```
{
  "personList": [
    {
      "name": "我是员工",
      "type": 1,
      "sex": 1,
      "uuid": "ngtwr",
      "imageUri": "_ZzEwMF9mb3JldmVyQnVja2V0_91b2035456f9407ea1b8d944003",
      "code": "877849",
      "identifyNum": "911102",
      "email": "MgpbsXyhik@126.com",
      "phone": "17792842739",
      "cardNum": "220306",
      "password": "255075",
      "entryTimeStamp": 1622535057000,
      "postion": "海龙",
      "birthdayStamp": 1591113600000,
      "uniqueIdentify": "650677",
      "ext": "备注",
      "orgUuid": "507cf621f4db4d0dac14e10baad2d442",
      "groupList": [
        "976055aae31445c5a583ebc1a53c7514"
      ]
    }
  ]
}
```

1.4.5. HTTP响应示例

响应 200

```
{
  "code": 0,
  "data": {
    "failures": [],
    "successes": [
      {
        "birthdayStamp": 1591113600000,
        "cardNum": "220306",
        "email": "MgpbsXyhik@126.com",
        "entryTimeStamp": 1622535057000,
        "ext": "备注",
        "groupList": [
          "976055aae31445c5a583ebc1a53c7514"
        ],
        "identifyNum": "911102",
        "imageUri": "_ZzEwMF9mb3JldmVyQnVja2V0_91b2035456f9407ea1b8d94",
        "name": "我是员工",
        "password": "255075",
        "phone": "17792842739",
        "postion": "海龙",
        "sex": 1,
        "staffCode": "877849",
        "type": 1,
        "uuid": "ngtwr",
        "orgUuid": "507cf621f4db4d0dac14e10baad2d442"
      }
    ]
  },
  "msg": "成功"
}
```

1.5. 批量删除人员

1.5.1. 基本信息

Path: /v1/api/person/batchDelete

Method: POST

接口描述: 一次删除多个人员

1.5.2. 请求参数

Headers

参数名称	参数值	是否必须	示例	备注
Content-Type	application/json	是		

Body

名称	类型	是否必须	备注
uuidList	string []	必须	用户UUID列表
└─		非必须	

1.5.3. 返回数据

名称	类型	是否必须	备注
successes	object []	非必须	成功列表
├─uuid	string	非必须	uuid
├─code	integer	非必须	结果状态码
├─msg	string	非必须	结果信息提示
failures	object []	非必须	失败列表
├─uuid	string	非必须	uuid
├─code	integer	非必须	结果状态码
├─msg	string	非必须	结果信息提示

1.5.4. HTTP请求示例

请求 path

```
/v1/api/person/batchDelete
```

请求 body

```
{
  "uuidList": [
    "7e6916df6a05444692b0d48621892551",
    "d3eef33925544fac9c83f650be8e727b"
  ]
}
```

1.5.5. HTTP响应示例

响应 200

```
{
  "code": 0,
  "data": {
    "failures": [],
    "successes": [
      {
        "uuid": "7e6916df6a05444692b0d48621892551"
      },
      {
        "uuid": "d3eef33925544fac9c83f650be8e727b"
      }
    ]
  },
  "msg": "成功"
}
```

1.6. 修改人员

1.6.1. 基本信息

Path: /v1/api/person/update

Method: POST

接口描述: 修改人员信息

1.6.2. 请求参数

Headers

参数名称	参数值	是否必须	示例	备注
Content-Type	application/json	是		

Body

名称	类型	是否必须	备注
uuid	string	必须	用户UUID
name	string	非必须	用户名, 长度:[1,40]
sex	integer	非必须	性别, 详情请查看数据字典“性别”
imageUri	string	非必须	用户识别照片的uri
code	string	非必须	员工编码,员工可填,若填写需保证唯一, 格式: 允许大小写英文字母,数字, 长度:[1,32]
identifyNum	string	非必须	身份证号, 格式:允许大小写英文字母,数字, 长度:[1,32]
uniqueIdentify	string	非必须	唯一标识, 若填写需要保证唯一, 格式:最大40位数字、字母和ascii码的字符
visitFirm	string	非必须	访客所属单位, 访客可填, 格式: 汉字,大小写英文字母,数字, 长度: [1,40]
visitStartTimeStamp	integer	非必须	拜访起始时间(时间戳,毫秒), 访客必填
visitEndTimeStamp	integer	非必须	拜访结束时间(时间戳,毫秒), 访客必填
visitReason	string	非必须	拜访原因, 访客可填,格式: 任意字符, 长度: [1, 255]
visitedUuid	string	非必须	受访人的uuid, 访客必填
visitType	integer	非必须	访客类型, 访客可填, 1 普通访客, 2 VIP

名称	类型	是否必须	备注
email	string	非必须	邮箱, 格式: 符合一般邮箱格式即可
phone	string	非必须	手机号, 格式: 数字, 长度: [6, 18], 访客类型手机号必填, 其他人员类型非必填
cardNum	string	非必须	卡号, 员工可填, 若填写需保证唯一, 格式: 数字, 长度: [1, 20]
password	string	非必须	密码, 员工可填, 格式: 数字, 长度: [4, 6]
entryTimeStamp	integer	非必须	入职时间(时间戳, 毫秒), 员工可填
postion	string	非必须	职位, 员工可填, 格式: 任意字符, 长度: [0, 64]
birthdayStamp	integer	非必须	生日(时间戳, 毫秒)
ext	string	非必须	备注, 格式: 任意字符, 长度: [0, 255]
groupList	string []	非必须	加入的组集合
└		非必须	
autograph	string	非必须	个性签名
orgUuid	string	非必须	组织uuid

1.6.3. 返回数据

OK

1.6.4. HTTP请求示例

请求 path

```
/v1/api/person/update
```

请求 body

```
{
  "uuid": "363227b5360f4a29b28d0a9543c14914",
  "name": "我是update员工",
  "type": 1,
  "sex": 1,
  "imageUri": "_ZzEwMF9mb3JldmVyQnVja2V0_2fb0ce763b5c4cb29a7d64af407a2f43",
  "code": "910025",
  "identifyNum": "185864",
  "email": "OpMJvwJc62@126.com",
  "phone": "13403743852",
  "cardNum": "247870",
  "password": "451548",
  "entryTimeStamp": 1622535055000,
  "postion": "海龙",
  "birthdayStamp": 1591113600000,
  "uniqueIdentify": "992514",
  "ext": "备注",
  "groupList": [
    "976055aee31445c5a583ebc1a53c7514"
  ]
}
```

1.6.5. HTTP响应示例

响应 200

```
{
  "code": 0,
  "msg": "成功"
}
```

1.7. 搜索人员列表

1.7.1. 基本信息

Path: /v1/api/person/list

Method: POST

接口描述: 搜索人员, 下列字段支持模糊搜索: name、phone、identifyNum、email、code、cardNum

1.7.2. 请求参数

Headers

参数名称	参数值	是否必须	示例	备注
Content-Type	application/json	是		

Body

名称	类型	是否必须	备注
pageNum	integer	非必须	当前页码,不传默认1
pageSize	integer	非必须	分页大小, 不传默认20
type	integer	非必须	人员类型, 详情请查看数据字典“人员类型”
name	string	非必须	用户姓名
phone	string	非必须	手机号
identifyNum	string	非必须	身份证号
email	string	非必须	邮箱
code	string	非必须	员工编号
cardNum	string	非必须	卡号
uniqueIdentify	string	非必须	唯一标识
staffEntryMinStamp	integer	非必须	员工入职日期最小值(时间戳,毫秒)
staffEntryMaxStamp	integer	非必须	员工入职日期最大值(时间戳,毫秒)
visitorCreateMinStamp	integer	非必须	访客创建日期最小值(时间戳,毫秒)

名称	类型	是否必须	备注
visitorCreateMaxStamp	integer	非必须	访客创建日期最大值(时间戳,毫秒)
filterVisitedType	integer	非必须	筛选访客状态类型, 0 不筛选, 1 有效, 2 无效
simpleInfo	boolean	非必须	是否仅返回简单信息(id+name), 默认false
isPinyinOrder	boolean	非必须	是否按拼音排序, 默认true
visitType	integer	非必须	访客类型, 详情请查看数据字典“访客类型”
visitedName	string	非必须	受访员工姓名
orgUuid	string	非必须	组织uuid

1.7.3. 返回数据

名称	类型	是否必须	备注
pageNum	integer	非必须	当前页码
pageSize	integer	非必须	分页大小
total	integer	非必须	总数
list	object []	非必须	
└─uuid	string	非必须	用户UUID
└─name	string	非必须	用户名
└─type	integer	非必须	人员类型, 详情请查看数据字典“人员类型”
└─sex	integer	非必须	性别, 详情请查看数据字典“性别”
└─imageUri	string	非必须	用户识别照片的url
└─code	string	非必须	员工编码, 员工非必须, 唯一
└─identifyNum	string	非必须	身份证号
└─visitStartTime	string	非必须	拜访起始时间(时间戳, 毫秒), 访客必须
└─visitEndTime	string	非必须	拜访结束时间(时间戳, 毫秒), 访客必须

名称	类型	是否必须	备注
└─visitReason	string	非必须	拜访原因, 访客非必须
└─visitedUuid	string	非必须	受访人的UUID, 访客必须
└─visitedName	string	非必须	受访人的姓名, 访客必须
└─visitType	integer	非必须	访客类型, 访客必须, 1 普通访客, 2 VIP
└─visitedStatus	boolean	非必须	访客状态, 详情请查看数据字典“访客状态”
└─email	string	非必须	邮箱
└─phone	string	非必须	手机号
└─cardNum	string	非必须	卡号, 员工非必须, 唯一
└─password	string	非必须	密码, 员工非必须
└─entryTime	string	非必须	入职时间(时间戳, 毫秒), 员工非必须
└─postion	string	非必须	职位, 员工非必须
└─birthday	string	非必须	生日(时间戳, 毫秒), 员工非必须
└─uniqueIdentify	string	非必须	唯一标识, 唯一

名称	类型	是否必须	备注
ext	string	非必须	备注
groupList	object []	非必须	人员所属组列表
uuid	string	非必须	组uuid
name	string	非必须	组名
visitFirm	string	非必须	访客所属单位, 访客非必须
isPassRecord	boolean	非必须	是否有通行记录
isAttendanceRecord	boolean	非必须	是否有考勤记录
autograph	string	非必须	个性签名
orgUuid	string	非必须	组织uuid
orgName	string	非必须	组织名称

1.7.4. HTTP请求示例

请求 path

```
/v1/api/person/list
```

请求 body

```
{
  "type": 1,
  "name": "备用员工1"
}
```

1.7.5. HTTP响应示例

响应 200

```
{
  "code": 0,
  "data": {
    "list": [
      {
        "code": "13925810",
        "ext": "",
        "groupList": [
          {
            "name": "员工默认组",
            "uuid": "976055aae31445c5a583ebc1a53c7514"
          }
        ],
        "identifyNum": "1234",
        "isAttendanceRecord": false,
        "isPassRecord": false,
        "name": "123456",
        "password": "123456",
        "postion": "",
        "sex": 0,
        "type": 1,
        "uuid": "950d737b2051410986698adb493965d5"
      }
    ],
    "pageNum": 1,
    "pageSize": 20,
    "total": 1
  },
  "msg": "成功"
}
```

1.8. 批量查询人员详情信息

1.8.1. 基本信息

Path: /v1/api/person/uuid/list

Method: POST

接口描述: 通过uuid列表一次获取多个人员信息

1.8.2. 请求参数

Headers

参数名称	参数值	是否必须	示例	备注
Content-Type	application/json	是		

Body

名称	类型	是否必须	备注
uuidList	string []	必须	人员UUID列表
└			

1.8.3. 返回数据

名称	类型	是否必须	备注
	object []	非必须	
└─uuid	string	非必须	用户UUID
└─name	string	非必须	用户名
└─type	integer	非必须	人员类型, 详情请查看数据字典“人员类型”
└─sex	integer	非必须	性别, 详情请查看数据字典“性别”
└─imageUri	string	非必须	用户识别照片的url
└─code	string	非必须	员工编码,员工非必须,唯一
└─identifyNum	string	非必须	身份证号
└─visitStartTime	string	非必须	拜访起始时间(时间戳,毫秒), 访客必须
└─visitEndTime	string	非必须	拜访结束时间(时间戳,毫秒), 访客必须
└─visitReason	string	非必须	拜访原因, 访客非必须
└─visitedUuid	string	非必须	受访人的UUID,访客必须
└─visitedName	string	非必须	受访人的姓名, 访客必须
└─visitType	integer	非必须	访客类型, 访客必须, 1 普通访客, 2 VIP
└─visitedStatus	boolean	非必须	访客状态, 详情请查看数据字典“访客状态”
└─email	string	非必须	邮箱
└─phone	string	非必须	手机号
└─cardNum	string	非必须	卡号,员工非必须, 唯一

名称	类型	是否必须	备注
└password	string	非必须	密码, 员工非必须
└entryTime	string	非必须	入职时间(时间戳, 毫秒), 员工非必须
└postion	string	非必须	职位, 员工非必须
└birthday	string	非必须	生日(时间戳, 毫秒), 员工非必须
└uniqueIdentify	string	非必须	唯一标识, 唯一
└ext	string	非必须	备注
└groupList	object []	非必须	人员所属组列表
└uuid	string	非必须	组uuid
└name	string	非必须	组名
└visitFirm	string	非必须	访客所属单位, 访客非必须
└autograph	string	非必须	个性签名, 非必须
└orgUuid	string	非必须	组织uuid
└orgName	string	非必须	组织名称

1.8.4. HTTP请求示例

请求 path

```
/v1/api/person/uuid/list
```

请求 body

```
{
  "uuidList": [
    "70d6d5e509334d89b97626ffafcd8a5"
  ]
}
```

1.8.5. HTTP响应示例

响应 200

```
{
  "code": 0,
  "data": [
    {
      "birthday": 1591113600000,
      "cardNum": "622736",
      "code": "436373",
      "email": "wvUIB2KMN6@126.com",
      "entryTime": 1626204061000,
      "ext": "备注",
      "groupList": [
        {
          "name": "员工默认组",
          "uuid": "976055aae31445c5a583ebc1a53c7514"
        }
      ],
      "identifyNum": "119398",
      "imageUri": "http://10.171.4.233/pub/_Y2FjdHVzMTAwX2ZvcnV2ZXJCdWNR",
      "name": "备用员工01",
      "phone": "13412431439",
      "postion": "海龙",
      "sex": 1,
      "type": 1,
      "uuid": "70d6d5e509334d89b97626ffafcd8a5"
    }
  ],
  "msg": "成功"
}
```

1.9. 查询人员详情信息

1.9.1. 基本信息

Path： /v1/api/person/query

Method： POST

接口描述：通过uuid获取人员信息

1.9.2. 请求参数

Headers

参数名称	参数值	是否必须	示例	备注
Content-Type	application/json	是		

Body

名称	类型	是否必须	备注
uuid	string	必须	人员uuid

1.9.3. 返回数据

名称	类型	是否必须	备注
uuid	string	必须	用户UUID
name	string	必须	用户名
type	integer	必须	人员类型, 详情请查看数据字典“人员类型”
sex	integer	非必须	性别, 详情请查看数据字典“性别”
imageUri	string	非必须	用户识别照片的url
code	string	非必须	员工编码, 员工非必须, 唯一
identifyNum	string	非必须	身份证号
visitStartTime	string	非必须	拜访起始时间(时间戳, 毫秒), 访客必须
visitEndTime	string	非必须	拜访结束时间(时间戳, 毫秒), 访客必须
visitReason	string	非必须	拜访原因, 访客非必须
visitedUuid	string	非必须	受访人的UUID, 访客必须
visitedName	string	非必须	受访人的姓名, 访客必须
visitType	integer	非必须	访客类型, 访客必须, 1 普通访客, 2 VIP
visitedStatus	boolean	非必须	访客状态, 详情请查看数据字典“访客状态”
email	string	非必须	邮箱
phone	string	非必须	手机号
cardNum	string	非必须	卡号, 员工非必须, 唯一
password	string	非必须	密码, 员工非必须
entryTime	string	非必须	入职时间(时间戳, 毫秒), 员工非必须

名称	类型	是否必须	备注
postion	string	非必须	职位, 员工非必须
birthday	string	非必须	生日(时间戳,毫秒), 员工非必须
uniqueIdentify	string	非必须	唯一标识, 唯一
ext	string	非必须	备注
groupList	object []	非必须	人员所属组列表
└─ uuid	string	非必须	组uuid
└─ name	string	非必须	组名
visitFirm	string	非必须	访客所属单位, 访客非必须
autograph	string	非必须	个性签名
orgUuid	string	非必须	组织uuid
orgName	string	非必须	组织名称

1.9.4. HTTP请求示例

请求 path

```
/v1/api/person/query
```

请求 body

```
{
  "uuid": "a826d9004fd0430497a4f49aaadf96c4"
}
```

1.9.5. HTTP响应示例

响应 200

```

{
  "code": 0,
  "data": {
    "birthday": 1591113600000,
    "cardNum": "554263",
    "code": "265699",
    "email": "trktPk4GQz@126.com",
    "entryTime": 1626204046000,
    "ext": "备注",
    "groupList": [
      {
        "name": "员工默认组",
        "uuid": "976055aae31445c5a583ebc1a53c7514"
      }
    ],
    "identifyNum": "091751",
    "imageUri": "http://10.171.4.233/pub/_Y2FjdHVzMTAwX2ZvcmlV2ZXJCdWlrZXQ=",
    "name": "备用员工01",
    "phone": "13685185710",
    "postion": "海龙",
    "sex": 1,
    "type": 1,
    "uuid": "a826d9004fd0430497a4f49aaadf96c4"
  },
  "msg": "成功"
}

```

1.10. 获取员工、访客的二维码

1.10.1. 基本信息

Path: /v1/api/person/visitorCode

Method: POST

接口描述: 员工、访客获取二维码使用。

1.10.2. 请求参数

Headers

参数名称	参数值	是否必须	示例	备注
Content-Type	application/json	是		

Body

名称	类型	是否必须	备注
uuid	string	必须	员工、访客uuid

1.10.3. 返回数据

名称	类型	是否必须	备注
qrCode	string	非必须	员工、访客二维码图片(base64编码的字符串)
expireTime	integer	非必须	二维码图片的过期分钟数

1.10.4. HTTP请求示例

请求 path

```
/v1/api/person/visitorCode
```

请求 body

```
{
  "uuid": "72ec703308e840cfb05287ac9f268be5"
}
```

1.10.5. HTTP响应示例

响应 200

```
{
  "code": 0,
  "data": {
    "expireTime": 10,
    "qrCode": "data:image/png;base64,R0lGODlhYADIAPAAAAAAP///ywAAAAyADIAE."
  },
  "msg": "成功"
}
```

1.11. 人员uuid自定义开关

1.11.1. 基本信息

Path: /v1/api/person/personUuidSwitch

Method: GET

接口描述: 是否开启自定义人员uuid字段的控制开关，默认关闭、不支持第三方自定义人员uuid（默认系统内部生成），详情查看接口“批量添加人员”。

1.11.2. 请求参数

无

1.11.3. 返回数据

名称	类型	是否必须	备注
personUuidSwitch	integer	必须	0:关闭，不支持用户自定义uuid；1:开启，支持用户自定义uuid
personUuidSwitchMsg	string	必须	人员uuid自定义是否开启消息

1.11.4. HTTP请求示例

请求 path

```
/v1/api/person/personUuidSwitch
```

请求参数

无

1.11.5. HTTP响应示例

响应 200

```
{
  "code": 0,
  "data": {
    "personUuidSwitch": 1,
    "personUuidSwitchMsg": "已开启人员uuid自定义，支持第三方自定义传入"
  },
  "msg": "成功"
}
```

或

```
{
  "code": 0,
  "data": {
    "personUuidSwitch": 0,
    "personUuidSwitchMsg": "已关闭人员uuid自定义，此时uuid系统内部生成"
  },
  "msg": "成功"
}
```


- 1. 组织管理
 - 1.1. 添加组织
 - 1.1.1. 基本信息
 - 1.1.2. 请求参数
 - 1.1.3. 返回数据
 - 1.1.4. HTTP请求示例
 - 1.1.5. HTTP响应示例
 - 1.2. 修改组织信息
 - 1.2.1. 基本信息
 - 1.2.2. 请求参数
 - 1.2.3. 返回数据
 - 1.2.4. HTTP请求示例
 - 1.2.5. HTTP响应示例
 - 1.3. 查询组织列表
 - 1.3.1. 基本信息
 - 1.3.2. 请求参数
 - 1.3.3. 返回数据
 - 1.3.4. HTTP请求示例
 - 1.3.5. HTTP响应示例
 - 1.4. 删除组织
 - 1.4.1. 基本信息
 - 1.4.2. 请求参数
 - 1.4.3. 返回数据
 - 1.4.4. HTTP请求示例
 - 1.4.5. HTTP响应示例
 - 1.5. 获取最上级组织信息
 - 1.5.1. 基本信息
 - 1.5.2. 请求参数
 - 1.5.3. 返回数据
 - 1.5.4. HTTP请求示例
 - 1.5.5. HTTP响应示例

1. 组织管理

1.1. 添加组织

1.1.1. 基本信息

Path: /v1/api/org/add

Method: POST

接口描述: 添加组织接口

1.1.2. 请求参数

Headers

参数名称	参数值	是否必须	示例	备注
Content-Type	application/json	是		无

Body

名称	类型	是否必须	备注
fullName	string	必须	名称（长度为0-16位）
parentUuid	string	必须	父机构uuid

1.1.3. 返回数据

名称	类型	是否必须	备注
uuid	string	非必须	UUID
fullName	string	非必须	名称
parentUuid	string	非必须	父机构uuid

1.1.4. HTTP请求示例**请求 path**

```
/v1/api/org/add
```

请求 body

```
{
  "fullName": "二级部门",
  "parentUuid": "5136133420854963ad5c939f6f10026e"
}
```

1.1.5. HTTP响应示例**响应 200**

```
{
  "code": 0,
  "data": {
    "uuid": "7d3ebca932cf4672bca66cc98f233cb1",
    "fullName": "二级部门",
    "parentUuid": "5136133420854963ad5c939f6f10026e",
  },
  "msg": "成功"
}
```

1.2. 修改组织信息

1.2.1. 基本信息

Path: /v1/api/org/update

Method: POST

接口描述: 修改部门信息，只允许修改名称，不支持调整部门层级；

1.2.2. 请求参数

Headers

参数名称	参数值	是否必须	示例	备注
Content-Type	application/json	是		无

Body

名称	类型	是否必须	备注
uuid	string	必须	组织或公司uuid
fullName	string	非必须	名称

1.2.3. 返回数据

OK

1.2.4. HTTP请求示例

请求 path

```
/v1/api/org/update
```

请求 body

```
{
  "fullName": "二级部门",
  "uuid": "5136133420854963ad5c939f6f10026e"
}
```

1.2.5. HTTP响应示例

响应 200

```
{
  "code": 0,
  "msg": "成功"
}
```

1.3. 查询组织列表

1.3.1. 基本信息

Path: /v1/api/org/tree

Method: POST

接口描述: 查询组织列表信息，支持根据部门名称检索；

1.3.2. 请求参数

Headers

参数名称	参数值	是否必须	示例	备注
Content-Type	application/json	是		无

Body

名称	类型	是否必须	备注
orgName	string	非必须	机构名

1.3.3. 返回数据

名称	类型	是否必须	备注
	object []	非必须	组织列表
 orgUuid	string	非必须	组织uuid
 orgName	string	非必须	组织名称
 parentUuid	string	非必须	父组织uuid
 childNum	integer	非必须	组织下人员数量(仅为该组织,不包括其子组织)
 childTotal	integer	非必须	组织下人员数量(包括其下所有子组织)

1.3.4. HTTP请求示例

请求 path

```
/v1/api/org/tree
```

请求 body

```
{}
```

1.3.5. HTTP响应示例

响应 200

```
{
  "code":0,
  "data":[
    {
      "childNum":18,
      "childTotal":28,
      "orgUuid":"5136133420854963ad5c939f6f10026e",
      "orgName":"一级区域"
    },
    {
      "childNum":4,
      "childTotal":4,
      "orgUuid":"12bed310d562435fbecfd97e02d69a31",
      "orgName":"二级区域",
      "parentUuid":"5136133420854963ad5c939f6f10026e"
    },
    {
      "childNum":6,
      "childTotal":6,
      "orgUuid":"5486ac8a8e16476ab7eaa0d6a56a399e",
      "orgName":"test",
      "parentUuid":"5136133420854963ad5c939f6f10026e"
    }
  ],
  "msg":"成功"
}
```

1.4. 删除组织

1.4.1. 基本信息

Path： /v1/api/org/delete

Method： POST

接口描述： 删除组织接口

1.4.2. 请求参数

Headers

参数名称	参数值	是否必须	示例	备注
Content-Type	application/json	是		无

Body

名称	类型	是否必须	备注
uuid	string	必须	组织uuid
personInOrgHandleMode	integer	非必须	部门内人员处理模式，0表示删除，1表示移动到父部门内 默认为0

1.4.3. 返回数据

OK

1.4.4. HTTP请求示例

请求 path

/v1/api/org/delete

请求 body

```
{
  "uuid": "507cf621f4db4d0dac14e10baad2d442",
  "personInOrgHandleMode": 1
}
```

1.4.5. HTTP响应示例

响应 200

```
{
  "code": 0,
  "msg": "成功"
}
```

1.5. 获取最上级组织信息

1.5.1. 基本信息

Path: /v1/api/org/top/detail

Method: POST

接口描述: 获取一级（最上级）组织信息接口；

1.5.2. 请求参数

Headers

参数名称	参数值	是否必须	示例	备注
Content-Type	application/json	是		无

Body

无

1.5.3. 返回数据

名称	类型	是否必须	备注
orgUuid	string	非必须	组织uuid
orgName	string	非必须	组织名称

1.5.4. HTTP请求示例

请求 path

```
/v1/api/org/top/detail
```

请求 body

```
{}
```

1.5.5. HTTP响应示例

响应 200

```
{
  "code": 0,
  "data": {
    "orgUuid": "5136133420854963ad5c939f6f10026e",
    "orgName": "一级组织",
  },
  "msg": "成功"
}
```

- 1. 组管理
 - 1.1. 批量添加人员组
 - 1.1.1. 基本信息
 - 1.1.2. 请求参数
 - 1.1.3. 返回数据
 - 1.1.4. HTTP请求示例
 - 1.1.5. HTTP响应示例
 - 1.2. 删除人员组
 - 1.2.1. 基本信息
 - 1.2.2. 请求参数
 - 1.2.3. 返回数据
 - 1.2.4. HTTP请求示例
 - 1.2.5. HTTP响应示例
 - 1.3. 修改人员组
 - 1.3.1. 基本信息
 - 1.3.2. 请求参数
 - 1.3.3. 返回数据
 - 1.3.4. HTTP请求示例
 - 1.3.5. HTTP响应示例
 - 1.4. 人员组列表
 - 1.4.1. 基本信息
 - 1.4.2. 请求参数
 - 1.4.3. 返回数据
 - 1.4.4. HTTP请求示例
 - 1.4.5. HTTP响应示例
 - 1.5. 查询人员组详情
 - 1.5.1. 基本信息
 - 1.5.2. 请求参数
 - 1.5.3. 返回数据
 - 1.5.4. HTTP请求示例
 - 1.5.5. HTTP响应示例
 - 1.6. 批量移除组内人员
 - 1.6.1. 基本信息
 - 1.6.2. 请求参数
 - 1.6.3. 返回数据
 - 1.6.4. HTTP请求示例
 - 1.6.5. HTTP响应示例
 - 1.7. 查询组内人员详情
 - 1.7.1. 基本信息
 - 1.7.2. 请求参数
 - 1.7.3. 返回数据
 - 1.7.4. HTTP请求示例
 - 1.7.5. HTTP响应示例
 - 1.8. 编辑组内人员
 - 1.8.1. 基本信息
 - 1.8.2. 请求参数
 - 1.8.3. 返回数据
 - 1.8.4. HTTP请求示例

- 1.8.5. HTTP响应示例
- 1.9. 获取默认组信息（不建议使用）
 - 1.9.1. 基本信息
 - 1.9.2. 请求参数
 - 1.9.3. 返回数据
 - 1.9.4. HTTP请求示例
 - 1.9.5. HTTP响应示例

1. 组管理

1.1. 批量添加人员组

1.1.1. 基本信息

Path: /v1/api/pass/group/batchAdd

Method: POST

接口描述: 批量添加多个组

1.1.2. 请求参数

Headers

参数名称	参数值	是否必须	示例	备注
Content-Type	application/json	是		

Body

名称	类型	是否必须	备注
groupList	object []	非必须	人员组列表
└─uuid	string	非必须	人员组唯一标志uuid，长度1~40，不支持特殊字符
└─name	string	必须	组名称，唯一，长度2~32，不支持特殊字符
└─type	integer	必须	组类型，详情请查看数据字典“组类型”

1.1.3. 返回数据

名称	类型	是否必须	备注
successes	object []	非必须	成功列表
└─uuid	string	非必须	人员组唯一标志uuid
└─code	integer	非必须	结果状态码
└─name	string	非必须	人员组名称
└─msg	string	非必须	结果信息提示
└─type	integer	非必须	1 员工 2 访客 3 重点人员
failures	object []	非必须	失败列表
└─uuid	string	非必须	人员组唯一标志uuid
└─code	integer	非必须	结果状态码
└─name	string	非必须	人员组名称
└─msg	string	非必须	结果信息提示
└─type	integer	非必须	1 员工 2 访客 3 重点人员

1.1.4. HTTP请求示例

请求 path

```
/v1/api/pass/group/batchAdd
```

请求 body

```
{
  "groupList": [
    {
      "name": "我是员工组",
      "type": 1
    },
    {
      "name": "我是访客组",
      "type": 2
    },
    {
      "name": "我是重点人员组",
      "type": 3
    }
  ]
}
```

1.1.5. HTTP响应示例

响应 200

```
{
  "code": 0,
  "data": {
    "failures": [],
    "successes": [
      {
        "isAll": 0,
        "name": "我是员工组",
        "type": 1,
        "uuid": "64d49abe65e0458d8eb3edb5d7c52bb3"
      },
      {
        "isAll": 0,
        "name": "我是访客组",
        "type": 2,
        "uuid": "28da70b018ff402089c04d3a023bb7ea"
      },
      {
        "isAll": 0,
        "name": "我是重点人员组",
        "type": 3,
        "uuid": "b9b54f8cd9b24224919883a531244b00"
      }
    ]
  },
  "msg": "成功"
}
```

1.2. 删除人员组

1.2.1. 基本信息

Path: /v1/api/pass/group/delete

Method: POST

接口描述: 删除组信息

1.2.2. 请求参数

Headers

参数名称	参数值	是否必须	示例	备注
Content-Type	application/json	是		

Body

名称	类型	是否必须	备注
uuid	string	必须	组uuid

1.2.3. 返回数据

```
OK
```

1.2.4. HTTP请求示例

请求 path

```
/v1/api/pass/group/delete
```

请求 body

```
{
  "uuid": "158feab6a61746698ce6aa43264e8e4a"
}
```

1.2.5. HTTP响应示例

响应 200

```
{
  "code": 0,
  "msg": "成功"
}
```

1.3. 修改人员组

1.3.1. 基本信息

Path: /v1/api/pass/group/update

Method: POST

接口描述: 修改组名字

1.3.2. 请求参数

Headers

参数名称	参数值	是否必须	示例	备注
Content-Type	application/json	是		

Body

名称	类型	是否必须	备注
uuid	string	必须	人员组唯一标志uuid
name	string	必须	组名称

1.3.3. 返回数据

OK

1.3.4. HTTP请求示例

请求 path

/v1/api/pass/group/update

请求 body

```
{
  "uuid": "2075f5ceb40d456db65af7e6515b59c4",
  "name": "update员工组"
}
```

1.3.5. HTTP响应示例

响应 200

```
{
  "code": 0,
  "msg": "成功"
}
```

1.4. 人员组列表

1.4.1. 基本信息

Path: /v1/api/pass/group/list

Method: POST

接口描述: 查询1-员工； 2-访客； 3-重点人员 不同分类下的所有的组，支持通过name模糊查询

1.4.2. 请求参数

Headers

参数名称	参数值	是否必须	示例	备注
Content-Type	application/json	是		

Body

名称	类型	是否必须	备注
pageNum	integer	非必须	分页页码，默认为 1
pageSize	integer	非必须	分页大小，默认为 20
type	integer	必须	组类别：1-员工； 2-访客； 3-重点人员
name	string	非必须	组名

1.4.3. 返回数据

名称	类型	是否必须	备注
pageNum	integer	非必须	当前页码
pageSize	integer	非必须	分页大小
total	integer	非必须	总数
list	object []	非必须	
└─uuid	string	非必须	组UUID
└─name	string	非必须	组名
└─type	integer	非必须	组类型，详情请查看数据字典“组类型”
└─personTotal	integer	非必须	总人数

1.4.4. HTTP请求示例

请求 path

```
/v1/api/pass/group/list
```

请求 body

```
{
  "type": 1,
  "name": "员工默认组",
  "pageNum": 1,
  "pageSize": 10
}
```

1.4.5. HTTP响应示例

响应 200

```
{
  "code": 0,
  "data": {
    "list": [
      {
        "isAll": 0,
        "name": "员工默认组",
        "personTotal": 8,
        "type": 1,
        "uuid": "976055aae31445c5a583ebc1a53c7514"
      }
    ],
    "pageNum": 1,
    "pageSize": 10,
    "total": 1
  },
  "msg": "成功"
}
```

1.5. 查询人员组详情

1.5.1. 基本信息

Path: /v1/api/pass/group/query

Method: POST

接口描述: 查询指定组的详细信息，包括组内人员总数

1.5.2. 请求参数

Headers

参数名称	参数值	是否必须	示例	备注
Content-Type	application/json	是		

Body

名称	类型	是否必须	备注
uuid	string	必须	组uuid

1.5.3. 返回数据

名称	类型	是否必须	备注
uuid	string	非必须	uuid
name	string	非必须	组名
type	integer	非必须	组类型，详情请查看数据字典“组类型”
personTotal	integer	非必须	组内员工数

1.5.4. HTTP请求示例

请求 path

```
/v1/api/pass/group/query
```

请求 body

```
{
  "uuid": "00eea45c96e64954a8de2d89aa5adcb9"
}
```

1.5.5. HTTP响应示例

响应 200

```
{
  "code": 0,
  "data": {
    "name": "员工组备用",
    "personTotal": 1,
    "type": 1,
    "uuid": "00eea45c96e64954a8de2d89aa5adcb9"
  },
  "msg": "成功"
}
```

1.6. 批量移除组内人员

1.6.1. 基本信息

Path: /v1/api/pass/group/batchDeletePerson

Method: POST

接口描述: 一次删除组内多个人员

1.6.2. 请求参数

Headers

参数名称	参数值	是否必须	示例	备注
Content-Type	application/json	是		

Body

名称	类型	是否必须	备注
groupUuid	string	必须	人员组唯一标志uuid
personUuids	string []	必须	人员uuid列表

1.6.3. 返回数据

OK

1.6.4. HTTP请求示例**请求 path**`/v1/api/pass/group/batchDeletePerson`**请求 body**

```
{
  "groupUuid": "6eed172bb5084455bbd7d29d53564289",
  "personUuids": [
    "57f1acb42a0046ff9c2e957756b3e7b8"
  ]
}
```

1.6.5. HTTP响应示例**响应 200**

```
{
  "code": 0,
  "msg": "成功"
}
```

1.7. 查询组内人员详情**1.7.1. 基本信息****Path:** `/v1/api/pass/group/queryPersonListInGroup`**Method:** POST

接口描述：查询指定组内人员详情，支持通过name、personCode模糊查询

1.7.2. 请求参数

Headers

参数名称	参数值	是否必须	示例	备注
Content-Type	application/json	是		

Body

名称	类型	是否必须	备注
pageNum	integer	非必须	分页页码，默认为 1
pageSize	integer	非必须	分页大小，默认为 20
uuid	string	必须	人员组唯一标志uuid
type	integer	必须	组类型，详情请查看数据字典“组类型”
name	string	非必须	人员姓名
personCode	string	非必须	人员编号

1.7.3. 返回数据

名称	类型	是否必须	备注
pageNum	integer	非必须	当前页码
pageSize	integer	非必须	分页大小
total	integer	非必须	总数
list	object []	非必须	
└─groupUuid	string	非必须	组uuid
└─personUuid	string	非必须	人员uuid
└─imageUri	string	非必须	用户图片
└─personName	string	非必须	人员名
└─personCode	string	非必须	员工编号(员工组)
└─visitTargetUuid	string	非必须	拜访用户uuid
└─visitTarget	string	非必须	拜访用户姓名
└─visitStartTime	string	非必须	拜访起始时间(时间戳)
└─visitEndTime	string	非必须	拜访结束时间(时间戳)
└─status	string	非必须	访客状态：无效、有效
└─phone	string	非必须	手机号码
└─identifyNum	string	非必须	身份证号码
└─type	integer	非必须	人员类型, 1: 员工; 2: 访客; 3: 重点人员
└─visitReason	string	非必须	拜访原因, 访客可填, 格式: 任意字符, 长度: 1-255

1.7.4. HTTP请求示例

请求 path

```
/v1/api/pass/group/queryPersonListInGroup
```

请求 body

```
{
  "type": 1,
  "uuid": "1b3c5dedd5984c628b1beb6b80d0e102",
  "pageSize": 10,
  "pageNum": 1
}
```

1.7.5. HTTP响应示例

响应 200

```
{
  "code": 0,
  "data": {
    "list": [
      {
        "groupUuid": "1b3c5dedd5984c628b1beb6b80d0e102",
        "imageUri": "http://10.171.4.130/pub/_ZzEwMF9mb3JldmVyQnVja2V0",
        "personCode": "40906240",
        "personName": "备用员工1",
        "personUuid": "14cc526644b740d9a94009e0ff25e51e",
        "phone": "13684505620",
        "type": 1
      }
    ],
    "pageNum": 1,
    "pageSize": 10,
    "total": 1
  },
  "msg": "成功"
}
```

1.8. 编辑组内人员

1.8.1. 基本信息

Path: /v1/api/pass/group/batchUpdatePerson

Method: POST

接口描述: 组内批量增加或删除人员，personUuids为空，删除组内所有人员，personUuids不为空，通过 groupUuid获取组内已有人员为originPersonUuids，进行差集计算，例：待删除：originPersonUuids-personUuids，待添加：personUuids-originPersonUuids

1.8.2. 请求参数

Headers

参数名称	参数值	是否必须	示例	备注
Content-Type	application/json	是		

Body

名称	类型	是否必须	备注
groupUuid	string	非必须	人员组唯一标志uuid
personUids	string []	非必须	人员uuid列表

1.8.3. 返回数据

OK

1.8.4. HTTP请求示例

请求 path

/v1/api/pass/group/batchUpdatePerson

请求 body

```
{
  "groupUuid": "b0b9ed9f106f4b7cb12ae5ccd44c70b8",
  "personUids": [
    "6e386a7b0dcd4df38c57bc990d5ce62e",
    "c90a3919a62442059d5319e1b9c30bcd"
  ]
}
```

1.8.5. HTTP响应示例

响应 200

```
{
  "code": 0,
  "msg": "成功"
}
```

1.9. 获取默认组信息（不建议使用）

1.9.1. 基本信息

Path: /v1/api/pass/group/getDefaultGroup

Method: POST

接口描述: 获取每个分类下一个默认的组

1.9.2. 请求参数

Headers

参数名称	参数值	是否必须	示例	备注
Content-Type	application/json	是		

Body

名称	类型	是否必须	备注
type	integer	必须	组类别: 1-员工; 2-访客; 3-重点人员

1.9.3. 返回数据

名称	类型	是否必须	备注
uuid	string	必须	uuid
name	string	必须	组名
type	integer	必须	组类型, 详情请查看数据字典“组类型”
personTotal	integer	必须	组内员工数

1.9.4. HTTP请求示例

请求 path

```
/v1/api/pass/group/getDefaultGroup
```

请求 body

```
{  
  "type": 1  
}
```

1.9.5. HTTP响应示例

响应 200

```
{
  "code": 0,
  "data": {
    "name": "员工默认组",
    "personTotal": 0,
    "type": 1,
    "uuid": "976055aae31445c5a583ebc1a53c7514"
  },
  "msg": "成功"
}
```

- 1. 设备管理
 - 1.1. 设备添加
 - 1.1.1. 基本信息
 - 1.1.2. 请求参数
 - 1.1.3. 返回数据
 - 1.1.4. HTTP请求示例
 - 1.1.5. HTTP响应示例
 - 1.2. 发现设备列表
 - 1.2.1. 基本信息
 - 1.2.2. 请求参数
 - 1.2.3. 返回数据
 - 1.2.4. HTTP请求示例
 - 1.2.5. HTTP响应示例
 - 1.3. 面板机设备导入
 - 1.3.1. 基本信息
 - 1.3.2. 请求参数
 - 1.3.3. 返回数据
 - 1.3.4. HTTP请求示例
 - 1.3.5. HTTP响应示例
 - 1.4. 删除设备预校验
 - 1.4.1. 基本信息
 - 1.4.2. 请求参数
 - 1.4.3. 返回数据
 - 1.4.4. HTTP请求示例
 - 1.4.5. HTTP响应示例
 - 1.5. 删除设备
 - 1.5.1. 基本信息
 - 1.5.2. 请求参数
 - 1.5.3. 返回数据
 - 1.5.4. HTTP请求示例
 - 1.5.5. HTTP响应示例
 - 1.6. 修改设备信息
 - 1.6.1. 基本信息
 - 1.6.2. 请求参数
 - 1.6.3. 返回数据
 - 1.6.4. HTTP请求示例
 - 1.6.5. HTTP响应示例
 - 1.7. 查询设备列表
 - 1.7.1. 基本信息
 - 1.7.2. 请求参数
 - 1.7.3. 返回数据
 - 1.7.4. HTTP请求示例
 - 1.7.5. HTTP响应示例
 - 1.8. 查询设备信息（不建议使用）
 - 1.8.1. 基本信息
 - 1.8.2. 请求参数
 - 1.8.3. 返回数据
 - 1.8.4. HTTP请求示例

- 1.8.5. HTTP响应示例
- 1.9. 查询设备详情
 - 1.9.1. 基本信息
 - 1.9.2. 请求参数
 - 1.9.3. 返回数据
 - 1.9.4. HTTP请求示例
 - 1.9.5. HTTP响应示例
- 1.10. 查询算力设备可绑定的设备列表
 - 1.10.1. 基本信息
 - 1.10.2. 请求参数
 - 1.10.3. 返回数据
 - 1.10.4. HTTP请求示例
 - 1.10.5. HTTP响应示例
- 1.11. 按设备类型分组查询设备列表（不建议使用）
 - 1.11.1. 基本信息
 - 1.11.2. 请求参数
 - 1.11.3. 返回数据
 - 1.11.4. HTTP请求示例
 - 1.11.5. HTTP响应示例
- 1.12. 查询设备型号关系（不建议使用）
 - 1.12.1. 基本信息
 - 1.12.2. 请求参数
 - 1.12.3. 返回数据
 - 1.12.4. HTTP请求示例
 - 1.12.5. HTTP响应示例
- 1.13. 设备绑定
 - 1.13.1. 基本信息
 - 1.13.2. 请求参数
 - 1.13.3. 返回数据
 - 1.13.4. HTTP请求示例
 - 1.13.5. HTTP响应示例
- 1.14. 设备解绑
 - 1.14.1. 基本信息
 - 1.14.2. 请求参数
 - 1.14.3. 返回数据
 - 1.14.4. HTTP请求示例
 - 1.14.5. HTTP响应示例
- 1.15. 查询门禁、人员管控设备列表
 - 1.15.1. 基本信息
 - 1.15.2. 请求参数
 - 1.15.3. 返回数据
 - 1.15.4. HTTP请求示例
 - 1.15.5. HTTP响应示例
- 1.16. 添加人员管控设备
 - 1.16.1. 基本信息
 - 1.16.2. 请求参数
 - 1.16.3. 返回数据
 - 1.16.4. HTTP请求示例

- 1.16.5. HTTP响应示例
- 1.17. 移除人员管控设备
 - 1.17.1. 基本信息
 - 1.17.2. 请求参数
 - 1.17.3. 返回数据
 - 1.17.4. HTTP请求示例
 - 1.17.5. HTTP响应示例
- 1.18. 查询可布控设备列表
 - 1.18.1. 基本信息
 - 1.18.2. 请求参数
 - 1.18.3. 返回数据
 - 1.18.4. HTTP请求示例
 - 1.18.5. HTTP响应示例

1. 设备管理

1.1. 设备添加

1.1.1. 基本信息

Path: /v1/api/device/add

Method: POST

接口描述: 设备新增接口，仅支持网络摄像机、智能摄像头、智能服务器、存储服务器的添加，面板机设备需通过**面板机设备导入**添加至系统，仅添加网络摄像机支持Onvif协议，并且Onvif协议下网络摄像机可以无须传递rtsp视频流地址，智能摄像机、网络摄像机、智能服务器中的MegBox-B3R-1012(H1)、MegCube-B4H16-311设备添加时必须添加设备模式，部分设备（详细请查看数据字典）支持以主动模式添加，主动模式下无需填写IP地址，但必须填写snCode和端口号；

1.1.2. 请求参数

Headers

参数名称	参数值	是否必须	示例	备注
Content-Type	application/json	是		

Body

名称	类型	是否必须	备注
name	string	必须	设备名称,长度1-50位
location	string	必须	设备位置,长度1-50位
deviceType	integer	必须	设备类型 1、智能摄像头 3、智能服务器 4、网络摄像机 5、存储服务器
deviceModel	integer	必须	设备型号（详细型号描述请查看数据字典）与"设备的类型"相对应
deviceMode	integer	非必须	设备模式（服务器无，其他设备必填） 1、人脸识别 4、人脸抓拍
ipAddress	string	必须	设备IP
mac	string	非必须	mac地址
userName	string	必须	用户名长度1-50位
userPassword	string	必须	用户密码
rtspUrl	string	非必须	视频流地址:b3r不需要填改字段
port	integer	非必须	端口：添加设备的端口,需要和设备登录端口保持一致,智能摄像头、网络摄像机默认的登录端口是3000,B3R默认登录端口是80
protocol	integer	非必须	协议类型:1rtspt协议 2onvif协议（默认为1）
activeMode	integer	非必须	是否支持主动模式，0：非主动模式，1：主动模式 默认 0
snCode	string	非必须	设备sn码，PDNS接入时必须填

名称	类型	是否必须	备注
joinProtocol	integer	非必须	接入协议: (智能摄像机填写, 其他设备不需要) 1-SDK, 2-1400, 默认1

1.1.3. 返回数据

名称	类型	是否必须	备注
successes	object []	必须	成功列表
└─ uuid	string	必须	设备uuid
└─ code	integer	非必须	结果状态码
└─ name	string	必须	设备名称
└─ location	string	必须	设备位置
└─ msg	string	非必须	结果信息提示
└─ deviceType	integer	必须	设备类型 1、智能摄像头 3、智能服务器 4、网络摄像机 5、存储服务器
└─ deviceModel	integer	必须	设备型号（详细型号描述请查看数据字典）与"设备的类型"相对应
└─ deviceMode	integer	非必须	设备模式（服务器无，其他设备必填） 1、人脸识别 4、人脸抓拍
└─ ipAddress	string	必须	设备IP
└─ mac	string	非必须	mac地址
└─ userName	string	非必须	用户名
└─ userPassword	string	非必须	用户密码
└─ rtspUrl	string	非必须	视频流地址
└─ port	integer	非必须	端口

名称	类型	是否必须	备注
 protocol	integer	非必须	协议类型:1rtspt协议 2onvif协议
 activeMode	integer	非必须	是否支持主动模式, 0: 非主动模式, 1: 主动模式 支持PDNS的设备类型: 智能摄像头: MegEye-C4N-888、大华、海康、宇视、科大、华为不支持, 网络摄像机MegEye-C4N-888、IPC不支持、 智能服务器B3R、B4H支持、core不支持、存储服务器不支持, 面板机不支持
 snCode	string	非必须	设备sn码, PDNS接入时必须填
failures	object []	必须	失败列表
 uuid	string	必须	设备uuid
 code	integer	非必须	结果状态码
 name	string	必须	设备名称
 location	string	必须	设备位置
 msg	string	非必须	结果信息提示
 deviceType	integer	必须	设备类型 1、智能摄像头 3、智能服务器 4、网络摄像机 5、存储服务器
 deviceModel	integer	必须	设备型号 (详细型号描述请查看数据字典) 与"设备的类型"相对应
 deviceMode	integer	非必须	设备模式 (服务器无, 其他设备必填) 1、人脸识别 4、人脸抓拍
 ipAddress	string	必须	设备IP

名称	类型	是否必须	备注
 mac	string	非必须	mac地址
 userName	string	非必须	用户名
 userPassword	string	非必须	用户密码
 rtspUrl	string	非必须	视频流地址
 port	integer	非必须	端口
 protocol	integer	非必须	协议类型:1rtspt协议 2onvif协议
 activeMode	integer	非必须	是否支持主动模式, 0: 非主动模式, 1: 主动模式 支持PDNS的设备类型: 智能摄像头: MegEye-C4N-888、大华、海康、宇视、科大、华为不支持, 网络摄像机MegEye-C4N-888、IPC不支持、 智能服务器B3R、B4H支持、core不支持、存储服务器不支持, 面板机不支持
 snCode	string	非必须	设备sn码, PDNS接入时必须填

1.1.4. HTTP请求示例

请求 path

```
/v1/api/device/add
```

请求 body

```
{
  "name": "c4r-322",
  "location": "openapi",
  "deviceType": 1,
  "deviceModel": 12,
  "deviceMode": 4,
  "ipAddress": "10.171.4.89",
  "userName": "admin",
  "userPassword": "admin123",
  "rtspUrl": "rtsp://admin:admin123@10.171.4.89:554/1/1",
  "port": 3000
}
```

1.1.5. HTTP响应示例

响应 200

```
{
  "code": 0,
  "data": {
    "failures": [
    ],
    "successes": [
      {
        "activeMode": 0,
        "deviceMode": 4,
        "deviceModel": 12,
        "deviceType": 1,
        "ipAddress": "10.171.4.89",
        "location": "openapi",
        "name": "c4r-322",
        "port": 3000,
        "protocol": 1,
        "rtspUrl": "rtsp://admin:admin123@10.171.4.89:554/1/1",
        "userName": "admin",
        "userPassword": "admin123",
        "uuid": "ac70de8cac3747d289d1a9f3b122be6e"
      }
    ]
  },
  "msg": "成功"
}
```

1.2. 发现设备列表

1.2.1. 基本信息

Path: /v1/api/device/queryReportDevice

Method: POST

接口描述: 查询面板机注册系统但未进行面板机导入的设备列表，通过面板机注册至鸿图系统后，可在发现设备列表查询到此设备并将设备导入至系统；

1.2.2. 请求参数

Headers

参数名称	参数值	是否必须	示例	备注
Content-Type	application/json	是		

Body

名称	类型	是否必须	备注
pageNum	integer	非必须	当前页码,不传默认1
snCode	string	非必须	设备编码
deviceType	integer	非必须	设备类型 2 智能面板机
pageSize	integer	非必须	分页大小, 不传默认20
discoveryMethod	integer	非必须	发现方式 1-主动上报, 详情请查看数据字典“接入模式”

1.2.3. 返回数据

名称	类型	是否必须	备注
pageNum	integer	必须	当前页码
pageSize	integer	必须	分页大小
total	integer	必须	总数
list	object []	非必须	
└─ uuid	string	必须	设备uuid
└─ deviceType	integer	必须	设备类型 2 智能面板机
└─ snCode	string	必须	设备编码
└─ gmtCreate	string	必须	操作时间
└─ discoveryMethod	integer	必须	发现方式 1-主动上报，详情请查看数据字典“接入模式”

1.2.4. HTTP请求示例

请求 path

```
/v1/api/device/queryReportDevice
```

请求 body

```
{
  "pageNum": 1,
  "snCode": "M014200332004009999",
  "deviceType": 2,
  "pageSize": 20,
  "discoveryMethod": 1
}
```

1.2.5. HTTP响应示例

响应 200

```
{
  "code": 0,
  "data": {
    "list": [
      {
        "deviceType": 2,
        "discoveryMethod": 1,
        "gmtCreate": 1626202654000,
        "snCode": "M014200332004009999",
        "uuid": "7uK5mvPufquC0YvzE0s0b7zNA9090qxz"
      }
    ],
    "pageNum": 1,
    "pageSize": 20,
    "total": 1
  },
  "msg": "成功"
}
```

1.3. 面板机设备导入

1.3.1. 基本信息

Path： /v1/api/device/importDevice

Method： POST

接口描述： 面板机注册至系统后，在发现设备列表发现此设备，通过此接口导入至鸿图系统，仅面板机支持，其他类型设备需通过设备添加接口添加；

1.3.2. 请求参数

Headers

参数名称	参数值	是否必须	示例	备注
Content-Type	application/json	是		

Body

名称	类型	是否必须	备注
name	string	必须	设备名称
location	string	必须	设备位置
deviceType	integer	必须	设备类型: 2 智能面板机
deviceModel	integer	必须	设备型号（详细型号描述请查看数据字典）与"设备的类型"相对应
deviceMode	integer	必须	设备模式 1-人脸识别
snCode	string	必须	设备编码
discoveryMethod	integer	必须	

1.3.3. 返回数据

OK

1.3.4. HTTP请求示例

请求 path

/v1/api/device/importDevice

请求 body

```
{
  "name": "w5k-i8-fake",
  "location": "openapi",
  "deviceType": 2,
  "deviceModel": 1,
  "deviceMode": 1,
  "snCode": "M014200332004009999",
  "discoveryMethod": 1
}
```

1.3.5. HTTP响应示例

响应 200

```
{
  "code": 0,
  "msg": "成功"
}
```

1.4. 删除设备预校验

1.4.1. 基本信息

Path: /v1/api/device/preDelete

Method: POST

接口描述: 删除设备前，调用此接口进行预删除校验，防止绑定、布控等正在使用的设备删除；

1.4.2. 请求参数

Headers

参数名称	参数值	是否必须	示例	备注
Content-Type	application/json	是		

Body

名称	类型	是否必须	备注
uuid	string	必须	设备uuid

1.4.3. 返回数据

名称	类型	是否必须	备注
tag	integer	必须	标识 0-可删除 1-已绑定 2-已布控 3-强制删除 5-内置算力芯片不可删除(仅2K有),tag在0,3状态下表示可删除 4-中心算力设备
deployType	integer	非必须	布控类型 3-门禁点布控

1.4.4. HTTP请求示例

请求 path

```
/v1/api/device/preDelete
```

请求 body

```
{
  "uuid": "eabe5bbbd4e24faba672aa45d09a6607"
}
```

1.4.5. HTTP响应示例

响应 200

```
{
  "code": 0,
  "data": {
    "tag": 0
  },
  "msg": "成功"
}
```

1.5. 删除设备

1.5.1. 基本信息

Path: /v1/api/device/delete

Method: POST

接口描述: 已添加设备的删除接口，删除设备前需调用删除设备预校验接口；

1.5.2. 请求参数

Headers

参数名称	参数值	是否必须	示例	备注
Content-Type	application/json	是		

Body

名称	类型	是否必须	备注
uuid	string	必须	设备uuid

1.5.3. 返回数据

```
OK
```

1.5.4. HTTP请求示例

请求 path

```
/v1/api/device/delete
```

请求 body

```
{
  "uuid": "eabe5bbbd4e24faba672aa45d09a6607"
}
```

1.5.5. HTTP响应示例

响应 200

```
{
  "code": 0,
  "msg": "成功"
}
```

1.6. 修改设备信息

1.6.1. 基本信息

Path: /v1/api/device/update

Method: POST

接口描述: 修改设备信息接口: 1.已绑定算力设备的相机只允许修改基础信息（名称、位置、mac地址）; 2.未绑定的相机可以修改几乎全部信息（不可修改设备类型、设备型号、设备模式、协议类型、SN码、主动模式）; 3.内置算力芯片不可调用该接口(需要传入全量参数)

1.6.2. 请求参数

Headers

参数名称	参数值	是否必须	示例	备注
Content-Type	application/json	是		

Body

名称	类型	是否必须	备注
uuid	string	必须	设备uuid
name	string	必须	设备名称
location	string	必须	设备位置
ipAddress	string	必须	设备IP: 绑定设备后不可修改
mac	string	非必须	mac地址:非必须,
userName	string	非必须	用户名:绑定设备后不可修改
userPassword	string	非必须	用户密码: 绑定设备后不可修改
port	integer	非必须	端口:智能摄像头、网络摄像机默认的登录端口是3000,B3R默认登录端口是80
rtspUrl	string	非必须	视频流地址:非必须, 绑定设备后不可修改
activeMode	integer	非必须	是否支持主动模式 1 主动模式 0 非主动 默认0

1.6.3. 返回数据

OK

1.6.4. HTTP请求示例

请求 path

/v1/api/device/update

请求 body


```
{
  "uuid": "6d4a8daf3cc44551b49211a825c6076c",
  "name": "c3s-123",
  "location": "openapi0",
  "ipAddress": "10.171.4.92",
  "userName": "admin",
  "userPassword": "admin123",
  "port": 34567,
  "rtspUrl": "rtsp://10.171.4.92/user=admin&password=admin123&channel=1&stream=0"
}
```

1.6.5. HTTP响应示例

响应 200

```
{
  "code": 0,
  "msg": "成功"
}
```

1.7. 查询设备列表

1.7.1. 基本信息

Path： /v1/api/device/list

Method： POST

接口描述： 分页查询设备列表，返回设备的基础信息；

1.7.2. 请求参数

Headers

参数名称	参数值	是否必须	示例	备注
Content-Type	application/json	是		

Body

名称	类型	是否必须	备注
name	string	非必须	设备名称
pageNum	integer	非必须	当前页码,不传默认1
deviceType	integer	非必须	设备类型 1 智能摄像头 2 智能面板机 3 智能服务器 4 网络摄像机 5 存储服务器
pageSize	integer	非必须	分页大小, 不传默认20

1.7.3. 返回数据

名称	类型	是否必须	备注
pageNum	integer	必须	当前页码
pageSize	integer	必须	分页大小
total	integer	必须	总数
list	object []	非必须	
└─ uuid	string	必须	设备uuid
└─ location	string	必须	设备位置
└─ name	string	必须	设备名称
└─ deviceType	integer	必须	设备类型 1 智能摄像头 2 智能面板机 3 智能服务器 4 网络摄像机 5 存储服务器
└─ status	integer	必须	设备状态 3 在线 4 离线
└─ ipAddress	string	必须	设备IP
└─ mac	string	非必须	mac地址
└─ deviceMode	integer	必须	设备模式（详细模式请查看数据字典）
└─ upgradeStatus	integer	非必须	升级状态 0 升级中 1 升级成功 2 升级失败 4 可升级 5 升级完成
└─ upgradeAction	integer	非必须	升级操作 0-更新版本（置灰） 1-更新版本 2-历史版本
└─ currentVersion	string	非必须	当前版本

名称	类型	是否必须	备注
 checkFirm	integer	非必须	设备型号是否是固件 0 表示不是 1 表示是固件

1.7.4. HTTP请求示例

请求 path

```
/openapi/v1/device/list
```

请求 body

```
{
  "pageNum": 1,
  "pageSize": 5,
  "name": "",
  "deviceType": 4
}
```

1.7.5. HTTP响应示例

响应 200

```
body{
  "code": 0,
  "data": {
    "list": [{
      "checkFirm": 0,
      "deviceMode": 1,
      "deviceType": 4,
      "ipAddress": "10.171.8.140",
      "location": "openapi",
      "name": "c3v-120-p",
      "status": 3,
      "upgradeAction": 0,
      "uuid": "0e8b42ac716d4f849348d0c81c4db693"
    }
  ],
  "pageNum": 1,
  "pageSize": 5,
  "total": 1
},
"msg": "成功"
}
```

1.8. 查询设备信息（不建议使用）

1.8.1. 基本信息

Path: /v1/api/device/queryDeviceInfo

Method: POST

接口描述: 查询设备详情接口，返回设备部门信息

1.8.2. 请求参数

Headers

参数名称	参数值	是否必须	示例	备注
Content-Type	application/json	是		

Body

名称	类型	是否必须	备注
uuid	string	必须	设备uuid

1.8.3. 返回数据

名称	类型	是否必须	备注
uuid	string	必须	uuid
name	string	必须	设备名称
ipAddress	string	非必须	设备IP
location	string	非必须	设备位置
deviceType	integer	必须	设备类型 1 智能摄像头 2 智能面板机 3 智能服务器 4 网络摄像机
mac	string	非必须	mac地址
deviceMode	integer	非必须	设备模式（详细模式请查看数据字典）
userName	string	非必须	用户名
userPassword	string	非必须	用户密码
deviceModel	integer	必须	设备型号（详细型号描述请查看数据字典）与"设备的类型"相对应
discoveryMethod	integer	必须	发现方式 1-主动上报，详情请查看数据字典“接入模式”
snCode	string	非必须	设备编码
rtspUrl	string	非必须	视频流地址
port	integer	非必须	端口号

名称	类型	是否必须	备注
protocol	integer	非必须	协议类型:1rtspt协议 2onvif协议
activeMode	integer	非必须	是否支持主动模式 1 主动模式 0 非主动 默认0
isSupportActiveSdk	integer	非必须	是否支持activeSdk 1支持 0不支持
cmdStatus	integer	非必须	常开常闭状态码 17 正常 15 常开 16 常闭
joinProtocol	integer	非必须	1-SDK, 2-1400

1.8.4. HTTP请求示例

请求 path

```
/openapi/v1/device/queryDeviceInfo
```

请求 body

```
{  
  "uuid":"eabe5bbbd4e24faba672aa45d09a6607"  
}
```

1.8.5. HTTP响应示例

响应 200

```
{
  "code":0,
  "data":{
    "activeMode":0,
    "cmdStatus":17,
    "deviceMode":4,
    "deviceModel":4,
    "deviceType":1,
    "discoveryMethod":2,
    "ipAddress":"10.171.4.92",
    "isSupportActiveSdk":1,
    "joinProtocol":1,
    "location":"openapi",
    "mac":"88:88:88:88:88:88:81",
    "name":"c3s-123",
    "port":34567,
    "rtspUrl":"rtsp://10.171.4.92/user=admin&password=admin123&channel=1&...",
    "userName":"admin",
    "userPassword":"admin123",
    "uuid":"9fe34c48421a42119313ee31440e0297"
  },
  "msg":"成功"
}
```

1.9. 查询设备详情

1.9.1. 基本信息

Path： /v1/api/device/query

Method： POST

接口描述： 查询设备详细信息接口， 包含设备全部信息；

1.9.2. 请求参数

Headers

参数名称	参数值	是否必须	示例	备注
Content-Type	application/json	是		

Body

名称	类型	是否必须	备注
uuid	string	必须	设备uuid

1.9.3. 返回数据

名称	类型	是否必须	备注
uuid	string	必须	uuid
name	string	必须	设备名称
ipAddress	string	必须	设备IP
location	string	必须	设备位置
deviceType	integer	必须	设备类型 1、智能摄像头 2、智能面板机 3、智能服务器 4、网络摄像机 5、存储服务器
featureVersion	string	非必须	算法版本号
appVersion	string	非必须	软件版本号
status	integer	必须	设备状态 3、在线 4、离线
cpuUsage	number	非必须	cpu使用率
memoryUsage	number	非必须	内存使用率
diskUsage	number	非必须	存储使用率
cpuTemperature	number	非必须	cpu温度
memoryValue	number	非必须	内存空间
diskValue	number	非必须	存储空间

名称	类型	是否必须	备注
deviceMode	integer	非必须	设备模式 1、人脸识别 4、人脸抓拍
mac	string	非必须	mac地址
bindAiServer	object	非必须	设备为端且端绑定智能服务器才有返回
└─ uuid	string	必须	设备uuid
└─ name	string	必须	设备name
└─ deviceType	integer	必须	3、智能服务器
└─ status	integer	必须	设备状态 3、在线 4、离线
bindStorageServer	object	非必须	设备为端且端绑定存储服务器才有返回
└─ uuid	string	必须	设备uuid
└─ name	string	必须	设备name
└─ deviceType	integer	必须	5、存储服务器
└─ status	integer	必须	设备状态 3、在线 4、离线
bindDevices	object []	非必须	
└─ uuid	string	必须	设备uuid
└─ name	string	必须	设备name
└─ deviceType	integer	必须	设备类型 1、智能摄像头 2、智能面板机 4、网络摄像机

名称	类型	是否必须	备注
status	integer	必须	设备状态 3、在线 4、离线
serverMode	integer	必须	服务器模式：1 人脸识别模式4 人脸抓拍模式 6 人脸识别标准模式 7 人脸识别高性能模式
protocol	integer	非必须	协议类型:1rtspt协议 2onvif协议
snCode	String	非必须	设备编码
activeMode	integer	非必须	是否支持主动模式 1 主动模式 0 非主动 默认0
isSupportActiveSdk	integer	非必须	是否支持activeSdk 1支持 0不支持
cmdStatus	integer	非必须	常开常闭状态码 17 正常 15 常开 16 常闭
joinProtocol	integer	非必须	1-SDK, 2-1400

1.9.4. HTTP请求示例

请求 path

```
/v1/api/device/query
```

请求 body

```
{ "uuid": "a80f416fe6344a388b2c41ca1eca4c7a" }
```

1.9.5. HTTP响应示例

响应 200

```
{
  "code":0,
  "data":{
    "activeMode":0,
    "bindAiServer":{
      "deviceType":3,
      "name":"b3r",
      "status":3,
      "uuid":"536ac92a3e304da9bffd8c2c94d65f6"
    },
    "bindDevices":[

    ],
    "cmdStatus":17,
    "deviceMode":4,
    "deviceType":1,
    "ipAddress":"10.171.4.92",
    "isSupportActiveSdk":1,
    "joinProtocol":1,
    "location":"openapi",
    "mac":"88:88:88:88:88:88:81",
    "name":"c3s-123",
    "status":3,
    "uuid":"a80f416fe6344a388b2c41ca1eca4c7a"
  },
  "msg":"成功"
}
```

1.10. 查询算力设备可绑定的设备列表

1.10.1. 基本信息

Path： /v1/api/device/list/enbind/group

Method： POST

接口描述： 算力设备为B3R等设备时， 查询此算力设备可绑定的设备列表

1.10.2. 请求参数


Headers

参数名称	参数值	是否必须	示例	备注
Content-Type	application/json	是		

Body

名称	类型	是否必须	备注
uuid	string	必须	设备uuid

1.10.3. 返回数据

名称	类型	是否必须	备注
	object []	非必须	
 deviceType	integer	必须	设备类型 1 智能摄像头 2 智能面板机 3 智能服务器 4 网络摄像机 5 存储服务器
 deviceList	object []	非必须	设备列表信息
uuid 	string	必须	设备uuid
name 	string	必须	设备名称
bindType 	integer	必须	是否绑定设备 0-未绑定 1-绑定

1.10.4. HTTP请求示例

请求 path

```
/v1/api/device/list/enbind/group
```

请求 body

```
{
  "uuid": "6b9a32333c4c42a1912d2fb31b38d01d"
}
```

1.10.5. HTTP响应示例

响应 200

```

{
  "code": 0,
  "data": [
    {
      "deviceList": [
        {
          "bindType": 0,
          "name": "智能摄像头-MegEye-C3S-123-人脸抓拍",
          "uuid": "3ff7113b697244ce9f203114e4ce22df"
        },
        {
          "bindType": 0,
          "name": "智能摄像头-MegEye-C4H-142-人脸抓拍",
          "uuid": "1bbd8d1fa8de454eb90eaacef53654b4"
        },
        {
          "bindType": 0,
          "name": "智能摄像头-MegEye-C4S-222-人脸抓拍",
          "uuid": "21b049c1987d45c686f2d85b7eb23ce4"
        },
        {
          "bindType": 0,
          "name": "智能摄像头-MegEye-C4H-242-人脸抓拍",
          "uuid": "3a2fb9a7b24246c6bbc6a1c86833e386"
        }
      ],
      "deviceType": 1
    },
    {
      "deviceList": [
        {
          "bindType": 0,
          "name": "网络摄像机-MegEye-C4H-142-人脸识别",
          "uuid": "52c9c2d2310e4aa2b480650c59e66818"
        },
        {
          "bindType": 0,
          "name": "网络摄像机-IPC-人脸识别",
          "uuid": "c23a575a536e48f3b5e1cb249adf15fc"
        }
      ],
      "deviceType": 4
    }
  ],
  "msg": "成功"
}

```

1.11. 按设备类型分组查询设备列表（不建议使用）

1.11.1. 基本信息

Path: /v1/api/device/listBygroup

Method: POST

接口描述: 按照设备类型分组查询设备列表信息

1.11.2. 请求参数

Headers

参数名称	参数值	是否必须	示例	备注
Content-Type	application/json	是		

1.11.3. 返回数据

名称	类型	是否必须	备注
	object []	非必须	
 deviceType	integer	必须	设备类型 1 智能摄像头 2 智能面板机 3 智能服务器 4 网络摄像机 5 存储服务器
 deviceList	object []	非必须	设备列表信息
uuid 	string	必须	设备uuid
name 	string	必须	设备名称

1.11.4. HTTP请求示例

请求 path

```
/v1/api/device/listBygroup
```

请求 body

```
{}
```

1.11.5. HTTP响应示例

响应 200

```

{
  "code": 0,
  "data": [
    {
      "deviceList": [
        {
          "name": "警戒设备",
          "uuid": "34e2f5500c7b4a5f9109f9cbd5565734"
        }
      ],
      "deviceType": 1
    },
    {
      "deviceList": [],
      "deviceType": 2
    },
    {
      "deviceList": [
        {
          "name": "算力引擎",
          "uuid": "bc869ea7467d44bab1808fa7822ea0a2"
        },
        {
          "name": "智能服务器",
          "uuid": "f4d3a4895b1744bbb6affb1a6a2e50d3"
        },
        {
          "name": "b3r",
          "uuid": "9597252269014ad7810e1b021546eeee"
        }
      ],
      "deviceType": 3
    },
    {
      "deviceList": [],
      "deviceType": 4
    },
    {
      "deviceList": [
        {
          "name": "存储服务器",
          "uuid": "07f6da0f62bb4685bb477c356acae311"
        }
      ],
      "deviceType": 5
    }
  ],
  "msg": "成功"
}

```

1.12. 查询设备型号关系（不建议使用）

1.12.1. 基本信息

Path: /v1/api/device/queryModelRelation

Method: POST

接口描述: 可以查询系统所有设备型号支持的设备模式、是否支持主动模式等信息，考虑到此接口过于依赖业务，后期将不建议使用；

1.12.2. 请求参数

Headers

参数名称	参数值	是否必须	示例	备注
Content-Type	application/json	是		

Body

名称	类型	是否必须	备注
relationType	integer	必须	映射类型 1: 返回全部映射关系 2: 返回除智能面板机外的其他四种设备/服务器映射关系 3: 返回智能面板机的映射关系

1.12.3. 返回数据

名称	类型	是否必须	备注
	object []	非必须	
└ deviceType	integer	非必须	设备类型id, 1 智能摄像头 2 智能面板机 3 智能服务器 4 网络摄像机 5 存储服务器
└ deviceTypeStr	string	非必须	类型名称 智能摄像头 智能面板机 智能服务器 网络摄像机 存储服务器
└ modelList	object []	非必须	设备型号和模型关系
└ deviceModel	integer	非必须	设备型号id
└ deviceModelStr	string	非必须	设备型号名称
└ isSupportActiveSdk	integer	非必须	是否支持avtiveSDK,0 不支持avtiveSDK, 1支持avtiveSDK
└ modeList	object []	非必须	设备模式
└ deviceMode	integer	非必须	模式id(服务器无模式) 1-人脸识别 4-人脸抓拍
└ deviceModeStr	string	非必须	模式名称 人脸识别 人脸抓拍

1.12.4. HTTP请求示例

请求 path

```
/openapi/v1/device/list
```

请求 body

通行记录回调

```
{  
  "relationType": 1  
}
```

1.12.5. HTTP响应示例

响应 200

```

{
  "code": 0,
  "data": [
    {
      "deviceType": 1,
      "deviceTypeStr": "智能摄像头",
      "modelList": [
        {
          "deviceModel": 4,
          "deviceModelStr": "MegEye-C3S-123",
          "isSupportActiveSdk": 1,
          "modeList": [
            {
              "deviceMode": 4,
              "deviceModeStr": "人脸抓拍模式"
            }
          ]
        },
        {
          "deviceModel": 5,
          "deviceModelStr": "MegEye-C4H-141",
          "isSupportActiveSdk": 1,
          "modeList": [
            {
              "deviceMode": 4,
              "deviceModeStr": "人脸抓拍模式"
            }
          ]
        },
        {
          "deviceModel": 6,
          "deviceModelStr": "MegEye-C4H-241",
          "isSupportActiveSdk": 1,
          "modeList": [
            {
              "deviceMode": 4,
              "deviceModeStr": "人脸抓拍模式"
            },
            {
              "deviceMode": 3,
              "deviceModeStr": "警戒模式"
            }
          ]
        }
      ],
      {
        "deviceModel": 7,
        "deviceModelStr": "MegEye-C4N-888",
        "isSupportActiveSdk": 0,
        "modeList": [
          {
            "deviceMode": 2,
            "deviceModeStr": "结构化模式"
          }
        ]
      },
      {
        "deviceModel": 8,
        "deviceModelStr": "MegEye-C4S-222",
        "isSupportActiveSdk": 1,
        "modeList": [
          {
            "deviceMode": 4,
            "deviceModeStr": "人脸抓拍模式"
          }
        ]
      }
    ]
  ]
}

```

```

    },
    {
      "deviceModel": 9,
      "deviceModelStr": "MegEye-C4S-322",
      "isSupportActiveSdk": 1,
      "modeList": [
        {
          "deviceMode": 4,
          "deviceModeStr": "人脸抓拍模式"
        }
      ]
    },
    {
      "deviceModel": 10,
      "deviceModelStr": "MegEye-C4R-122",
      "isSupportActiveSdk": 1,
      "modeList": [
        {
          "deviceMode": 4,
          "deviceModeStr": "人脸抓拍模式"
        }
      ]
    },
    {
      "deviceModel": 11,
      "deviceModelStr": "MegEye-C4R-222",
      "isSupportActiveSdk": 1,
      "modeList": [
        {
          "deviceMode": 4,
          "deviceModeStr": "人脸抓拍模式"
        },
        {
          "deviceMode": 3,
          "deviceModeStr": "警戒模式"
        }
      ]
    },
    {
      "deviceModel": 12,
      "deviceModelStr": "MegEye-C4R-322",
      "isSupportActiveSdk": 1,
      "modeList": [
        {
          "deviceMode": 4,
          "deviceModeStr": "人脸抓拍模式"
        },
        {
          "deviceMode": 3,
          "deviceModeStr": "警戒模式"
        }
      ]
    },
    {
      "deviceModel": 29,
      "deviceModelStr": "海康摄像头",
      "isSupportActiveSdk": 0,
      "modeList": [
        {
          "deviceMode": 4,
          "deviceModeStr": "人脸抓拍模式"
        }
      ]
    },
    {
      "deviceModel": 30,

```

```

        "deviceModelStr": "大华摄像头",
        "isSupportActiveSdk": 0,
        "modeList": [
            {
                "deviceMode": 4,
                "deviceModeStr": "人脸抓拍模式"
            }
        ]
    },
    {
        "deviceModel": 31,
        "deviceModelStr": "宇视摄像头",
        "isSupportActiveSdk": 0,
        "modeList": [
            {
                "deviceMode": 4,
                "deviceModeStr": "人脸抓拍模式"
            }
        ]
    },
    {
        "deviceModel": 32,
        "deviceModelStr": "科大摄像头",
        "isSupportActiveSdk": 0,
        "modeList": [
            {
                "deviceMode": 4,
                "deviceModeStr": "人脸抓拍模式"
            }
        ]
    },
    {
        "deviceModel": 33,
        "deviceModelStr": "华为摄像头",
        "isSupportActiveSdk": 0,
        "modeList": [
            {
                "deviceMode": 4,
                "deviceModeStr": "人脸抓拍模式"
            }
        ]
    }
],
{
    "deviceType": 2,
    "deviceTypeStr": "智能面板机",
    "modelList": [
        {
            "deviceModel": 1,
            "deviceModelStr": "MegEye-W5K-I8",
            "isSupportActiveSdk": 0,
            "modeList": [
                {
                    "deviceMode": 1,
                    "deviceModeStr": "人脸识别模式"
                }
            ]
        },
        {
            "deviceModel": 2,
            "deviceModelStr": "MegEye-W4K-E7",
            "isSupportActiveSdk": 0,
            "modeList": [
                {
                    "deviceMode": 1,

```

```

        "deviceModeStr": "人脸识别模式"
    }
}
},
{
    "deviceModel": 3,
    "deviceModelStr": "MegEye-W4K-I51",
    "isSupportActiveSdk": 0,
    "modelList": [
        {
            "deviceMode": 1,
            "deviceModeStr": "人脸识别模式"
        }
    ]
},
{
    "deviceModel": 34,
    "deviceModelStr": "MegEye-W4K-I52",
    "isSupportActiveSdk": 0,
    "modelList": [
        {
            "deviceMode": 1,
            "deviceModeStr": "人脸识别模式"
        }
    ]
},
{
    "deviceModel": 35,
    "deviceModelStr": "MegEye-W3K-45S",
    "isSupportActiveSdk": 0,
    "modelList": [
        {
            "deviceMode": 1,
            "deviceModeStr": "人脸识别模式"
        }
    ]
},
{
    "deviceModel": 36,
    "deviceModelStr": "MegEye-W4K-I31",
    "isSupportActiveSdk": 0,
    "modelList": [
        {
            "deviceMode": 1,
            "deviceModeStr": "人脸识别模式"
        }
    ]
}
]
},
{
    "deviceType": 3,
    "deviceTypeStr": "智能服务器",
    "modelList": [
        {
            "deviceModel": 18,
            "deviceModelStr": "CORE",
            "isSupportActiveSdk": 0,
            "modelList": []
        },
        {
            "deviceModel": 22,
            "deviceModelStr": "MegBox-B3R-412",
            "isSupportActiveSdk": 1,
            "modelList": []
        }
    ]
},

```

```

    {
      "deviceModel": 23,
      "deviceModelStr": "MegBox-B3R-1012",
      "isSupportActiveSdk": 1,
      "modeList": []
    },
    {
      "deviceModel": 37,
      "deviceModelStr": "MegBox-B3R-1012(H1)",
      "isSupportActiveSdk": 1,
      "modeList": [
        {
          "deviceMode": 1,
          "deviceModeStr": "人脸识别模式"
        },
        {
          "deviceMode": 5,
          "deviceModeStr": "混合模式"
        }
      ]
    },
    {
      "deviceModel": 39,
      "deviceModelStr": "MegCube-B4H16-311",
      "isSupportActiveSdk": 1,
      "modeList": [
        {
          "deviceMode": 5,
          "deviceModeStr": "混合模式"
        }
      ]
    }
  ],
  {
    "deviceType": 4,
    "deviceTypeStr": "网络摄像机",
    "modelList": [
      {
        "deviceModel": 4,
        "deviceModelStr": "MegEye-C3S-123",
        "isSupportActiveSdk": 1,
        "modeList": [
          {
            "deviceMode": 2,
            "deviceModeStr": "结构化模式"
          },
          {
            "deviceMode": 3,
            "deviceModeStr": "警戒模式"
          },
          {
            "deviceMode": 1,
            "deviceModeStr": "人脸识别模式"
          }
        ]
      }
    ]
  },
  {
    "deviceModel": 5,
    "deviceModelStr": "MegEye-C4H-141",
    "isSupportActiveSdk": 1,
    "modeList": [
      {
        "deviceMode": 2,
        "deviceModeStr": "结构化模式"
      }
    ]
  },

```



```

        {
            "deviceMode": 3,
            "deviceModeStr": "警戒模式"
        },
        {
            "deviceMode": 1,
            "deviceModeStr": "人脸识别模式"
        }
    ]
},
{
    "deviceModel": 6,
    "deviceModelStr": "MegEye-C4H-241",
    "isSupportActiveSdk": 1,
    "modeList": [
        {
            "deviceMode": 2,
            "deviceModeStr": "结构化模式"
        },
        {
            "deviceMode": 3,
            "deviceModeStr": "警戒模式"
        },
        {
            "deviceMode": 1,
            "deviceModeStr": "人脸识别模式"
        }
    ]
},
{
    "deviceModel": 7,
    "deviceModelStr": "MegEye-C4N-888",
    "isSupportActiveSdk": 0,
    "modeList": [
        {
            "deviceMode": 2,
            "deviceModeStr": "结构化模式"
        },
        {
            "deviceMode": 3,
            "deviceModeStr": "警戒模式"
        },
        {
            "deviceMode": 1,
            "deviceModeStr": "人脸识别模式"
        }
    ]
},
{
    "deviceModel": 8,
    "deviceModelStr": "MegEye-C4S-222",
    "isSupportActiveSdk": 1,
    "modeList": [
        {
            "deviceMode": 2,
            "deviceModeStr": "结构化模式"
        },
        {
            "deviceMode": 3,
            "deviceModeStr": "警戒模式"
        },
        {
            "deviceMode": 1,
            "deviceModeStr": "人脸识别模式"
        }
    ]
}
]

```

```

    },
    {
      "deviceModel": 9,
      "deviceModelStr": "MegEye-C4S-322",
      "isSupportActiveSdk": 1,
      "modeList": [
        {
          "deviceMode": 2,
          "deviceModeStr": "结构化模式"
        },
        {
          "deviceMode": 3,
          "deviceModeStr": "警戒模式"
        },
        {
          "deviceMode": 1,
          "deviceModeStr": "人脸识别模式"
        }
      ]
    },
    {
      "deviceModel": 10,
      "deviceModelStr": "MegEye-C4R-122",
      "isSupportActiveSdk": 1,
      "modeList": [
        {
          "deviceMode": 2,
          "deviceModeStr": "结构化模式"
        },
        {
          "deviceMode": 3,
          "deviceModeStr": "警戒模式"
        },
        {
          "deviceMode": 1,
          "deviceModeStr": "人脸识别模式"
        }
      ]
    },
    {
      "deviceModel": 11,
      "deviceModelStr": "MegEye-C4R-222",
      "isSupportActiveSdk": 1,
      "modeList": [
        {
          "deviceMode": 2,
          "deviceModeStr": "结构化模式"
        },
        {
          "deviceMode": 3,
          "deviceModeStr": "警戒模式"
        },
        {
          "deviceMode": 1,
          "deviceModeStr": "人脸识别模式"
        }
      ]
    },
    {
      "deviceModel": 12,
      "deviceModelStr": "MegEye-C4R-322",
      "isSupportActiveSdk": 1,
      "modeList": [
        {
          "deviceMode": 2,
          "deviceModeStr": "结构化模式"
        }
      ]
    }
  ]
}

```

```

        },
        {
            "deviceMode": 3,
            "deviceModeStr": "警戒模式"
        },
        {
            "deviceMode": 1,
            "deviceModeStr": "人脸识别模式"
        }
    ]
},
{
    "deviceModel": 21,
    "deviceModelStr": "IPC",
    "isSupportActiveSdk": 0,
    "modeList": [
        {
            "deviceMode": 2,
            "deviceModeStr": "结构化模式"
        },
        {
            "deviceMode": 3,
            "deviceModeStr": "警戒模式"
        },
        {
            "deviceMode": 1,
            "deviceModeStr": "人脸识别模式"
        }
    ]
}
],
},
{
    "deviceType": 5,
    "deviceTypeStr": "存储服务器",
    "modelList": [
        {
            "deviceModel": 24,
            "deviceModelStr": "MegNVR-N4K-001",
            "isSupportActiveSdk": 0,
            "modeList": []
        },
        {
            "deviceModel": 25,
            "deviceModelStr": "MegNVR-N4M-001",
            "isSupportActiveSdk": 0,
            "modeList": []
        },
        {
            "deviceModel": 26,
            "deviceModelStr": "MegNVR-N8K-281",
            "isSupportActiveSdk": 0,
            "modeList": []
        },
        {
            "deviceModel": 27,
            "deviceModelStr": "MegNVR-N8M-281",
            "isSupportActiveSdk": 0,
            "modeList": []
        },
        {
            "deviceModel": 28,
            "deviceModelStr": "MegNVR-N8P-281",
            "isSupportActiveSdk": 0,
            "modeList": []
        }
    ]
}

```

```
    }
  },
  "msg": "成功"
}
```

1.13. 设备绑定

1.13.1. 基本信息

Path: /v1/api/device/bind

Method: POST

接口描述: 算力设备/存储设备与端设备进行绑定，绑定后可进行布控等操作，是全量绑定，已绑定的设备和入参进行差集运算，得到需要解绑的设备和需要绑定的设备；

1.13.2. 请求参数

Headers

参数名称	参数值	是否必须	示例	备注
Content-Type	application/json	是		

Body

名称	类型	是否必须	备注
uuid	string	必须	当前设备uuid，即设备的唯一标识 deviceToken
deviceUids	string []	必须	绑定的设备uuid列表
└─		非必须	

1.13.3. 返回数据

OK

1.13.4. HTTP请求示例

请求 path

/v1/api/device/bind

请求 body

```
{
  "uuid": "07f6da0f62bb4685bb477c356acae311",
  "deviceUids": [
    "0ee82c3c7b424f20b01e182046c0f16e"
  ]
}
```

1.13.5. HTTP响应示例

响应 200

```
{
  "code": 0,
  "msg": "成功"
}
```

1.14. 设备解绑

1.14.1. 基本信息

Path： /v1/api/device/unbind

Method： POST

接口描述：算力设备/存储设备与端设备进行解绑操作（单一解绑），已布控设备不允许解绑；

1.14.2. 请求参数

Headers

参数名称	参数值	是否必须	示例	备注
Content-Type	application/json	是		

Body

名称	类型	是否必须	备注
uuid	string	必须	设备uuid，即设备的唯一标识 deviceToken
unBindDeviceUuid	string	必须	被解绑设备id

1.14.3. 返回数据

OK

1.14.4. HTTP请求示例

请求 path

```
/v1/api/device/unbind
```

请求 body

```
{
  "uuid": "15ca93618388448ab1afc9115d5a580b",
  "unBindDeviceUuid": "bd022d30511849a29326e51d5959b757"
}
```

1.14.5. HTTP响应示例

响应 200

```
{
  "code": 0,
  "msg": "成功"
}
```

1.15. 查询门禁、人员管控设备列表

1.15.1. 基本信息

Path: /v1/api/device/control/list

Method: POST

接口描述: 查询可添加、已添加的门禁/人员管控设备列表，方便进行添加门禁/人员管控设备；

1.15.2. 请求参数

Headers

参数名称	参数值	是否必须	示例	备注
Content-Type	application/json	是		

Body

名称	类型	是否必须	备注
location	string	非必须	设备位置
pageNum	integer	非必须	当前页码,不传默认1
name	string	非必须	设备名称
pageSize	integer	非必须	分页大小, 不传默认20
deviceType	integer	非必须	设备类型 1、智能摄像头 2、智能面板机 4、网络摄像机
deviceModel	integer	非必须	设备型号（详细型号描述请查看数据字典）与"设备的类型"相对应
queryFlag	integer	非必须	可添加设备查询标识 0 -查询可绑定的列表 1 查询已绑定的设备（添加人员管控设备时）不传递查询全部
flagType	integer	必须	查询标识 0 - 查询门禁点设备 1 - 查询人员管控设备

1.15.3. 返回数据

名称	类型	是否必须	备注
pageNum	integer	必须	当前页码
pageSize	integer	必须	分页大小
total	integer	必须	总数
list	object []	非必须	
└─ deviceUuid	string	必须	设备uuid
└─ location	string	必须	设备位置
└─ name	string	必须	设备名称
└─ deviceType	integer	必须	设备类型 1 智能摄像头 2 智能面板机 4 网络摄像机
└─ deviceModel	integer	必须	设备型号（详细型号描述请查看数据字典）与"设备的类型"相对应
└─ status	integer	必须	设备状态 3、在线 4、离线
└─ switchIp	string	非必须	绑定开关,通行设备列表中使用
└─ isPass	integer	必须	是否绑定门禁设备 0、不是 1、是
└─ personControlFlag	integer	必须	人员管控设备标识 0、不是 1、是

1.15.4. HTTP请求示例

请求 path

```
/v1/api/device/control/list
```

请求 body


```
{
  "queryFlag": 1,
  "flagType": 1
}
```

1.15.5. HTTP响应示例

响应 200

```
{
  "code": 0,
  "data": {
    "list": [
      {
        "deviceModel": 21,
        "deviceType": 4,
        "deviceUuid": "d00fcb3875e0426ead1729bd701fb614",
        "isPass": 1,
        "location": "openapi",
        "name": "c3v-32j",
        "personControlFlag": 1,
        "status": 3
      },
      {
        "deviceModel": 21,
        "deviceType": 4,
        "deviceUuid": "923d3d9a9cc64b7d88c4a8515c370ef5",
        "isPass": 0,
        "location": "openapi",
        "name": "c2",
        "personControlFlag": 1,
        "status": 3
      }
    ],
    "pageNum": 1,
    "pageSize": 20,
    "total": 2
  },
  "msg": "成功"
}
```

1.16. 添加人员管控设备

1.16.1. 基本信息

Path: /v1/api/device/control/add

Method: POST

接口描述: 将已绑定的设备添加人员管控设备，添加完成后设备方可进行通行、安防布控；

1.16.2. 请求参数

Headers

参数名称	参数值	是否必须	示例	备注
Content-Type	application/json	是		

Body

名称	类型	是否必须	备注
removeDeviceUuids	string []	必须	移除人员管控设备id
└─		非必须	
addDeviceUuids	string []	必须	添加人员管控设备id
└─		非必须	

1.16.3. 返回数据

OK

1.16.4. HTTP请求示例

请求 path

/v1/api/device/control/add

请求 body

<pre>{ "removeDeviceUuids": [], "addDeviceUuids": ["de5a5008983e4632bcd1d2d07e16a044", "41d1feb198154927b6baded54ec12482"] }</pre>
--

1.16.5. HTTP响应示例

响应 200

<pre>{ "code": 0, "msg": "成功" }</pre>

1.17. 移除人员管控设备

1.17.1. 基本信息

Path: /v1/api/device/control/remove

Method: POST

接口描述: 将设备从人员管控设备中移除，已布控设备可强制移除；

1.17.2. 请求参数

Headers

参数名称	参数值	是否必须	示例	备注
Content-Type	application/json	是		

Body

名称	类型	是否必须	备注
uuid	string	必须	设备uuid

1.17.3. 返回数据

OK

1.17.4. HTTP请求示例

请求 path

/v1/api/device/control/remove

请求 body

```
{
  "uuid": "95350c2ebf104c3d8a936d2bf39b296e"
}
```

1.17.5. HTTP响应示例

响应 200

```
{
  "code": 0,
  "msg": "成功"
}
```

1.18. 查询可布控设备列表

1.18.1. 基本信息

Path： /v1/api/device/queryDeviceControl

Method： POST

接口描述： 根据设备模式与布控类型查询设备列表信息，可用于查询可以进行布控通行的设备列表；

1.18.2. 请求参数

Headers

参数名称	参数值	是否必须	示例	备注
Content-Type	application/json	是		

Body

名称	类型	是否必须	备注
deviceModes	integer []	必须	设备模式 1-人脸识别 4-人脸抓拍
└		非必须	
queryType	integer	非必须	查询类型 0-查询全部 1-查询全局陌生人员布控，但是面板机要求是有门禁权限 2-查询员工、访客、重点人员布控

1.18.3. 返回数据

名称	类型	是否必须	备注
	object []	非必须	
uuid	string	必须	设备uuid
name	string	必须	设备名称
rtspUrl	string	非必须	视频流地址
type	integer	非必须	是否是all-in-one模式(type=0)下面的警戒模式还是sensor+compute的警戒模式(type=1),其他模式(type=2)
activeMode	integer	非必须	是否支持主动模式 1 主动模式
snCode	String	非必须	设备sn码

1.18.4. HTTP请求示例

请求 path

```
/v1/api/device/queryDeviceControl
```

请求 body

```
{
  "deviceModes": [
    1
  ],
  "queryType": 0
}
```

1.18.5. HTTP响应示例

响应 200

```
{
  "code": 0,
  "data": [{
    "name": "w5k-i8-fake",
    "uuid": "0c55b20861cb4ed19538bd1000000015"
  }, {
    "name": "c3v-32j",
    "rtspUrl": "rtsp://10.172.198.82:8554/koala_C2_26ren_02.264",
    "uuid": "b96f9a1e9f7b431f85ec45a0dae00cf8"
  }, {
    "name": "c2",
    "rtspUrl": "rtsp://10.172.198.82:8554/koala_C2_26ren_01.264",
    "uuid": "7fd010c85d044c72982b931ee4ac2202"
  }
],
  "msg": "成功"
}
```

- 1. 时间计划管理
 - 1.1. 添加时间计划集合
 - 1.1.1. 基本信息
 - 1.1.2. 请求参数
 - 1.1.3. 返回数据
 - 1.1.4. HTTP请求示例
 - 1.1.5. HTTP响应示例
 - 1.2. 修改时间计划
 - 1.2.1. 基本信息
 - 1.2.2. 请求参数
 - 1.2.3. 返回数据
 - 1.2.4. HTTP请求示例
 - 1.2.5. HTTP响应示例
 - 1.3. 批量删除时间计划
 - 1.3.1. 基本信息
 - 1.3.2. 请求参数
 - 1.3.3. 返回数据
 - 1.3.4. HTTP请求示例
 - 1.3.5. HTTP响应示例
 - 1.4. 时间计划列表
 - 1.4.1. 基本信息
 - 1.4.2. 请求参数
 - 1.4.3. 返回数据
 - 1.4.4. HTTP请求示例
 - 1.4.5. HTTP响应示例
 - 1.5. 查询时间计划详情列表
 - 1.5.1. 基本信息
 - 1.5.2. 请求参数
 - 1.5.3. 返回数据
 - 1.5.4. HTTP请求示例
 - 1.5.5. HTTP响应示例

1. 时间计划管理

1.1. 添加时间计划集合

1.1.1. 基本信息

Path: /v1/api/pass/schedule/addBatch

Method: POST

接口描述: 添加时间计划

1.1.2. 请求参数

Headers

参数名称	参数值	是否必须	示例	备注
Content-Type	application/json	是		

Body

名称	类型	是否必须	备注
scheduleList	object []	必须	时间计划添加集合
└─ uuid	string	必须	uuid, 唯一, 长度1-40, 不支持特殊字符
└─ name	string	必须	时间计划名称,唯一, 长度1-32, 不支持特殊字符
└─ holiday	object []	非必须	节假日(事件时间计划不支持假日)
date └─	string	必须	时间: '2020-01-01'
state └─	boolean	必须	状态开关: false 关闭 true 开启
during └─	object []	非必须	持续时间, 如果state为true, 则该参数必须传, 且不能是空集合,参数按照集合里面对象参数说明为准
from └─	string	必须	开始时间, 格式: 00:00
to └─	string	必须	截止时间, 格式: 23:59
└─ workDay	object []	必须	工作日
week └─	integer	必须	星期: 1,2,3,4,5,6,7
during └─	object []	必须	时间段
from └─	string	必须	开始时间, 格式: 00:00
to └─	string	必须	截止时间, 格式: 23:59
└─ type	integer	必须	类型, 1 通行, 3 事件

1.1.3. 返回数据

名称	类型	是否必须	备注
successes	object []	必须	成功列表
└─ uuid	string	必须	uuid
└─ code	integer	非必须	结果状态码
└─ name	string	必须	时间计划名称
└─ holiday	object []	非必须	节假日
date └─	string	必须	时间: '2020-01-01'
state └─	boolean	必须	状态开关: false 关闭 true 开启
during └─	object []	非必须	持续时间, 如果state为true, 则该参数必须传, 且不能是空集合, 参数按照集合里面对象参数说明为准
from └─	string	必须	开始时间, 格式: 00:00
to └─	string	必须	截止时间, 格式: 23:59
└─ msg	string	非必须	结果信息提示
└─ workDay	object []	必须	工作日
week └─	integer	必须	星期: 1,2,3,4,5,6,7
during └─	object []	必须	时间段
from └─	string	必须	开始时间, 格式: 00:00
to └─	string	必须	截止时间, 格式: 23:59

名称	类型	是否必须	备注
└ type	integer	必须	类型, 1 通行, 3 事件
failures	object []	必须	失败列表
└ uuid	string	必须	uuid
└ code	integer	非必须	结果状态码
└ name	string	必须	时间计划名称
└ holiday	object []	非必须	节假日
date └	string	必须	时间: '2020-01-01'
state └	boolean	必须	状态开关: false 关闭 true 开启
during └	object []	非必须	持续时间, 如果state为true, 则该参数必须传, 且不能是空集合, 参数按照集合里面对象参数说明为准
from └	string	必须	开始时间, 格式: 00:00
to └	string	必须	截止时间, 格式: 23:59
└ msg	string	非必须	结果信息提示
└ workDay	object []	必须	工作日
week └	integer	必须	星期: 1,2,3,4,5,6,7
during └	object []	必须	时间段
from └	string	必须	开始时间, 格式: 00:00

名称	类型	是否必须	备注
to └	string	必须	截止时间，格式：23:59
└ type	integer	必须	类型, 1 通行, 3 事件

1.1.4. HTTP请求示例

请求 path

```
/v1/api/pass/schedule/addBatch
```

请求 body

```
{
  "scheduleList": {
    "name": "通行时间计划",
    "type": 1,
    "uuid": "passSchedule",
    "workDay": [
      {
        "week": 1,
        "during": [
          {
            "from": "08:00",
            "to": "10:00"
          },
          {
            "from": "10:30",
            "to": "14:00"
          },
          {
            "from": "15:01",
            "to": "20:00"
          }
        ]
      },
      {
        "week": 2,
        "during": [
          {
            "from": "08:00",
            "to": "10:00"
          },
          {
            "from": "10:30",
            "to": "20:00"
          }
        ]
      },
      {
        "week": 3,
        "during": [
          {
            "from": "08:00",
            "to": "20:00"
          }
        ]
      },
      {
        "week": 4,
        "during": [
          {
            "from": "08:00",
            "to": "10:00"
          },
          {
            "from": "10:30",
            "to": "14:00"
          },
          {
            "from": "15:01",
            "to": "20:00"
          }
        ]
      },
      {
        "week": 5,
        "during": [
```

```
        {
          "from": "08:00",
          "to": "10:00"
        },
        {
          "from": "10:30",
          "to": "19:00"
        }
      ]
    },
    ],
    "holiday": [
      {
        "date": "2020-10-01",
        "during": [
          {
            "from": "08:00",
            "to": "10:00"
          },
          {
            "from": "10:30",
            "to": "14:00"
          },
          {
            "from": "15:01",
            "to": "20:00"
          }
        ]
      },
      {
        "date": "2020-10-02",
        "during": [
          {
            "from": "08:00",
            "to": "10:00"
          },
          {
            "from": "10:30",
            "to": "20:00"
          }
        ]
      },
      {
        "date": "2020-10-03",
        "during": [
          {
            "from": "08:00",
            "to": "20:00"
          }
        ]
      }
    ]
  }
}
```

1.1.5. HTTP响应示例

响应 200

```

{
  "code": 0,
  "data": {
    "failures": [],
    "successes": [
      {
        "holiday": [
          {
            "date": "2020-10-01",
            "during": [
              {
                "from": "08:00",
                "to": "10:00"
              },
              {
                "from": "10:30",
                "to": "14:00"
              },
              {
                "from": "15:01",
                "to": "20:00"
              }
            ],
            "state": true
          },
          {
            "date": "2020-10-02",
            "during": [
              {
                "from": "08:00",
                "to": "10:00"
              },
              {
                "from": "10:30",
                "to": "20:00"
              }
            ],
            "state": true
          },
          {
            "date": "2020-10-03",
            "during": [
              {
                "from": "08:00",
                "to": "20:00"
              }
            ],
            "state": true
          }
        ],
        "name": "通行时间计划",
        "type": 1,
        "uuid": "passSchedule",
        "workDay": [
          {
            "during": [
              {
                "from": "08:00",
                "to": "10:00"
              },
              {
                "from": "10:30",
                "to": "14:00"
              },
              {

```

```

        "from": "15:01",
        "to": "20:00"
    }
},
    "week": 1
},
{
    "during": [
        {
            "from": "08:00",
            "to": "10:00"
        },
        {
            "from": "10:30",
            "to": "20:00"
        }
    ],
    "week": 2
},
{
    "during": [
        {
            "from": "08:00",
            "to": "20:00"
        }
    ],
    "week": 3
},
{
    "during": [
        {
            "from": "08:00",
            "to": "10:00"
        },
        {
            "from": "10:30",
            "to": "14:00"
        },
        {
            "from": "15:01",
            "to": "20:00"
        }
    ],
    "week": 4
},
{
    "during": [
        {
            "from": "08:00",
            "to": "10:00"
        },
        {
            "from": "10:30",
            "to": "19:00"
        }
    ],
    "week": 5
}
    ]
}
    ],
    "msg": "成功"
}

```


1.2. 修改时间计划

1.2.1. 基本信息

Path: /v1/api/pass/schedule/update

Method: POST

接口描述: 修改时间计划

1.2.2. 请求参数

Headers

参数名称	参数值	是否必须	示例	备注
Content-Type	application/json	是		

Body

名称	类型	是否必须	备注
uuid	string	必须	uuid, 唯一, 长度1-40, 不支持特殊字符
name	string	必须	时间计划名称, 唯一, 长度1-32, 不支持特殊字符
holiday	object []	非必须	节假日(事件时间计划不支持假日)
date	string	必须	时间: '2020-01-01'
state	boolean	必须	状态开关: false 关闭 true 开启
during	object []	非必须	持续时间, 如果state为true, 则该参数必须传, 且不能是空集合, 参数按照集合里面对象参数说明为准
from	string	必须	开始时间, 格式: 00:00
to	string	必须	截止时间, 格式: 23:59
workDay	object []	必须	工作日
week	integer	必须	星期: 1,2,3,4,5,6,7
during	object []	必须	时间段
from	string	必须	开始时间, 格式: 00:00
to	string	必须	截止时间, 格式: 23:59

1.2.3. 返回数据

OK

1.2.4. HTTP请求示例

请求 path

通行记录回调

```
/v1/api/pass/schedule/update
```

请求 body

```

{
  "name": "update通行时间计划",
  "uuid": "ff8dc9acb1b6471fbc1bf80fed24e17c",
  "workDay": [
    {
      "week": 1,
      "during": [
        {
          "from": "08:00",
          "to": "10:00"
        },
        {
          "from": "10:30",
          "to": "14:00"
        },
        {
          "from": "15:01",
          "to": "20:00"
        }
      ]
    },
    {
      "week": 2,
      "during": [
        {
          "from": "08:00",
          "to": "10:00"
        },
        {
          "from": "10:30",
          "to": "20:00"
        }
      ]
    },
    {
      "week": 3,
      "during": [
        {
          "from": "08:00",
          "to": "20:00"
        }
      ]
    },
    {
      "week": 4,
      "during": [
        {
          "from": "08:00",
          "to": "10:00"
        },
        {
          "from": "10:30",
          "to": "14:00"
        },
        {
          "from": "15:01",
          "to": "20:00"
        }
      ]
    },
    {
      "week": 5,
      "during": [
        {
          "from": "08:00",

```

```

        "to": "10:00"
      },
      {
        "from": "10:30",
        "to": "19:00"
      }
    ]
  },
  {
    "week": 6,
    "during": [
      {
        "from": "08:00",
        "to": "10:00"
      },
      {
        "from": "10:30",
        "to": "14:00"
      },
      {
        "from": "15:01",
        "to": "20:00"
      }
    ]
  },
  {
    "week": 7,
    "during": [
      {
        "from": "08:00",
        "to": "10:00"
      },
      {
        "from": "10:30",
        "to": "20:00"
      }
    ]
  }
],
"holiday": [
  {
    "date": "2022-10-05",
    "during": [
      {
        "from": "08:00",
        "to": "10:00"
      },
      {
        "from": "10:30",
        "to": "14:00"
      },
      {
        "from": "15:01",
        "to": "20:00"
      }
    ]
  }
],
{
  "date": "2024-08-22",
  "during": [
    {
      "from": "08:00",
      "to": "10:00"
    },
    {
      "from": "10:30",

```

```
        "to": "20:00"
      }
    ],
    {
      "date": "2025-08-15",
      "during": [
        {
          "from": "08:00",
          "to": "20:00"
        }
      ]
    },
    {
      "date": "2023-08-08",
      "during": [
        {
          "from": "08:00",
          "to": "20:00"
        }
      ]
    }
  ]
}
```

1.2.5. HTTP响应示例

响应 200

```
{
  "code": 0,
  "msg": "成功"
}
```

1.3. 批量删除时间计划

1.3.1. 基本信息

Path： /v1/api/pass/schedule/delete

Method： POST

接口描述： 删除时间计划

1.3.2. 请求参数

Headers

参数名称	参数值	是否必须	示例	备注
Content-Type	application/json	是		

Body

名称	类型	是否必须	备注
uuidList	string []	必须	时间计划uuid集合

1.3.3. 返回数据

名称	类型	是否必须	备注
successes	object []	必须	成功列表
└─ uuid	string	必须	uuid
└─ code	integer	非必须	结果状态码
└─ msg	string	非必须	结果信息提示
failures	object []	必须	失败列表
└─ uuid	string	必须	uuid
└─ code	integer	非必须	结果状态码
└─ msg	string	非必须	结果信息提示

1.3.4. HTTP请求示例

请求 path

```
/v1/api/pass/schedule/delete
```

请求 body

```
{
  "uuidList": [
    "faede9d7207e45d687cf2a5591fece52"
  ]
}
```

1.3.5. HTTP响应示例

响应 200

```
{
  "code": 0,
  "data": {
    "failures": [],
    "successes": [
      {
        "uuid": "faede9d7207e45d687cf2a5591fece52"
      }
    ]
  },
  "msg": "成功"
}
```

1.4. 时间计划列表

1.4.1. 基本信息

Path: /v1/api/pass/schedule/page

Method: POST

接口描述: 根据条件查询时间计划

1.4.2. 请求参数

Headers

参数名称	参数值	是否必须	示例	备注
Content-Type	application/json	是		

Body

名称	类型	是否必须	备注
pageNum	integer	非必须	分页页码，默认为 1
name	string	非必须	时间计划名称，支持模糊查询
pageSize	integer	非必须	分页大小，默认为 20
type	integer	必须	用途类型：1 通行模块, 3 事件模块

1.4.3. 返回数据

名称	类型	是否必须	备注
pageNum	integer	必须	当前页码
pageSize	integer	必须	分页大小
total	integer	必须	总数
list	object []	非必须	
└─ uuid	string	必须	uuid
└─ name	string	必须	时间计划名称
└─ type	integer	必须	类型, 1 通行, 3 事件

1.4.4. HTTP请求示例

请求 path

```
/v1/api/pass/schedule/page
```

请求 body

```
{
  "type":1,
  "name":"通行时间计划备用1",
  "pageNum":1,
  "pageSize":15
}
```

1.4.5. HTTP响应示例

响应 200

```
{
  "code": 0,
  "data": {
    "list": [
      {
        "name": "通行时间计划备用1",
        "type": 1,
        "uuid": "d77146e899424cb0970e4dbb3fd408b4"
      }
    ],
    "pageNum": 1,
    "pageSize": 15,
    "total": 1
  },
  "msg": "成功"
}
```

1.5. 查询时间计划详情列表

1.5.1. 基本信息

Path: /v1/api/pass/schedule/details

Method: POST

接口描述: 根据uuid查询时间计划详情

1.5.2. 请求参数

Headers

参数名称	参数值	是否必须	示例	备注
Content-Type	application/json	是		

Body

名称	类型	是否必须	备注
uuidList	string []	必须	时间计划uuid集合

1.5.3. 返回数据

名称	类型	是否必须	备注
	object []	非必须	
└─ uuid	string	必须	uuid
└─ name	string	必须	时间计划名称
└─ workDay	object []	必须	工作日
└─ week	integer	非必须	
└─ during	object []	非必须	
└─ from	string	非必须	
└─ to	string	非必须	
└─ holiday	object []	非必须	节假日
└─ date	string	非必须	
└─ state	boolean	非必须	
└─ during	object []	非必须	
└─ from	string	非必须	
└─ to	string	非必须	
└─ type	integer	必须	类型, 1 通行, 3 事件

1.5.4. HTTP请求示例

请求 path

```
/v1/api/pass/schedule/details
```

请求 body

```
{  
  "uuidList": [  
    "624a6ffde9684236b3629214938bc1a7"  
  ]  
}
```

1.5.5. HTTP响应示例

响应 200

```

{
  "code": 0,
  "data": [
    {
      "holiday": [
        {
          "date": "2020-10-01",
          "during": [
            {
              "from": "08:00",
              "to": "10:00"
            },
            {
              "from": "10:30",
              "to": "14:00"
            },
            {
              "from": "15:01",
              "to": "20:00"
            }
          ],
          "state": true
        },
        {
          "date": "2020-10-02",
          "during": [
            {
              "from": "08:00",
              "to": "10:00"
            },
            {
              "from": "10:30",
              "to": "20:00"
            }
          ],
          "state": true
        },
        {
          "date": "2020-10-03",
          "during": [
            {
              "from": "08:00",
              "to": "20:00"
            }
          ],
          "state": true
        }
      ],
      "name": "通行时间计划备用1",
      "type": 1,
      "uuid": "624a6ffde9684236b3629214938bc1a7",
      "workDay": [
        {
          "during": [
            {
              "from": "08:00",
              "to": "10:00"
            },
            {
              "from": "10:30",
              "to": "14:00"
            },
            {
              "from": "15:01",
              "to": "20:00"
            }
          ]
        }
      ]
    }
  ]
}

```

```
    }
  ],
  "week": 1
},
{
  "during": [
    {
      "from": "08:00",
      "to": "10:00"
    },
    {
      "from": "10:30",
      "to": "20:00"
    }
  ],
  "week": 2
},
{
  "during": [
    {
      "from": "08:00",
      "to": "20:00"
    }
  ],
  "week": 3
},
{
  "during": [
    {
      "from": "08:00",
      "to": "10:00"
    },
    {
      "from": "10:30",
      "to": "14:00"
    },
    {
      "from": "15:01",
      "to": "20:00"
    }
  ],
  "week": 4
},
{
  "during": [
    {
      "from": "08:00",
      "to": "10:00"
    },
    {
      "from": "10:30",
      "to": "19:00"
    }
  ],
  "week": 5
}
]
}
],
"msg": "成功"
}
```

- 1. 门禁点管理
 - 1.1. 查询绑定B3R的端设备列表
 - 1.1.1. 基本信息
 - 1.1.2. 请求参数
 - 1.1.3. 返回数据
 - 1.1.4. HTTP请求示例
 - 1.1.5. HTTP响应示例
 - 1.2. 查询门禁点设备列表
 - 1.2.1. 基本信息
 - 1.2.2. 请求参数
 - 1.2.3. 返回数据
 - 1.2.4. HTTP请求示例
 - 1.2.5. HTTP响应示例
 - 1.3. 查询同步失败人员信息
 - 1.3.1. 基本信息
 - 1.3.2. 请求参数
 - 1.3.3. 返回数据
 - 1.3.4. HTTP请求示例
 - 1.3.5. HTTP响应示例
 - 1.4. 查询绑定设备类型
 - 1.4.1. 基本信息
 - 1.4.2. 请求参数
 - 1.4.3. 返回数据
 - 1.4.4. HTTP请求示例
 - 1.4.5. HTTP响应示例
 - 1.5. 查询设备配置信息
 - 1.5.1. 基本信息
 - 1.5.2. 请求参数
 - 1.5.3. 返回数据
 - 1.5.4. HTTP请求示例
 - 1.5.5. HTTP响应示例
 - 1.6. 批量添加和移除门禁点
 - 1.6.1. 基本信息
 - 1.6.2. 请求参数
 - 1.6.3. 返回数据
 - 1.6.4. HTTP请求示例
 - 1.6.5. HTTP响应示例
 - 1.7. 移除门禁点
 - 1.7.1. 基本信息
 - 1.7.2. 请求参数
 - 1.7.3. 返回数据
 - 1.7.4. HTTP请求示例
 - 1.7.5. HTTP响应示例
 - 1.8. 绑定开关
 - 1.8.1. 基本信息
 - 1.8.2. 请求参数
 - 1.8.3. 返回数据
 - 1.8.4. HTTP请求示例

- 1.8.5. HTTP响应示例
- 1.9. 远程常开
 - 1.9.1. 基本信息
 - 1.9.2. 请求参数
 - 1.9.3. 返回数据
 - 1.9.4. HTTP请求示例
 - 1.9.5. HTTP响应示例
- 1.10. 远程常闭
 - 1.10.1. 基本信息
 - 1.10.2. 请求参数
 - 1.10.3. 返回数据
 - 1.10.4. HTTP请求示例
 - 1.10.5. HTTP响应示例
- 1.11. 远程开门
 - 1.11.1. 基本信息
 - 1.11.2. 请求参数
 - 1.11.3. 返回数据
 - 1.11.4. HTTP请求示例
 - 1.11.5. HTTP响应示例
- 1.12. 恢复正常
 - 1.12.1. 基本信息
 - 1.12.2. 请求参数
 - 1.12.3. 返回数据
 - 1.12.4. HTTP请求示例
 - 1.12.5. HTTP响应示例
- 1.13. 配置设备
 - 1.13.1. 基本信息
 - 1.13.2. 请求参数
 - 1.13.3. 返回数据
 - 1.13.4. HTTP请求示例
 - 1.13.5. HTTP响应示例
- 1.14. 查询同步列表（废弃）
 - 1.14.1. 基本信息
 - 1.14.2. 请求参数
 - 1.14.3. 返回数据
 - 1.14.4. HTTP请求示例
 - 1.14.5. HTTP响应示例
- 1.15. 同步失败详情（废弃）
 - 1.15.1. 基本信息
 - 1.15.2. 请求参数
 - 1.15.3. 返回数据
 - 1.15.4. HTTP请求示例
 - 1.15.5. HTTP响应示例

1. 门禁点管理

1.1. 查询绑定B3R的端设备列表

1.1.1. 基本信息

Path: /v1/api/door/queryB3rInBind

Method: POST

接口描述: 查询在绑定在同一B3R下的设备

1.1.2. 请求参数

Headers

参数名称	参数值	是否必须	示例	备注
Content-Type	application/json	是		

Body

名称	类型	是否必须	备注
pageNum	integer	非必须	当前页码,不传默认1
uuid	string	必须	当前设备uuid
pageSize	integer	非必须	分页大小, 不传默认20

1.1.3. 返回数据

名称	类型	是否必须	备注
pageNum	integer	必须	当前页码
pageSize	integer	必须	分页大小
total	integer	必须	总数
list	object []	非必须	
└─ uuid	string	必须	设备uuid
└─ location	string	非必须	设备位置
└─ name	string	必须	设备名称
└─ deviceType	integer	必须	设备类型: 1、智能摄像头 2、智能面板机 4、网络摄像机
└─ deviceModel	integer	必须	设备型号（详细型号描述请查看数据字典）与"设备的类型"相对应

1.1.4. HTTP请求示例

请求 path

```
/v1/api/door/queryB3rInBind
```

请求 body

```
{
  "pageNum": 1,
  "uuid": "6cd110a091e7454386f6fbd6da59bf7a",
  "pageSize": 5
}
```

1.1.5. HTTP响应示例

响应 200

```
body
{
  "code": 0,
  "data": {
    "list": [
      {
        "deviceModel": 4,
        "deviceType": 1,
        "location": "openapi",
        "name": "c3s-123",
        "uuid": "f80c718df8934c54a0ffde761a45269e"
      },
      {
        "deviceModel": 21,
        "deviceType": 4,
        "location": "openapi",
        "name": "c2",
        "uuid": "6cd110a091e7454386f6fbd6da59bf7a"
      }
    ],
    "pageNum": 1,
    "pageSize": 10,
    "total": 2
  },
  "msg": "成功"
}
```

1.2. 查询门禁点设备列表

1.2.1. 基本信息

Path: /v1/api/door/list

Method: POST

接口描述: 分页查询门禁点设备的信息

1.2.2. 请求参数

Headers

参数名称	参数值	是否必须	示例	备注
Content-Type	application/json	是		

Body

名称	类型	是否必须	备注
location	string	非必须	设备位置
pageNum	integer	非必须	当前页码,不传默认1
name	string	非必须	设备名称
pageSize	integer	非必须	分页大小, 不传默认20
deviceType	integer	非必须	设备类型 1、智能摄像头 2、智能面板机 4、网络摄像机
deviceModel	integer	非必须	设备型号（详细型号描述请查看数据字典）与"设备的类型"相对应
passModel	boolean	必须	true则查询所有可以添加到门禁设备列表的设备（不包含已布控重点人员的设备），false查询已绑定的设备即门禁控制列表信息；【注】添加门禁设备查询接口/v1/api/device/control/list

1.2.3. 返回数据

名称	类型	是否必须	备注
pageNum	integer	必须	当前页码
pageSize	integer	必须	分页大小
total	integer	必须	总数
list	object []	非必须	
└─ uuid	string	必须	设备uuid
└─ location	string	必须	设备位置
└─ name	string	必须	设备名称
└─ deviceType	integer	必须	设备类型 1、智能摄像头 2、智能面板机 4、网络摄像机
└─ deviceModel	integer	必须	设备型号（详细型号描述请查看数据字典）与"设备的类型"相对应
└─ status	integer	必须	设备状态 3 在线 4 离线
└─ synStatus	integer	非必须	同步状态 0 未同步 1 已同步 2 同步失败
└─ switchIp	string	非必须	绑定开关
└─ isPass	integer	必须	是否绑定门禁设备 0 不是 1是
└─ failNum	integer	非必须	同步失败的数量
└─ cmdStatus	integer	非必须	常开常闭状态

1.2.4. HTTP请求示例

请求 path

```
/v1/api/door/list
```

请求 body

```
{
  "passModel": false,
  "location": "OpenAPI测试",
  "pageNum": 1,
  "name": "智能面板机-MegEye-W5K-I8",
  "pageSize": 10,
  "deviceType": 2,
  "deviceModel": 1
}
```

1.2.5. HTTP响应示例

响应 200

```
{
  "code": 0,
  "data": {
    "list": [
      {
        "deviceModel": 1,
        "deviceType": 2,
        "isPass": 1,
        "location": "OpenAPI测试",
        "name": "智能面板机-MegEye-W5K-I8",
        "status": 4,
        "uuid": "8b2e4da9392b4847aff5faf9dcb17a3c"
      }
    ],
    "pageNum": 1,
    "pageSize": 10,
    "total": 1
  },
  "msg": "成功"
}
```

1.3. 查询同步失败人员信息

1.3.1. 基本信息

Path: /v1/api/door/queryFeatureFailList

Method: POST

接口描述: 查询同步设备失败的人员信息

1.3.2. 请求参数

Headers

参数名称	参数值	是否必须	示例	备注
Content-Type	application/json	是		

Body

名称	类型	是否必须	备注
pageNum	integer	非必须	当前页码,不传默认1
noticeCode	string	必须	通知类型: 1-人员照片下发失败, 6-人脸特征抽取失败
pageSize	integer	非必须	分页大小, 不传默认20

1.3.3. 返回数据

名称	类型	是否必须	备注
pageNum	integer	必须	当前页码
pageSize	integer	必须	分页大小
total	integer	必须	总数
list	object []	非必须	
└─ personUuid	string	非必须	人员personUuid
└─ name	string	非必须	人员名
└─ imageUri	string	非必须	用户识别照片的uri
└─ type	integer	非必须	用户类别, 1 员工 2访客 3黑名单, 4 重点人员
└─ reasonStr	String	非必须	失败原因描述

1.3.4. HTTP请求示例**请求 path**

```
/v1/api/door/queryFeatureFailList
```

请求 body

```
{
  "pageNum": 1,
  "pageSize": 5,
  "noticeCode": "1"
}
```

1.3.5. HTTP响应示例

响应 200

```
{
  "code": 0,
  "data": {
    "list": [{
      "imageUri": "http://10.171.4.194:12029/pri/_Y2FjdHVzMTAwX2Zvcml",
      "name": "noface5",
      "personUuid": "66609bb7ca8a47438cd79e0057716714",
      "type": 1
    }, {
      "imageUri": "http://10.171.4.194:12029/pri/_Y2FjdHVzMTAwX2Zvcml",
      "name": "noface2",
      "personUuid": "cda917efcc394fcab8e95db7c82625ae",
      "type": 1,
      "reasonStr": ""
    }
  ],
  "pageNum": 1,
  "pageSize": 5,
  "total": 2
},
  "msg": "成功"
}
```

1.4. 查询绑定设备类型

1.4.1. 基本信息

Path： /v1/api/door/queryBindType

Method： POST

接口描述：

1.4.2. 请求参数

Headers

参数名称	参数值	是否必须	示例	备注
Content-Type	application/json	是		

Body

名称	类型	是否必须	备注
uuid	string	必须	端设备uuid

1.4.3. 返回数据

名称	类型	是否必须	备注
bindType	integer	必须	绑定类型，详情请查看数据字典“绑定类型”

1.4.4. HTTP请求示例

请求 path

```
/v1/api/door/queryBindType
```

请求 body

```
{
  "uuid": "8b2e4da9392b4847aff5faf9dcb17a3c"
}
```

1.4.5. HTTP响应示例

响应 200

```
{'code': 0, 'data': {'bindType': 3}, 'msg': '成功'}
```

1.5. 查询设备配置信息

1.5.1. 基本信息

Path: /v1/api/door/queryConfig

Method: POST

接口描述: 查询已经绑定的设备的一些具体的配置信息，用于查看或者修改设备的配置参数参考“配置设备”接口。

1.5.2. 请求参数

Headers

参数名称	参数值	是否必须	示例	备注
Content-Type	application/json	是		

Body

名称	类型	是否必须	备注
uuid	string	必须	设备uuid，即设备的唯一标识deviceToken

1.5.3. 返回数据




名称	类型	是否必须	
uuid	string	必须	设备uuid
customConfig	object	必须	个性化配置信息
└─ companyName	string	非必须	公司名称，由16个
└─ showPersonName	boolean	非必须	人员名称是否显示
└─ pictureType	integer	非必须	人员照片类型，详
└─ standByTime	integer	非必须	进入待机时间(mir
└─ fillLightness	integer	非必须	通行补光亮度
└─ recognitionInterval	number	非必须	二次识别间隔（秒
└─ rebootSchedule	string	非必须	定时重启 定时重启 期}/{小时}:{分钟}, 日, 取 * 时代表 分隔, 例如: */23 重启, 0,1,2,3,4/5 周三、周四、周五 重启,
└─ alarmInput1	integer	非必须	报警输入, 详情请
└─ alarmInput2	integer	非必须	报警输入, 详情请
└─ alarmOutput1	integer []	非必须	报警输出, 详情请

名称	类型	是否必须	
└		非必须	
└ recognitionTimes	integer	非必须	二次识别次数 0-5
└ lockSwitch	boolean	非必须	门锁控制 true 开 1
└ sensorType	integer	非必须	门磁类型 0 不使用
└ verificationMode	integer	非必须	通行认证方式, 请认证方式"
└ openTimeOut	integer	非必须	开门超时时长, 取
└ openDuration	number	非必须	门禁控制时间, 单位
└ openDelay	number	非必须	延迟门锁控制, 单位
└ wiegandMode	integer	非必须	韦根模式 0:关闭;1
└ wiegandFormat	integer	非必须	韦根格式 0:26 位;
└ wiegandOutput	integer	非必须	韦根输出内容 0:人
└ standbyWpUrl	string	非必须	自定义待机界面图
algorithmConfig	object	必须	人脸参数信息

名称	类型	是否必须	
└─ recognitionMode	integer	非必须	使用模式（面板机 据字典“使用模式”
└─ recognitionThreshold	number	非必须	识别阈值 最大值:
└─ livenessModel	integer	必须	通行活体模式（仅 1 开单目 2开双目
└─ livenessThreshold	number	非必须	活体检测阈值（活 要传） 最大值: 10
└─ maxDistance	integer	非必须	最大识别距离（面 2: 1.5 米; 3: 2
└─ pitchThreshold	number	非必须	垂直角度 0-90度
└─ yawThreshold	number	非必须	水平角度 0-90度
└─ rollThreshold	number	非必须	旋转角度 0-90度
└─ sharpnessThreshold	number	非必须	模糊度 0-1, 精度
└─ smallestFace	number	非必须	最小人脸（非面板
└─ strangerRecognitionThreshold	number	非必须	陌生人识别阈值 量
└─ faceAeMode	boolean	非必须	FACE AE, 开关 t
└─ maskSwitch	boolean	非必须	口罩识别模式, tr

名称	类型	是否必须	
└─ maskThreshold	number	非必须	口罩识别阈值 默认30
temperatureConfig	object	非必须	测温参数
└─ temperatureSwitch	boolean	必须	体温检测开关，默认开启
└─ highTemperatureOff	boolean	必须	高温禁入开关，默认开启，当温度高于阈值时不识别
└─ temperatureMode	boolean	必须	快速测试开关，默认开启，只做测温不做识别
└─ highThreshold	number	必须	高温阈值：默认37.5，范围36.5-38.5
└─ maskAlertSwitch	boolean	必须	未佩戴口罩提醒开关，默认开启，开启时对未佩戴口罩人员报警
└─ unMaskOff	boolean	必须	口罩佩戴禁入开关，默认关闭，true开启，默认未戴口罩人员时不识别
healthkitConfig	object	非必须	健康码参数
└─ healthKitMode	integer	非必须	是否开启健康码功能 0:不开启 1:开启
└─ healthKitProvinceList	object []	非必须	健康码省份列表 [{"cityCode": "100000", "cityName": "北京"}]
└─ cityCode	string	非必须	城市code
└─ cityName	string	非必须	城市名称
└─ healthKitProvince	string	非必须	健康码使用的省份code (cityCode值)

名称	类型	是否必须	
└ healthKitCoordinateMode	integer	非必须	坐标上传开启 0:关闭
└ healthKitLongitude	number	非必须	坐标经度 经度范围: -180.0000~180.0000
└ healthKitLatitude	number	非必须	坐标纬度 纬度范围: -90.0000~90.0000
└ healthKitGreen	integer	非必须	健康码绿码 0:关闭
└ healthKitRed	integer	非必须	健康码红码 0:关闭
└ healthKitYellow	integer	非必须	健康码黄码 0:关闭
└ healthKitUnknown	integer	非必须	健康码未知 0:关闭
└ healthKitTimeout	integer	非必须	健康码超时 0:关闭
└ healthKitImageUrl	string	非必须	二维码图片uri
└ healthKitIdcardVerify	integer	非必须	身份证模块开关 0:关闭
└ healthKitIdcardAutosave	integer	非必须	自动保存身份证号 0:关闭
└ healthKitScheduleVerify	integer	非必须	通行时间权限开关 0:关闭
└ healthKitHighTemperatureNoEntry	integer	非必须	高温禁行开关 0:关闭

名称	类型	是否必须	
 healthKitHighTemperatureThreshold	number	非必须	高温阈值 默认37.
 healthKitServerAddress	string	非必须	健康码平台地址
 healthkitTemperatureSensitivity	integer	非必须	健康码测温灵敏度

1.5.4. HTTP请求示例

请求 path

```
/v1/api/door/queryConfig
```

请求 body

```
{  
  "uuid": "b4d06c793e424e92a2d8bf7005b9e02a"  
}
```

1.5.5. HTTP响应示例

响应 200

```
{
  "code": 0,
  "data": {
    "algorithmConfig": {
      "livenessModel": 1,
      "livenessThreshold": 50,
      "pitchThreshold": 45,
      "recognitionThreshold": 68,
      "rollThreshold": 45,
      "sharpnessThreshold": 0.6,
      "smallestFace": 50,
      "strangerRecognitionThreshold": 60,
      "yawThreshold": 35
    },
    "customConfig": {
      "recognitionInterval": 5,
      "recognitionTimes": 9
    },
    "safeGuardConfig": {
      "livenessModel": 1,
      "livenessThreshold": 50,
      "pitchThreshold": 45,
      "recognitionInterval": 0.5,
      "recognitionThreshold": 69,
      "rollThreshold": 45,
      "sharpnessThreshold": 0.6,
      "smallestFace": 60,
      "strangerRecognitionThreshold": 60,
      "yawThreshold": 35
    },
    "strangerConfig": {
      "pitchThreshold": 45,
      "recognitionInterval": 0.5,
      "recognitionThreshold": 69,
      "rollThreshold": 45,
      "sharpnessThreshold": 0.6,
      "smallestFace": 60,
      "strangerRecognitionThreshold": 60,
      "yawThreshold": 35
    },
    "uuid": "b4d06c793e424e92a2d8bf7005b9e02a"
  },
  "msg": "成功"
}
```

1.6. 批量添加和移除门禁点

1.6.1. 基本信息

Path: /v1/api/door/add

Method: POST

接口描述: 用于批量添加和移除门禁点

1.6.2. 请求参数

Headers

参数名称	参数值	是否必须	示例	备注
Content-Type	application/json	是		

Body

名称	类型	是否必须	备注
addDeviceUuids	string []	必须	新添设备uuid
└─		非必须	
removeDeviceUuids	string []	必须	删除设备uuid
└─		非必须	

1.6.3. 返回数据

OK

1.6.4. HTTP请求示例

请求 path

/v1/api/door/add

请求 body

<pre>{ "addDeviceUuids": ["eb4c9ed7d25348fba14ab429832d818b", "16fee62a23f84321bafc38bca7749896", "51971496c44d4c638aad1636a2dff64f", "f45c0378f8114b5082f1c39376d46556", "bbc1305df66b4b00a23cf8bb3cce8640", "a1233f7f1ae94bb0822d5a8e63b349eb", "51971496c44d4c638aad1636a2dff64f", "8b2e4da9392b4847aff5faf9dcb17a3c"], "removeDeviceUuids": [] }</pre>
--

1.6.5. HTTP响应示例

响应 200

<pre>{ "code": 0, "msg": "成功" }</pre>

1.7. 移除门禁点

1.7.1. 基本信息

Path: /v1/api/door/remove

Method: POST

接口描述:

1.7.2. 请求参数

Headers

参数名称	参数值	是否必须	示例	备注
Content-Type	application/json	是		

Body

名称	类型	是否必须	备注
uuid	string	必须	当前设备uuid, 即设备的唯一标识 deviceToken

1.7.3. 返回数据

OK

1.7.4. HTTP请求示例

请求 path

```
/v1/api/door/remove
```

请求 body

```
{
  "uuid": "8b2e4da9392b4847aff5faf9dcb17a3c"
}
```

1.7.5. HTTP响应示例

响应 200

```
{
  "code": 0,
  "msg": "成功"
}
```

1.8. 绑定开关

1.8.1. 基本信息

Path: /v1/api/door/bindSwitch

Method: POST

接口描述: 绑定开关，不传表示移除门禁开关

1.8.2. 请求参数

Headers

参数名称	参数值	是否必须	示例	备注
Content-Type	application/json	是		

Body

名称	类型	是否必须	备注
switchIp	string	非必须	开关IP,不传则表示移除门禁开关
uuid	string	必须	当前设备uuid，即设备的唯一标识 deviceToken

1.8.3. 返回数据

OK

1.8.4. HTTP请求示例

请求 path

/v1/api/door/bindSwitch

请求 body

```
{
  "uuid": "eb4c9ed7d25348fba14ab429832d818b",
  "switchIp": "10.20.30.40"
}
```

1.8.5. HTTP响应示例

响应 200

```
{
  "code": 0,
  "msg": "成功"
}
```

1.9. 远程常开

1.9.1. 基本信息

Path: /v1/api/door/alwaysOpenDoor

Method: POST

接口描述: 远程控制设备，将面板机设备远程常开，不支持非面板机设备

1.9.2. 请求参数

Headers

参数名称	参数值	是否必须	示例	备注
Content-Type	application/json	是		

Body

名称	类型	是否必须	备注
uuid	string	必须	当前设备uuid

1.9.3. 返回数据

OK

1.9.4. HTTP请求示例

请求 path

```
/v1/api/door/alwaysOpenDoor
```

请求 body

```
{
  "uuid": "51971496c44d4c638aad1636a2dff64f"
}
```

1.9.5. HTTP响应示例

响应 200

面板机

```
{
  "code": 0,
  "msg": "成功"
}
```

非面板机不支持

```
{
  "msg": "目前仅面板机支持远程长开、远程常闭、恢复正常",
  "code": 200314
}
```

1.10. 远程常闭

1.10.1. 基本信息

Path: /v1/api/door/alwaysCloseDoor

Method: POST

接口描述: 远程控制设备，将面板机设备远程常闭，不支持非面板机设备

1.10.2. 请求参数

Headers

参数名称	参数值	是否必须	示例	备注
Content-Type	application/json	是		

Body

名称	类型	是否必须	备注
uuid	string	必须	当前设备uuid

1.10.3. 返回数据

OK

1.10.4. HTTP请求示例

请求 path

/v1/api/door/alwaysCloseDoor

请求 body

```
{
  "uuid": "51971496c44d4c638aad1636a2dff64f"
}
```

1.10.5. HTTP响应示例

响应 200

面板机

```
{
  "code": 0,
  "msg": "成功"
}
```

非面板机不支持

```
{
  "msg": "目前仅面板机支持远程长开、远程常闭、恢复正常",
  "code": 200314
}
```

1.11. 远程开门

1.11.1. 基本信息

Path： /v1/api/door/openDoor

Method： POST

接口描述：远程控制设备，将设备远程开门

1.11.2. 请求参数

Headers

参数名称	参数值	是否必须	示例	备注
Content-Type	application/json	是		

Body

名称	类型	是否必须	备注
uuid	string	必须	当前设备uuid

1.11.3. 返回数据

```
OK
```

1.11.4. HTTP请求示例

请求 path

```
/v1/api/door/openDoor
```

请求 body

```
{
  "uuid": "51971496c44d4c638aad1636a2dff64f"
}
```

1.11.5. HTTP响应示例

响应 200

```
{
  "code": 0,
  "msg": "成功"
}
```

1.12. 恢复正常

1.12.1. 基本信息

Path: /v1/api/door/normal

Method: POST

接口描述: 远程控制设备，将远程常开、远程常闭 的面板机设备恢复正常，不支持非面板机设备

1.12.2. 请求参数

Headers

参数名称	参数值	是否必须	示例	备注
Content-Type	application/json	是		

Body

名称	类型	是否必须	备注
uuid	string	必须	当前设备uuid

1.12.3. 返回数据

OK

1.12.4. HTTP请求示例

请求 path

```
/v1/api/door/normal
```

请求 body

```
{
  "uuid": "51971496c44d4c638aad1636a2dff64f"
}
```

1.12.5. HTTP响应示例

响应 200

面板机

```
{
  "code": 0,
  "msg": "成功"
}
```

非面板机不支持

```
{
  "msg": "目前仅面板机支持远程长开、远程常闭、恢复正常",
  "code": 200314
}
```

1.13. 配置设备

1.13.1. 基本信息

Path: /v1/api/door/config

Method: POST

接口描述： 通过设备的唯一标识deviceToken修改、变更设备的配置参数，先要通过调用“查询设备配置信息”接口获取设备当前的参数配置，然后修改要变更的参数字段，将其他参数一起传入此接口即可， 例如接口入参示例中，修改字段recognitionTimes（二次识别次数），由9修改成8，请查看示例。

1.13.2. 请求参数

Headers

参数名称	参数值	是否必须	示例	备注
Content-Type	application/json	是		

Body



名称	类型	是否必须	
uuid	string	必须	设备uuid
customConfig	object	非必须	门禁参数，用于通
└─ companyName	string	非必须	公司名称，由16个
└─ showPersonName	boolean	非必须	人员名称是否显示
└─ pictureType	integer	非必须	人员照片类型，详
└─ standByTime	integer	非必须	进入待机时间(mir
└─ fillLightness	integer	非必须	通行补光亮度，最
└─ recognitionInterval	number	非必须	二次识别间隔（秒
└─ rebootSchedule	string	非必须	定时重启 定时重 期}/{小时}:{分钟}, 日，取 * 时代: 号分隔，例如：*/ 重启，0,1,2,3,4/5 周三、周四、周五 重启
└─ alarmInput1	integer	非必须	报警输入，详情详 入”
└─ alarmInput2	integer	非必须	报警输入，详情详 入”
└─ alarmOutput1	integer []	非必须	报警输出，详情详 出”

名称	类型	是否必须	
└		非必须	
└ recognitionTimes	integer	非必须	二次识别次数 0-5
└ lockSwitch	boolean	非必须	门锁控制 true 开 1
└ sensorType	integer	非必须	门磁类型 0 不使用
└ verificationMode	integer	非必须	通行认证方式, 请认证方式"
└ openTimeOut	integer	非必须	开门超时时长, 取
└ openDuration	number	非必须	门禁控制时间, 单位
└ openDelay	number	非必须	延迟门锁控制, 取值且只能有一位小数
└ wiegandMode	integer	非必须	韦根模式 0: 关闭; 1
└ wiegandFormat	integer	非必须	韦根格式 0: 26 位; 1
└ wiegandOutput	integer	非必须	韦根输出内容 0: 人
└ standbyWpUrl	string	非必须	自定义待机界面图
algorithmConfig	object	非必须	识别参数, 用于通行

名称	类型	是否必须	
└─ recognitionMode	integer	非必须	使用模式，详情请见“使用模式”
└─ recognitionThreshold	number	非必须	识别阈值 最大值:
└─ livenessModel	integer	非必须	通行活体模式 0 为
└─ livenessThreshold	number	非必须	活体检测阈值 最
└─ maxDistance	integer	非必须	最大识别距离 1: 米
└─ pitchThreshold	number	非必须	垂直角度 0-90度
└─ yawThreshold	number	非必须	水平角度 0-90度
└─ rollThreshold	number	非必须	旋转角度 0-90度
└─ sharpnessThreshold	number	非必须	模糊度 0.1-1.0
└─ smallestFace	number	非必须	最小人脸 1-500
└─ strangerRecognitionThreshold	number	非必须	陌生人识别阈值 最
└─ faceAeMode	boolean	非必须	FACE AE, 开关 t
└─ maskSwitch	boolean	非必须	口罩识别模式, tr

名称	类型	是否必须	
└─ maskThreshold	number	非必须	口罩识别阈值 默认30
temperatureConfig	object	非必须	测温参数
└─ temperatureSwitch	boolean	必须	体温检测开关，默认开启
└─ highTemperatureOff	boolean	必须	高温禁入开关，默认开启，当温度高于阈值时不
└─ temperatureMode	boolean	必须	快速测试开关，默认开启，只做测温不做识别
└─ highThreshold	number	必须	高温阈值：默认37.5，范围36.5-38.5
└─ maskAlertSwitch	boolean	必须	未佩戴口罩提醒开关，默认开启，开启时对未佩戴口罩
└─ unMaskOff	boolean	必须	口罩佩戴禁入开关，默认开启，true开启，默认未戴口罩人员时不
healthkitConfig	object	非必须	健康码参数
└─ healthKitMode	integer	非必须	是否开启健康码功能 0:不开启 1:开启
└─ healthKitProvinceList	object []	非必须	健康码省份列表 [{"cityCode": "100000", "cityName": "北京"}]
└─ cityCode	string	非必须	城市code
└─ cityName	string	非必须	城市名称
└─ healthKitProvince	string	非必须	健康码使用的省份 (cityCode值)

名称	类型	是否必须	
└─ healthKitCoordinateMode	integer	非必须	坐标上传开启 0:关闭
└─ healthKitLongitude	number	非必须	坐标经度 经度范围: -180.0000~180.0000
└─ healthKitLatitude	number	非必须	坐标纬度 纬度范围: -90.0000~90.0000
└─ healthKitGreen	integer	非必须	健康码绿码 0:关闭
└─ healthKitRed	integer	非必须	健康码红码 0:关闭
└─ healthKitYellow	integer	非必须	健康码黄码 0:关闭
└─ healthKitUnknown	integer	非必须	健康码未知 0:关闭
└─ healthKitTimeout	integer	非必须	健康码超时 0:关闭
└─ healthKitImageUrl	string	非必须	二维码图片uri
└─ healthKitIdcardVerify	integer	非必须	身份证模块开关 0:关闭
└─ healthKitIdcardAutosave	integer	非必须	自动保存身份证号 0:关闭
└─ healthKitScheduleVerify	integer	非必须	通行时间权限开关 0:关闭
└─ healthKitHighTemperatureNoEntry	integer	非必须	高温禁行开关 0:关闭

名称	类型	是否必须	
 healthKitHighTemperatureThreshold	number	非必须	高温阈值 默认37.
 healthKitServerAddress	string	非必须	健康码平台地址
 healthkitTemperatureSensitivity	integer	非必须	健康码测温灵敏度

1.13.3. 返回数据

OK

1.13.4. HTTP请求示例

请求 path

/v1/api/door/config

请求 body

```
{
  "uuid": "b4d06c793e424e92a2d8bf7005b9e02a",
  "algorithmConfig": {
    "livenessModel": 1,
    "livenessThreshold": 50,
    "pitchThreshold": 45,
    "recognitionThreshold": 68,
    "rollThreshold": 45,
    "sharpnessThreshold": 0.6,
    "smallestFace": 50,
    "strangerRecognitionThreshold": 60,
    "yawThreshold": 35
  },
  "customConfig": {
    "recognitionInterval": 5,
    "recognitionTimes": 8
  },
  "safeGuardConfig": {
    "livenessModel": 1,
    "livenessThreshold": 50,
    "pitchThreshold": 45,
    "recognitionInterval": 0.5,
    "recognitionThreshold": 69,
    "rollThreshold": 45,
    "sharpnessThreshold": 0.6,
    "smallestFace": 60,
    "strangerRecognitionThreshold": 60,
    "yawThreshold": 35
  },
  "strangerConfig": {
    "pitchThreshold": 45,
    "recognitionInterval": 0.5,
    "recognitionThreshold": 69,
    "rollThreshold": 45,
    "sharpnessThreshold": 0.6,
    "smallestFace": 60,
    "strangerRecognitionThreshold": 60,
    "yawThreshold": 35
  }
}
```

1.13.5. HTTP响应示例

响应 200

```
{
  "code": 0,
  "msg": "成功"
}
```

1.14. 查询同步列表（废弃）

1.14.1. 基本信息

Path: /v1/api/door/querySynList

Method: POST

接口描述： 获取绑定在同一个算力设备下的其他设备

1.14.2. 请求参数






Headers

参数名称	参数值	是否必须	示例	备注
Content-Type	application/json	是		

Body

名称	类型	是否必须	备注
pageNum	integer	非必须	当前页码,不传默认1
uuid	string	必须	当前设备uuid
location	string	必须	设备位置
pageSize	integer	非必须	分页大小, 不传默认20
name	string	必须	设备名称
deviceModel	integer	非必须	设备型号：1 MegEye-W5K-I8、2 MegEye-W4K-E7、3 MegEye-W4K-I51、4 MegEye-C3S-123、5 MegEye-C4H-141、6 MegEye-C4H-241、7 MegEye-C4N-888、8 MegEye-C4S-222、9 MegEye-C4S-322、10 MegEye-C4R-122、11 MegEye-C4R-222、12 MegEye-C4R-322、21 IPC、29 海康摄、31 宇视摄像头、32 科达摄像头、33 华为摄像头、34 MegEye-W4K-I52、35 MegEye-W3K-45S、36 MegEye-W4K-I31、40 MegEye-W5K-EA、41 MegEye-W5K-EA-D、42 MegEye-W5K-EA-T、43 MegEye-W5K-EA-DT

1.14.3. 返回数据

名称	类型	是否必须	备注
pageNum	integer	必须	当前页码
pageSize	integer	必须	分页大小
total	integer	必须	总数
list	object []	非必须	
 uuid	string	必须	设备uuid
 location	string	必须	设备位置
 name	string	必须	设备名称
 deviceType	integer	必须	设备类型 1、智能摄像头 2、智能面板机 4、网络摄像机
 deviceModel	integer	必须	设备型号（详细型号描述请查看数据字典）与"设备的类型"相对应

1.14.4. HTTP请求示例

请求 path

```
/v1/api/door/querySynList
```

请求 body

```
{
  "location": "",
  "pageNum": 1,
  "uuid": "8b2e4da9392b4847aff5faf9dcb17a3c",
  "pageSize": 5,
  "name": "",
  "deviceModel": 1
}
```

1.14.5. HTTP响应示例

响应 200

```
{
  "code": 0,
  "data": {
    "list": [],
    "pageNum": 1,
    "pageSize": 5,
    "total": 0
  },
  "msg": "成功"
}
```

1.15. 同步失败详情（废弃）

1.15.1. 基本信息

Path： /v1/api/door/syncDetail

Method： POST

接口描述： 查询同步设备的详细信息

1.15.2. 请求参数

Headers

参数名称	参数值	是否必须	示例	备注
Content-Type	application/json	是		

Body

名称	类型	是否必须	备注
pageNum	integer	非必须	当前页码,不传默认1
noticeCode	interger	必须	通知类型：1-人员照片下发失败，6-人脸特征抽取失败
pageSize	integer	非必须	分页大小，不传默认20

1.15.3. 返回数据

名称	类型	是否必须	备注
pageNum	integer	必须	当前页码
pageSize	integer	必须	分页大小
total	integer	必须	总数
list	object []	非必须	
└─ uuid	string	必须	personUuid
└─ name	string	必须	人员名
└─ imageUrl	string	非必须	用户识别照片的url
└─ type	integer	必须	用户类别, 1 员工 2 访客 4 重点人员
└─ reasonStr	string	非必须	失败原因描述

1.15.4. HTTP请求示例

请求 path

```
/v1/api/door/syncDetail
```

请求 body

```
{  
  "pageNum": 1,  
  "uuid": "8b2e4da9392b4847aff5faf9dcb17a3c",  
  "pageSize": 5  
}
```

1.15.5. HTTP响应示例

响应 200

```
{
  "code": 0,
  "data": {
    "list": [{
      "imageUrl": "https://10.171.4.194/pub/_Y2FjdHVzMTAwX2ZvcnV2ZXJ",
      "name": "noface5",
      "type": 1,
      "uuid": "66609bb7ca8a47438cd79e0057716714"
    }, {
      "imageUrl": "https://10.171.4.194/pub/_Y2FjdHVzMTAwX2ZvcnV2ZXJ",
      "name": "noface2",
      "type": 1,
      "uuid": "cda917efcc394fcab8e95db7c82625ae"
    }
  ],
  "pageNum": 1,
  "pageSize": 5,
  "total": 2
},
  "msg": "成功"
}
```

- 1. 门禁设备告警
 - 1.1. 查询门禁设备告警列表
 - 1.1.1. 基本信息
 - 1.1.2. 请求参数
 - 1.1.3. 返回数据
 - 1.1.4. HTTP请求示例
 - 1.1.5. HTTP响应示例

1. 门禁设备告警

1.1. 查询门禁设备告警列表

1.1.1. 基本信息

Path: /v1/api/device_alarm/list

Method: POST

接口描述: 分页查询门禁设备告警列表，支持条件搜索；

1.1.2. 请求参数

Headers

参数名称	参数值	是否必须	示例	备注
Content-Type	application/json	是		

Body

名称	类型	是否必须	备注
pageNum	integer	非必须	当前页码,不传默认1
pageSize	integer	非必须	分页大小, 不传默认20
deviceUuids	string []	非必须	设备uuid列表
└		非必须	
deviceName	string	非必须	设备名称, 支持模糊搜索
alarmStatus	integer	非必须	历史报警状态。0 - 报警恢复; 1 - 报警
alarmType	integer	非必须	报警类型。500000 - 存储空间占用超过 90% 报警; 500100 - 防拆报警; 500102 - 密码破解报警; 500104 - 门禁开放超时报警; 500200 - 报警输入报警; 500500 - 消防报警500108 - 测温模块失效;500201 - 人员底库超限
startAlarmTime	string	非必须	报警开始时间(包含)。格式为"2020-01-01 10:00:00", 或者填一个long类型的unix timestamp(毫秒)也行
endAlarmTime	string	非必须	报警结束时间 (不包含) 。格式为"2020-01-01 10:00:00", 或者填一个long类型的unix timestamp(毫秒)也行

1.1.3. 返回数据

名称	类型	是否必须	备注
pageNum	integer	必须	当前页码
pageSize	integer	必须	分页大小
total	integer	必须	总数
list	object []	非必须	
 deviceUuid	string	非必须	设备uuid
 deviceName	string	非必须	设备名称
 deviceLocation	string	非必须	设备位置
 deviceType	integer	非必须	设备的类型。1 - 智能摄像头; 2 - 智能面板机; 4 - 网络摄像机;
 deviceModel	integer	非必须	设备型号 (详细型号描述请查看数据字典) 与"设备的类型"相对应
 deviceStatus	integer	非必须	设备在发送报警信息时的状态。0 - 未激活; 1 - 在线未绑定; 2 - 离线未绑定; 3 - 在线; 4 - 离线; 5 - 解绑中
 alarmStatus	integer	非必须	历史报警状态。0 - 报警恢复; 1 - 报警
 alarmType	integer	非必须	报警类型。500000 - 存储空间占用超过 90% 报警; 500100 - 防拆报警; 500102 - 密码破解报警; 500104 - 门禁开放超时报警; 500200 - 报警输入报警; 500500 - 消防报警 500108 - 测温模块失效; 500201 - 人员底库超限

名称	类型	是否必须	备注
 alarmTime	string	非必须	报警时间。unix时间戳，毫秒
 gmtCreate	string	非必须	记录创建时间。unix时间戳，毫秒
 gmtModified	string	非必须	记录修改时间。unix时间戳，毫秒
 alarmStatusNow	integer	非必须	当前报警状态。0 - 报警恢复; 1 - 报警中

1.1.4. HTTP请求示例

请求 path

```
/v1/api/device_alarm/list
```

请求 body

```
{
  "deviceUuids": [
    "6c37967ca6594070afd9699b07ad55c6"
  ],
  "deviceName": "zxh",
  "alarmStatus": 1,
  "alarmType": 500100,
  "startAlarmTime": "",
  "endAlarmTime": ""
}
```

1.1.5. HTTP响应示例

响应 200

```
{
  "code": 0,
  "data": {
    "list": [
      {
        "alarmStatus": 1,
        "alarmStatusNow": 1,
        "alarmTime": 1626206007000,
        "alarmType": 500100,
        "deviceLocation": "Zxhcewen",
        "deviceModel": 1,
        "deviceName": "zxh",
        "deviceStatus": 3,
        "deviceType": 2,
        "deviceUuid": "6c37967ca6594070afd9699b07ad55c6",
        "gmtCreate": 1626206007000,
        "gmtModified": 1626206007000
      }
    ],
    "pageNum": 1,
    "pageSize": 20,
    "total": 1
  },
  "msg": "成功"
}
```

- 1. 通行权限管理
 - 1.1. 添加通行权限
 - 1.1.1. 基本信息
 - 1.1.2. 请求参数
 - 1.1.3. 返回数据
 - 1.1.4. HTTP请求示例
 - 1.1.5. HTTP响应示例
 - 1.2. 删除通行权限
 - 1.2.1. 基本信息
 - 1.2.2. 请求参数
 - 1.2.3. 返回数据
 - 1.2.4. HTTP请求示例
 - 1.2.5. HTTP响应示例
 - 1.3. 编辑通行权限
 - 1.3.1. 基本信息
 - 1.3.2. 请求参数
 - 1.3.3. 返回数据
 - 1.3.4. HTTP请求示例
 - 1.3.5. HTTP响应示例
 - 1.4. 通行权限列表
 - 1.4.1. 基本信息
 - 1.4.2. 请求参数
 - 1.4.3. 返回数据
 - 1.4.4. HTTP请求示例
 - 1.4.5. HTTP响应示例

1. 通行权限管理

1.1. 添加通行权限

1.1.1. 基本信息

Path: /v1/api/pass/authority/add

Method: POST

接口描述: 添加通行权限

1.1.2. 请求参数

Headers

参数名称	参数值	是否必须	示例	备注
Content-Type	application/json	是		

Body

名称	类型	是否必须	备注
deviceUids	string []	必须	设备uuid列表
groupUids	string []	必须	组uuid列表
scheduleUuid	string	必须	时间计划uuid

1.1.3. 返回数据

OK

1.1.4. HTTP请求示例

请求 path

```
/v1/api/pass/authority/add
```

请求 body

```
{
  "deviceUids": [
    "bdc0e8c22794d6bac22627bbd2c3cc4"
  ],
  "groupUids": [
    "976055a31445c5a583ebc1a53c7514",
    "90a41fddf76a4daa9dae47762ecfd359",
    "127d6dadbf1c4e3f93be34865ad0d135"
  ],
  "scheduleUuid": "5ce434bd925b4d56a8dded0542d02d81"
}
```

1.1.5. HTTP响应示例

响应 200

```
{
  "code": 0,
  "msg": "成功"
}
```

1.2. 删除通行权限

1.2.1. 基本信息

Path: /v1/api/pass/authority/delete

Method: POST

接口描述: 删除通行权限

1.2.2. 请求参数

Headers

参数名称	参数值	是否必须	示例	备注
Content-Type	application/json	是		

Body

名称	类型	是否必须	备注
deviceId	string	必须	设备UUID
scheduleId	string	必须	时间计划UUID

1.2.3. 返回数据

OK

1.2.4. HTTP请求示例

请求 path

```
/v1/api/pass/authority/delete
```

请求 body

```
{
  "deviceId": "bbc1305df66b4b00a23cf8bb3cce8640",
  "scheduleId": "5ce434bd925b4d56a8dded0542d02d81"
}
```

1.2.5. HTTP响应示例

响应 200

```
{
  "code": 0,
  "msg": "成功"
}
```

1.3. 编辑通行权限

1.3.1. 基本信息

Path: /v1/api/pass/authority/update

Method: POST

接口描述：修改通行权限

1.3.2. 请求参数

Headers

参数名称	参数值	是否必须	示例	备注
Content-Type	application/json	是		

Body

名称	类型	是否必须	备注
deviceId	string	必须	设备UUID
oldScheduleUuid	string	必须	操作前的时间计划UUID
newScheduleUuid	string	必须	新修改的时间计划UUID
oldGroupUuids	string []	必须	操作前的组ID数组
newGroupUuids	string []	必须	新修改的组ID数组

1.3.3. 返回数据

OK

1.3.4. HTTP请求示例

请求 path

/v1/api/pass/authority/update

请求 body

<pre>{ "deviceId": "X8vCh9TwXW51efuumZLjW0vIc7r6kVjP", "oldScheduleUuid": "5ce434bd925b4d56a8dded0542d02d81", "newScheduleUuid": "e59b8a92cbbd4341a5f67cfb83715c36", "oldGroupUuids": ["976055aae31445c5a583ebc1a53c7514"], "newGroupUuids": ["976055aae31445c5a583ebc1a53c7514"] }</pre>

1.3.5. HTTP响应示例

响应 200

```
{
  "code": 0,
  "msg": "成功"
}
```

1.4. 通行权限列表

1.4.1. 基本信息

Path: /v1/api/pass/authority/list

Method: POST

接口描述: 根据条件分页查询通行权限

1.4.2. 请求参数

Headers

参数名称	参数值	是否必须	示例	备注
Content-Type	application/json	是		

Body

名称	类型	是否必须	备注
pageNum	integer	非必须	分页页码，默认为 1
pageSize	integer	非必须	分页大小，默认为 20
deviceName	string	非必须	设备名称
deviceUuids	string []	非必须	设备id列表
groupUuid	string	非必须	组uuid
scheduleUuid	string	非必须	时间计划uuid

1.4.3. 返回数据

名称	类型	是否必须	备注
pageNum	integer	非必须	当前页码
pageSize	integer	非必须	分页大小
total	integer	非必须	总数
list	object []	非必须	
├ deviceUuid	string	非必须	设备uuid
├ deviceName	string	非必须	设备名称
├ groups	object []	非必须	组信息
├ groupUuid	string	非必须	组uuid
├ groupName	string	非必须	组名
├ scheduleUuid	string	非必须	时间计划Uuid
├ scheduleName	string	非必须	时间计划名

1.4.4. HTTP请求示例

请求 path

```
/v1/api/pass/authority/list
```

请求 body

```
{
  "pageNum": 1,
  "pageSize": 20,
  "deviceName": null,
  "deviceUuids": [],
  "groupUuid": "976055aee31445c5a583ebc1a53c7514",
  "scheduleUuid": null
}
```

1.4.5. HTTP响应示例

响应 200


```
{
  "code": 0,
  "data": {
    "list": [
      {
        "deviceName": "w5k-i8-fake",
        "deviceUuid": "SXiWVkcWomyldGFRyj4prJQpmoM7pzWP",
        "groups": [
          {
            "groupId": 1,
            "groupName": "员工默认组",
            "groupUuid": "976055aae31445c5a583ebc1a53c7514"
          }
        ],
        "scheduleName": "默认时间计划",
        "scheduleUuid": "5ce434bd925b4d56a8dded0542d02d81"
      }
    ],
    "pageNum": 1,
    "pageSize": 20,
    "total": 1
  },
  "msg": "成功"
}
```

- 1. 通行记录
 - 1.1. 查询通行记录列表
 - 1.1.1. 基本信息
 - 1.1.2. 请求参数
 - 1.1.3. 返回数据
 - 1.1.4. HTTP请求示例
 - 1.1.5. HTTP响应示例

1. 通行记录

1.1. 查询通行记录列表

1.1.1. 基本信息

Path： /v1/api/pass_record/list

Method： POST

接口描述： 分页查询通行记录，支持条件筛选；

1.1.2. 请求参数

Headers

参数名称	参数值	是否必须	示例	备注
Content-Type	application/json	是		

Body

名称	类型	是否必须	备注
pageNum	integer	非必须	当前页码,不传默认1
pageSize	integer	非必须	分页大小, 不传默认20
personUuid	string	非必须	人员uuid
personName	string	非必须	人员姓名, 支持模糊查询
personCode	string	非必须	员工编码
visitPersonName	string	非必须	受访人员姓名, 支持模糊查询
deviceUuids	string []	非必须	设备uuid列表
└		非必须	
deviceName	string	非必须	设备名称, 支持模糊查询
deviceLocation	string	非必须	设备位置, 支持模糊查询
recognitionType	integer	非必须	识别人员类型。1 - 员工; 2 - 普通访客; 3 - 重点人员; 4 - 陌生人; 5 - 未识别; 6 - VIP访客
verificationMode	integer	非必须	验证类型。0 - 人脸; 1 - 人脸或刷卡; 2 - 人脸及刷卡; 3 - 人脸及密码; 4 - 远程开门; 5 - 二维码

名称	类型	是否必须	备注
passType	integer	非必须	通行类型。0 - 未通行; 1 - 通行; 2 - 复合认证未通过
livenessType	integer	非必须	活体结果。0 - 非活体; 1 - 活体
startPassTime	string	非必须	时间段查询, 开始时间 (包含), 格式为"2020-01-01 10:00:00", 或者填一个long类型的unix timestamp(毫秒)也行。
endPassTime	string	非必须	时间段查询, 结束时间 (不包含), 格式为"2020-01-01 10:00:00", 或者填一个long类型的unix timestamp(毫秒)也行
maskType	integer	非必须	口罩状态, 0: 未检测, 1: 已佩戴, 2: 未佩戴
healthyState	integer	非必须	健康码状态 0绿码、1黄码、2红码、3未知、4超时

1.1.3. 返回数据

名称	类型	是否必须	备注
pageNum	integer	必须	当前页码
pageSize	integer	必须	分页大小
total	integer	必须	总数
list	object []	非必须	
└─ personUuid	string	非必须	人员uuid
└─ personName	string	非必须	人员姓名
└─ personType	integer	非必须	人员类型，详情请查看数据字典“人员类型”
└─ personCode	string	非必须	员工编码
└─ visitType	integer	非必须	访客类型，详情请查看数据字典“访客类型”
└─ visitPeronUuid	string	非必须	访问人员的uuid
└─ visitPersonType	integer	非必须	访问人员的类型，详情请查看数据字典“人员类型”
└─ visitPersonCode	string	非必须	访问人员的员工编码
└─ visitPersonName	string	非必须	访问人员的姓名
└─ visitReason	string	非必须	来访原因

名称	类型	是否必须	备注
└─ visitFirm	string	非必须	访客单位
└─ visitNumPlate	string	非必须	访客车牌
└─ deviceSetUuid	string	非必须	device set的uuid
└─ deviceUuid	string	非必须	抓拍设备的uuid
└─ deviceName	string	非必须	抓拍设备的名称
└─ deviceLocation	string	非必须	抓拍设备的位置
└─ deviceModel	integer	非必须	设备型号：智能摄像头，智能面板机，网络摄像机，详情请查看数据字典“设备类型，设备型号”
└─ deviceStatus	integer	非必须	抓拍设备在抓拍时的状态。0 - 未激活; 1 - 在线未绑定; 2 - 离线未绑定; 3 - 在线; 4 - 离线; 5 - 解绑中
└─ recognitionType	integer	非必须	识别人员类型。1 - 员工; 2 - 访客; 3 - 重点人员; 4 - 陌生人; 5 - 未识别; 6 - VIP访客
└─ verificationMode	integer	非必须	验证类型。0 - 人脸; 1 - 人脸或刷卡; 2 - 人脸及刷卡; 3 - 人脸及密码; 4 - 远程开门; 5 - 二维码
└─ passType	integer	非必须	通行类型。0 - 未通行; 1 - 通行; 2 - 复合认证未通过;
└─ livenessType	integer	非必须	活体结果。0 - 非活体; 1 - 活体; 2 - 未知

名称	类型	是否必须	备注
 recognitionScore	number	非必须	识别分数
 livenessScore	number	非必须	活体分数
 passTime	string	非必须	抓拍时间。ISO 8601的精确到秒的格式化时间。形如2020-04-15T17:16:20+08:00
 uuid	string	非必须	本条识别记录的uuid
 personImageUrl	string	非必须	人员底库图片url
 snapshotUrl	string	非必须	抓拍图url
 passTimestamp	integer	非必须	抓拍时间。unix时间戳，毫秒
 recognitionTypeDesc	string	非必须	识别人员类型的描述
 verificationModeDesc	string	非必须	验证类型的描述
 passTypeDesc	string	非必须	通行类型的描述
 temperature	number	非必须	人体温度（未检测的情况下，默认是0.0）
 temperatureType	integer	非必须	人体高温状态，0：未检测，1：正常，2：高温
 temperatureTypeDesc	string	非必须	人体高温状态的描述

名称	类型	是否必须	备注
maskType	integer	非必须	口罩状态, 0: 未检测, 1: 已佩戴, 2: 未佩戴
maskTypeDesc	string	非必须	口罩状态的描述
visitPersonType	integer	非必须	访问人员的类型。1 - 员工; 2 - 访客; 3 - 重点人员
healthyState	integer	非必须	健康码状态 0绿码、1黄码、2红码、3未知、4超时
healthkitPasstype	integer	非必须	0、未通过 1、通过 2、未使用
autograph	string	非必须	个性签名（陌生人没有值）

1.1.4. HTTP请求示例

请求 path

```
/v1/api/pass_record/list
```

请求 body

```
{
  "pageNum": 1,
  "pageSize": 10,
  "personUuid": "5b561102d7ba47458b26b51a07ee36de"
}
```

1.1.5. HTTP响应示例

响应 200


```
{
  "code": 0,
  "data": {
    "list": [{
      "deviceLocation": "openapi",
      "deviceModel": 21,
      "deviceName": "d1s-247",
      "deviceSetUuid": "50929fed98c74921a5ab4bbf26b14c85",
      "deviceUuid": "83ddaf6a033548a6879fcf7e708e0c74",
      "livenessType": 2,
      "maskType": 0,
      "maskTypeDesc": "未检测",
      "passTimestamp": 1629662018000,
      "passType": 1,
      "passTypeDesc": "通行",
      "personCode": "437247",
      "personImageUrl": "http://10.171.4.194/pub/_Y2FjdHVzMTAwX2Zvcml",
      "personName": "通行_员工",
      "personType": 1,
      "personUuid": "5b561102d7ba47458b26b51a07ee36de",
      "recognitionType": 1,
      "recognitionTypeDesc": "员工",
      "snapshotUrl": "http://10.171.4.194/pub/_Y2FjdHVzMTAwXzNt_fec9",
      "temperature": 0.0,
      "temperatureType": 0,
      "temperatureTypeDesc": "未检测",
      "uuid": "44467",
      "verificationMode": 0,
      "verificationModeDesc": "人脸",
      "autograph": "每天都要精彩"
    }
  ],
  "pageNum": 1,
  "pageSize": 10,
  "total": 1
},
  "msg": "成功"
}
```

- 1. 考勤设备管理
 - 1.1. 批量删除考勤设备
 - 1.1.1. 基本信息
 - 1.1.2. 请求参数
 - 1.1.3. 返回数据
 - 1.1.4. HTTP请求示例
 - 1.1.5. HTTP响应示例
 - 1.2. 批量添加考勤设备
 - 1.2.1. 基本信息
 - 1.2.2. 请求参数
 - 1.2.3. 返回数据
 - 1.2.4. HTTP请求示例
 - 1.2.5. HTTP响应示例
 - 1.3. 考勤设备列表查询
 - 1.3.1. 基本信息
 - 1.3.2. 请求参数
 - 1.3.3. 返回数据
 - 1.3.4. HTTP请求示例
 - 1.3.5. HTTP响应示例

1. 考勤设备管理

1.1. 批量删除考勤设备

1.1.1. 基本信息

Path: /v1/api/attendance/device/batchDelete

Method: POST

接口描述: 用于考勤设备的添加、删除和变更，考勤设备为门禁设备的子集，可用于考勤的设备为端设备列如面板机、摄像头以及网络摄像头等（详细请查看数据字典）；未布控的设备可以添加到考勤设备中，需要布控后才能用于考勤。

1.1.2. 请求参数

Headers

参数名称	参数值	是否必须	备注
Content-Type	application/json	是	

Body

名称	类型	是否必须	备注
uuids	string []	必须	考勤设备uuids

1.1.3. 返回数据

名称	类型	是否必须	备注
successes	object []	非必须	成功列表
└─ uuid	string	非必须	考勤设备uuid
└─ code	integer	非必须	
└─ msg	string	非必须	
failures	object []	非必须	失败列表
└─ uuid	string	非必须	考勤设备uuid
└─ code	integer	非必须	
└─ msg	string	非必须	

1.1.4. HTTP请求示例

请求 path

```
/v1/api/attendance/device/batchDelete
```

请求 body

```
{
  'uuids': ['e7d7e9d8f0984cddb099b002997aa59b', 'e0f99441af3a49979e705830e1b4d609']
}
```

1.1.5. HTTP响应示例

响应 200

```
{
  'code': 0,
  'data': {
    'failures': ['e0f99441af3a49979e705830e1b4d609'],
    'successes': [{
      'code': 0,
      'msg': '成功',
      'uuid': 'e7d7e9d8f0984cddb099b002997aa59b'
    }]
  },
  'msg': '成功'
}
```

1.2. 批量添加考勤设备

1.2.1. 基本信息

Path: /v1/api/attendance/device/batchAdd**Method:** POST**接口描述:** 从门禁点设备中选取一个/多个门禁点设备添加为考勤设备。

1.2.2. 请求参数

Headers

参数名称	参数值	是否必须	备注
Content-Type	application/json	是	

Body

名称	类型	是否必须	备注
uuids	string []	必须	门禁点设备uuids

1.2.3. 返回数据

名称	类型	是否必须	备注
successes	object []	非必须	添加成功列表
└─ uuid	string	非必须	门禁点设备uuid
└─ attendanceDeviceUuid	string	非必须	考勤设备uuid, 添加成功时必有
└─ code	integer	非必须	
└─ msg	string	非必须	
failures	object []	非必须	添加失败列表
└─ uuid	string	非必须	门禁点设备uuid
└─ attendanceDeviceUuid	string	非必须	考勤设备uuid, 添加成功时必有
└─ code	integer	非必须	
└─ msg	string	非必须	

1.2.4. HTTP请求示例

请求 path

```
/v1/api/attendance/device/batchAdd
```

请求 body

```
{
  "uuids": ["b1dafa998fde488ea63d5965a14a15d9"]
}
```

1.2.5. HTTP响应示例

响应 200

```
{
  "code": 0,
  "data": {
    "failures": [],
    "successes": [{
      "attendanceDeviceUuid": "e0f99441af3a49979e705830e1b4d609",
      "uuid": "b1dafa998fde488ea63d5965a14a15d9"
    }]
  },
  "msg": "成功"
}
```

1.3. 考勤设备列表查询

1.3.1. 基本信息

Path: /v1/api/attendance/device/list

Method: POST

接口描述: 获取考勤设备列表信息。

1.3.2. 请求参数

Headers

参数名称	参数值	是否必须	备注
Content-Type	application/json	是	

Body

名称	类型	是否必须	备注
pageNum	integer	非必须	当前页码,不传默认1
pageSize	integer	非必须	分页大小, 不传默认20
name	string	非必须	设备名称, 支持模糊查询
location	string	非必须	设备位置, 支持模糊查询
deviceType	integer	非必须	设备类型: 1-智能摄像头、2-智能面板机、4-网络摄像机
deviceModel	integer	非必须	设备型号 (详细型号描述请查看数据字典) 与"设备的类型"相对应

1.3.3. 返回数据

名称	类型	是否必须	备注
pageNum	integer	非必须	当前页码
pageSize	integer	非必须	分页大小
total	integer	非必须	总数
list	object []	非必须	
└─ uuid	string	非必须	考勤设备uuid
└─ name	string	非必须	考勤设备名称
└─ location	string	非必须	设备位置
└─ deviceType	integer	非必须	设备类型: 1-智能摄像头、2-智能面板机、4-网络摄像机
└─ deviceModel	integer	非必须	设备型号（详细型号描述请查看数据字典）与"设备的类型"相对应

1.3.4. HTTP请求示例

请求 path

```
/v1/api/attendance/device/list
```

请求 body

```
{
  "name": "",
  "location": "",
  "deviceModel": null,
  "deviceType": null,
  "pageNum": "1",
  "pageSize": 10
}
```

1.3.5. HTTP响应示例

响应 200

```
{
  "code": 0,
  "data": {
    "list": [{
      "deviceModel": 8,
      "deviceType": 1,
      "location": "测试",
      "name": "智能摄像头-MegEye-C4S-222-人脸抓拍",
      "uuid": "2612f42f1ab6442c861d06bcb9e8e618"
    }
  ],
  "pageNum": 1,
  "pageSize": 10,
  "total": 1
},
  "msg": "成功"
}
```


- 1. 考勤组管理
 - 1.1. 创建考勤组
 - 1.1.1. 基本信息
 - 1.1.2. 请求参数
 - 1.1.3. 返回数据
 - 1.1.4. HTTP请求示例
 - 1.1.5. HTTP响应示例
 - 1.2. 批量删除考勤组
 - 1.2.1. 基本信息
 - 1.2.2. 请求参数
 - 1.2.3. 返回数据
 - 1.2.4. HTTP请求示例
 - 1.2.5. HTTP响应示例
 - 1.3. 更新考勤组
 - 1.3.1. 基本信息
 - 1.3.2. 请求参数
 - 1.3.3. 返回数据
 - 1.3.4. HTTP请求示例
 - 1.3.5. HTTP响应示例
 - 1.4. 考勤员工分页列表查询
 - 1.4.1. 基本信息
 - 1.4.2. 请求参数
 - 1.4.3. 返回数据
 - 1.4.4. HTTP请求示例
 - 1.4.5. HTTP响应示例
 - 1.5. 考勤组分页列表查询
 - 1.5.1. 基本信息
 - 1.5.2. 请求参数
 - 1.5.3. 返回数据
 - 1.5.4. HTTP请求示例
 - 1.5.5. HTTP响应示例
 - 1.6. 考勤组详情查询
 - 1.6.1. 基本信息
 - 1.6.2. 请求参数
 - 1.6.3. 返回数据
 - 1.6.4. HTTP请求示例
 - 1.6.5. HTTP响应示例

1. 考勤组管理

1.1. 创建考勤组

1.1.1. 基本信息

Path: /v1/api/attendance/group/add

Method: POST

接口描述：此接口提供了考勤规则创建以及考勤员工的绑定功能，考勤规则以考勤周、日以及小时的维度提供规则制定；此外考勤组也提供了特殊日期例如法定节假日，公司活动等定制考勤规则；考勤组名称不能重复且一个只能在一个考勤组，人员变动以当前考勤组为准，就考勤规则被覆盖。

1.1.2. 请求参数

Headers

参数名称	参数值	是否必须	备注
Content-Type	application/json	是	

Body

名称	类型	是否必须	
name	string	必须	考勤组名称，唯一，长度2~32字符，不
groupPersons	string []	必须	考勤组员工uuids数组
attendanceTime	object []	必须	考勤时间段
└─ order	integer	必须	考勤时段编号，详情请查看数据字典“考
└─ onSt	string	必须	上班开始时间eg:09:00，需要精确到分
└─ onEt	string	必须	上班结束时间eg:09:00，需要精确到分
└─ offSt	string	必须	下班开始时间eg:09:00，需要精确到分
└─ offEt	string	必须	下班结束时间eg:09:00，需要精确到分
specialDate	object []	非必须	特殊日期（考勤周期weekPlan的辅助设 将来的某一天（或多天）是否打卡；2-1 立即生效，可将考勤组的创建日期设置
└─ date	string	必须	特殊日期，格式为"2021-11-27"
└─ isPunch	integer	非必须	打卡状态，默认不打卡，详情请查看数
weekPlan	string []	必须	考勤周期LIST，注： ['MONDAY','TUESDAY','WEDNESDAY
daySpan	integer	必须	跨天打卡考勤统计标识，详情请查看数

1.1.3. 返回数据

名称	类型	是否必须	
successPerson	string []	非必须	添加成功的人员uuid列表
failurePerson	string []	非必须	添加失败的人员uuid列表
name	string	非必须	考勤组名称
attendanceTime	object []	非必须	考勤时间段
└─ order	integer	非必须	考勤时段编号，详情请查看数据字典“考勤时段”
└─ onSt	string	非必须	上班开始时间eg:09:00，需要精确到分
└─ onEt	string	非必须	上班结束时间eg:09:30，需要精确到分
└─ offSt	string	非必须	下班开始时间eg:011:30，需要精确到分
└─ offEt	string	非必须	下班结束时间eg:12:00，需要精确到分
specialDate	object []	非必须	特殊日期，设置了特殊日期时必有
└─ date	string	非必须	特殊日期，"2021-11-27"
└─ isPunch	integer	非必须	打卡状态，详情请查看数据字典“打卡状态”
weekPlan	string []	非必须	考勤周期LIST，注： ['MONDAY','TUESDAY','WEDNESDAY'

名称	类型	是否必须	
daySpan	integer	非必须	跨天打卡考勤统计标识, 详情请查看数
uuid	string	非必须	考勤组uuid, 添加成功时必有

1.1.4. HTTP请求示例

请求 path

```
/v1/api/attendance/group/add
```

请求 body

```
{
  'name': '考勤4组and跨天第二天',
  'daySpan': 1,
  'groupPersons': ['59615f62a6c8492e877468da97cdaeb2'],
  'attendanceTime': [{
    'onSt': '10:27',
    'onEt': '11:27',
    'offSt': '12:27',
    'offEt': '08:27',
    'order': 1
  }],
  'specialDate': [{
    'date': "2021-11-27",
    'isPunch': 0
  }, {
    'date': "2021-11-27",
    'isPunch': 0
  }],
  'weekPlan': ['MONDAY', 'TUESDAY', 'WEDNESDAY', 'THURSDAY', 'FRIDAY', 'SATU']
}
```

1.1.5. HTTP响应示例

响应 200

```
{
  'code': 0,
  'data': {
    'attendanceTime': [{
      'offEt': '08:27',
      'offSt': '12:27',
      'onEt': '11:27',
      'onSt': '10:27',
      'order': 1
    }
  ],
  'daySpan': 1,
  'failurePerson': [],
  'name': '考勤4组and跨天第二天',
  'specialDate': [{
    'date': '2021-11-27',
    'isPunch': 0
  }, {
    'date': '2021-11-27',
    'isPunch': 0
  }
  ],
  'successePerson': ['59615f62a6c8492e877468da97cdaeb2'],
  'uuid': 'bfa7cc41e25a42ffbb129bc204705d4d',
  'weekPlan': ['MONDAY', 'TUESDAY', 'WEDNESDAY', 'THURSDAY', 'FRIDAY', 'SATURDAY', 'SUNDAY'],
  'msg': '成功'
}
```

1.2. 批量删除考勤组

1.2.1. 基本信息

Path: /v1/api/attendance/group/batchDelete

Method: POST

接口描述: 删除一个/多个考勤组，已经生成的考勤记录和考勤统计结果不受此影响

1.2.2. 请求参数

Headers

参数名称	参数值	是否必须	备注
Content-Type	application/json	是	

Body

名称	类型	是否必须	备注
uuids	string []	必须	考勤组uuids

1.2.3. 返回数据

名称	类型	是否必须	备注
successes	object []	非必须	成功列表
└─ uuid	string	非必须	考勤组uuid
└─ code	integer	非必须	
└─ msg	string	非必须	
failures	object []	非必须	失败列表
└─ uuid	string	非必须	考勤组uuid
└─ code	integer	非必须	
└─ msg	string	非必须	

1.2.4. HTTP请求示例

请求 path

```
/v1/api/attendance/group/batchDelete
```

请求 body

```
{
  'uuids': ['19ec0adec1244605ab6f7b864b53ef9b', '4c7ab506d8e64823835aa9f2cf7']
}
```

1.2.5. HTTP响应示例

响应 200

```
{
  'code': 0,
  'data': {
    'failures': [],
    'successes': [{
      'code': 0,
      'msg': '成功',
      'uuid': '19ec0adec1244605ab6f7b864b53ef9b'
    }, {
      'code': 0,
      'msg': '成功',
      'uuid': '4c7ab506d8e64823835aa9f2cf7b7146'
    }, {
      'code': 0,
      'msg': '成功',
      'uuid': '938f9abc54a64375ab17d54da4993001'
    }
  ]
},
'msg': '成功'
}
```

1.3. 更新考勤组

1.3.1. 基本信息

Path: /v1/api/attendance/group/update

Method: POST

接口描述: 更新考勤组主要提供考勤组名称、考勤规则、考勤人员的更新功能，对已经生成的考勤记录和考勤统计结果不受影响，接下来的考勤以变更后的规则生成考勤记录。

1.3.2. 请求参数

Headers

参数名称	参数值	是否必须	备注
Content-Type	application/json	是	

Body

名称	类型	是否必须	
uuid	string	必须	考勤组主键uuid
name	string	必须	考勤组名称，唯一，长度2~32字符，不
groupPersons	string []	必须	考勤组员工uuids数组
attendanceTime	object []	必须	考勤时间段
└─ order	integer	必须	考勤时段编号，详情请查看数据字典“考
└─ onSt	string	必须	上班开始时间eg:09:00，需要精确到分
└─ onEt	string	必须	上班结束时间eg:09:00，需要精确到分
└─ offSt	string	必须	下班开始时间eg:09:00，需要精确到分
└─ offEt	string	必须	下班结束时间eg:09:00，需要精确到分
specialDate	object []	非必须	特殊日期（考勤周期weekPlan的辅助设 来的某一天（或多天）是否打卡
└─ date	string	必须	特殊日期，格式为"2021-11-27"
└─ isPunch	integer	非必须	打卡状态，详情请查看数据字典“打卡状
weekPlan	string []	必须	考勤周期LIST，注： ['MONDAY','TUESDAY','WEDNESDAY
daySpan	integer	必须	跨天打卡考勤统计标识，详情请查看数

1.3.3. 返回数据

名称	类型	是否必须	
name	string	非必须	考勤组名称
attendanceTime	object []	非必须	考勤时间段
└─ order	integer	非必须	考勤时段编号，详情请查看数据字典“考勤时段”
└─ onSt	string	非必须	上班开始时间eg:09:00，需要精确到分
└─ onEt	string	非必须	上班结束时间eg:09:00，需要精确到分
└─ offSt	string	非必须	下班开始时间eg:09:00，需要精确到分
└─ offEt	string	非必须	下班结束时间eg:09:00，需要精确到分
specialDate	object []	非必须	特殊日期：设置了特殊日期时必有
└─ date	string	非必须	特殊日期，格式为"2021-11-27"
└─ isPunch	integer	非必须	打卡状态，详情请查看数据字典“打卡状态”
weekPlan	string []	非必须	考勤周期LIST，注： ['MONDAY','TUESDAY','WEDNESDAY']
daySpan	integer	非必须	跨天打卡考勤统计标识，详情请查看数据字典“跨天打卡”
successPerson	string []	非必须	添加成功的人员uuid列表

名称	类型	是否必须	
failurePerson	string []	非必须	添加失败的人员uuid列表
uuid	string	非必须	考勤组主键uuid

1.3.4. HTTP请求示例

请求 path

```
/v1/api/attendance/group/update
```

请求 body

```
{
  'uuid': '62a9aa86bc914884884f77adb863a97c',
  'name': 'update考勤3组and3个时段',
  'daySpan': 0,
  'groupPersons': ['0ebea90f7a6f4044b7aa8453975031f1'],
  'attendanceTime': [{
    'onSt': '05:00',
    'onEt': '06:30',
    'offSt': '08:00',
    'offEt': '08:30',
    'order': 1
  }, {
    'onSt': '10:00',
    'onEt': '10:30',
    'offSt': '14:00',
    'offEt': '14:30',
    'order': 2
  }
],
  'specialDate': [{
    'date': "2021-11-27",
    'isPunch': 1
  }, {
    'date': "2021-11-27",
    'isPunch': 1
  }
],
  'weekPlan': ['MONDAY', 'TUESDAY', 'WEDNESDAY', 'THURSDAY', 'FRIDAY']
}
```

1.3.5. HTTP响应示例

响应 200

```
{
  'code': 0,
  'data': {
    'attendanceTime': [{
      'offEt': '08:30',
      'offSt': '08:00',
      'onEt': '06:30',
      'onSt': '05:00',
      'order': 1
    }, {
      'offEt': '14:30',
      'offSt': '14:00',
      'onEt': '10:30',
      'onSt': '10:00',
      'order': 2
    }
  ],
  'daySpan': 0,
  'failurePerson': [],
  'name': 'update考勤3组and3个时段',
  'specialDate': [{
    'date': "2021-11-27",
    'isPunch': 1
  }, {
    'date': "2021-11-27",
    'isPunch': 1
  }
  ],
  'successePerson': ['0ebea90f7a6f4044b7aa8453975031f1'],
  'uuid': '62a9aa86bc914884884f77adb863a97c',
  'weekPlan': ['MONDAY', 'TUESDAY', 'WEDNESDAY', 'THURSDAY', 'FRIDAY']
},
  'msg': '成功'
}
```

1.4. 考勤员工分页列表查询

1.4.1. 基本信息

Path: /v1/api/attendance/group/person/list

Method: POST

接口描述: 获取考勤组/所有考勤组员工信息。

1.4.2. 请求参数

Headers

参数名称	参数值	是否必须	备注
Content-Type	application/json	是	

Body

名称	类型	是否必须	备注
pageNum	integer	非必须	当前页码,不传默认1
pageSize	integer	非必须	分页大小, 不传默认20
personName	string	非必须	员工名称, 支持模糊查询
personCode	string	非必须	员工编号, 支持模糊查询

1.4.3. 返回数据

名称	类型	是否必须	备注
pageNum	integer	非必须	当前页码
pageSize	integer	非必须	分页大小
total	integer	非必须	总数
list	object []	非必须	
├ groupUuid	string	非必须	考勤组uuid
├ groupName	string	非必须	考勤组名称
├ personUuid	string	非必须	员工uuid
├ personName	string	非必须	员工姓名
├ personCode	string	非必须	员工编号

1.4.4. HTTP请求示例

请求 path

```
/v1/api/attendance/group/person/list
```

请求 body

```
{
  'pageNum': 1,
  'pageSize': 15,
  'personName': '',
  'personCode': ''
}
```

1.4.5. HTTP响应示例

响应 200

```
{
  'code': 0,
  'data': {
    'list': [{
      'groupName': '考勤组备用3c',
      'groupUuid': '7da239698cf04f80804dd09772d4992a',
      'personCode': '196682',
      'personName': '考勤备用员工03',
      'personUuid': '194bbb3827a7457bbff8008cb6ac5212'
    }, {
      'groupName': '考勤组备用2b',
      'groupUuid': 'fe147fd0897440d99b0063cccad19f6a',
      'personCode': '582575',
      'personName': '考勤备用员工02',
      'personUuid': 'bf8ddcd90af54eee975f1bdb7d69f8d7'
    }, {
      'groupName': '考勤组备用1a',
      'groupUuid': '3259c3de22914008b7212f419c321e94',
      'personCode': '431575',
      'personName': '考勤备用员工01',
      'personUuid': 'a948148276694a96bc4c2e4c2701b86f'
    }
  ],
  'pageNum': 1,
  'pageSize': 15,
  'total': 3
},
  'msg': '成功'
}
```

1.5. 考勤组分页列表查询

1.5.1. 基本信息

Path: /v1/api/attendance/group/list

Method: POST

接口描述:

1.5.2. 请求参数

Headers

参数名称	参数值	是否必须	示例	备注
Content-Type	application/json	是		

Body

名称	类型	是否必须	备注
pageNum	integer	非必须	当前页码,不传默认1
pageSize	integer	非必须	分页大小, 不传默认20
name	string	非必须	考勤组名称, 支持模糊查询

1.5.3. 返回数据

名称	类型	是否必须	
pageNum	integer	非必须	当前页码
pageSize	integer	非必须	分页大小
total	integer	非必须	总数
list	object []	非必须	
└─ uuid	string	非必须	考勤组uuid
└─ name	string	非必须	考勤组名称
└─ attendanceTime	object []	非必须	考勤时间段
order └─	integer	非必须	考勤时段编号，详情请查看数据字典“考
└─ onSt	string	非必须	上班开始时间eg:09:00，需要精确到分
└─ onEt	string	非必须	上班结束时间eg:09:30，需要精确到分
└─ offSt	string	非必须	下班开始时间eg:11:00，需要精确到分
└─ offEt	string	非必须	下班结束时间eg:12:00，需要精确到分
└─ specialDate	object []	非必须	特殊日期：日期-是否打卡

名称	类型	是否必须	
 date	string	非必须	特殊日期，格式为 "2021-11-27"
isPunch 	integer	非必须	打卡状态，详情请查看数据字典“打卡状
 weekPlan	string []	非必须	考勤周期LIST，注： ['MONDAY','TUESDAY','WEDNESDAY
 daySpan	integer	非必须	跨天打卡考勤统计标识，详情请查看数
 totalPerson	integer	非必须	考勤组总人数

1.5.4. HTTP请求示例

请求 path

```
/v1/api/attendance/group/list
```

请求 body

```
{
  "pageNum": 1,
  "pageSize": 15,
  "name": "考勤组备用1a"
}
```

1.5.5. HTTP响应示例

响应 200

```
{
  "code": 0,
  "data": {
    "list": [{
      "attendanceTime": [{
        "offEt": "13:27",
        "offSt": "12:27",
        "onEt": "11:27",
        "onSt": "10:27",
        "order": 1
      }
    ],
    "daySpan": 0,
    "name": "考勤组备用1a",
    "specialDate": [{
      "isPunch": 1,
      "date": "2021-11-27"
    }
  ],
  "totalPerson": 1,
  "uuid": "788d7394e79f47a28e132acb97845022",
  "weekPlan": ["MONDAY", "TUESDAY", "WEDNESDAY", "THURSDAY", "FR
    }
  ],
  "pageNum": 1,
  "pageSize": 15,
  "total": 1
},
"msg": "成功"
}
```

1.6. 考勤组详情查询

1.6.1. 基本信息

Path: /v1/api/attendance/group/query/details

Method: POST

接口描述: 获取考勤组考勤规则详情和考勤组人数。

1.6.2. 请求参数

Headers

参数名称	参数值	是否必须	备注
Content-Type	application/json	是	

Body

名称	类型	是否必须	备注
uuid	string	必须	考勤组uuid

1.6.3. 返回数据

名称	类型	是否必须	
name	string	非必须	考勤组名称，查询成功时必有
groupPersons	object []	非必须	考勤组人员信息，查询成功时必有
└─ personUuid	string	非必须	员工uuid
└─ personName	string	非必须	员工姓名（仅返回前10人）
attendanceTime	object []	非必须	考勤时间段，查询成功时必有
└─ order	integer	非必须	考勤时段编号，详情请查看数据字典“考
└─ onSt	string	非必须	上班开始时间eg:09:00，需要精确到分
└─ onEt	string	非必须	上班结束时间eg:09:00，需要精确到分
└─ offSt	string	非必须	下班开始时间eg:09:00，需要精确到分
└─ offEt	string	非必须	下班结束时间eg:09:00，需要精确到分
specialDate	object []	非必须	特殊日期，设置了特殊日期且查询成功
└─ date	string	非必须	特殊日期，格式为"2021-11-27"
└─ isPunch	integer	非必须	打卡状态，详情请查看数据字典“打卡状

名称	类型	是否必须	
weekPlan	string []	非必须	考勤周期LIST，注： ['MONDAY','TUESDAY','WEDNESDAY']
daySpan	integer	非必须	跨天打卡考勤统计标识，详情请查看数
totalPerson	integer	非必须	考勤组总人数，查询成功时必有
uuid	string	非必须	考勤组uuid，查询成功时必有
code	integer	非必须	
msg	string	非必须	

1.6.4. HTTP请求示例

请求 path

```
/v1/api/attendance/group/query/details
```

请求 body

```
{
  'uuid': '8ac675e54e30497fac16dab9797643c4'
}
```

1.6.5. HTTP响应示例

响应 200

```
{
  'code': 0,
  'data': {
    'attendanceTime': [{
      'offEt': '13:27',
      'offSt': '12:27',
      'onEt': '11:27',
      'onSt': '10:27',
      'order': 1
    }
  ],
  'daySpan': 0,
  'groupPersons': [{
    'personName': '考勤备用员工01',
    'personUuid': '2daa3b6a383e4800b245de279efa03cb'
  }
  ],
  'name': '考勤组备用1a',
  'specialDate': [{
    'date': "2021-11-27",
    'isPunch': 1
  }
  ],
  'totalPerson': 1,
  'uuid': '8ac675e54e30497fac16dab9797643c4',
  'weekPlan': ['MONDAY', 'TUESDAY', 'WEDNESDAY', 'THURSDAY', 'FRIDAY', 'SATURDAY', 'SUNDAY'],
},
  'msg': '成功'
}
```

- 1. 考勤记录管理
 - 1.1. 员工月考勤记录分页列表查询
 - 1.1.1. 基本信息
 - 1.1.2. 请求参数
 - 1.1.3. 返回数据
 - 1.1.4. HTTP请求示例
 - 1.1.5. HTTP响应示例
 - 1.2. 员工考勤记录日历模式查询
 - 1.2.1. 基本信息
 - 1.2.2. 请求参数
 - 1.2.3. 返回数据
 - 1.2.4. HTTP请求示例
 - 1.2.5. HTTP响应示例
 - 1.3. 考勤记录分页列表查询
 - 1.3.1. 基本信息
 - 1.3.2. 请求参数
 - 1.3.3. 返回数据
 - 1.3.4. HTTP请求示例
 - 1.3.5. HTTP响应示例
 - 1.4. 考勤记录详情分页列表查询
 - 1.4.1. 基本信息
 - 1.4.2. 请求参数
 - 1.4.3. 返回数据
 - 1.4.4. HTTP请求示例
 - 1.4.5. HTTP响应示例
 - 1.5. 考勤记录调整
 - 1.5.1. 基本信息
 - 1.5.2. 请求参数
 - 1.5.3. 返回数据
 - 1.5.4. HTTP请求示例
 - 1.5.5. HTTP响应示例
 - 1.6. 考勤记录回调数据结构
 - 1.6.1. 回调数据示例

1. 考勤记录管理

1.1. 员工月考勤记录分页列表查询

1.1.1. 基本信息

Path: /v1/api/attendance/record/personByMonth/list

Method: POST

接口描述: 获取考勤员工一个月的考勤记录。

1.1.2. 请求参数

Headers

参数名称	参数值	是否必须	备注
Content-Type	application/json	是	

Body

名称	类型	是否必须	备注
pageNum	integer	非必须	当前页码,不传默认1
pageSize	integer	非必须	分页大小, 不传默认20
uuid	string	必须	考勤统计记录uuid, 通过考勤统计记录uuid查询考勤记录

1.1.3. 返回数据

名称	类型	是否必须	备注
pageNum	integer	非必须	当前页码
pageSize	integer	非必须	分页大小
total	integer	非必须	总数
list	object []	非必须	
└─ uuid	string	非必须	考勤记录uuid
└─ personName	string	非必须	员工姓名
└─ personCode	string	非必须	员工编号
└─ attendanceDate	string	非必须	考勤日期，时间戳
└─ attTimeResult	object []	非必须	考勤时段结果：打卡时间&考勤结果
└─ order	integer	非必须	考勤时段编号，详情请查看数据字典“考勤时段编号”
└─ inPunch	string	非必须	上班打卡时间eg:09:00:00，需要精确到秒
└─ outPunch	string	非必须	下班打卡时间eg:11:00:00，需要精确到秒
└─ result	integer	非必须	考勤结果，详情请查看数据字典“考勤结果”
└─ status	integer	非必须	考勤状态，详情请查看数据字典“考勤状态”

1.1.4. HTTP请求示例

请求 path

```
/v1/api/attendance/record/personByMonth/list
```

请求 body


```
{
  'pageNum': 1,
  'pageSize': 20,
  'uuid': '0d03c5c7eb804a8d8e9f873f56c0053f'
}
```

1.1.5. HTTP响应示例

响应 200

```
{
  'code': 0,
  'data': {
    'list': [{
      'attTimeResult': [{
        'inPunch': '15:49:41',
        'order': 1,
        'outPunch': '--',
        'result': 6
      }
    ],
    'attendanceDate': 1622390400000,
    'personCode': '118347',
    'personName': '考勤记录备用员工01',
    'status': 4,
    'uuid': '196d09c65707480f94a5e6876374c099'
  },
  'pageNum': 1,
  'pageSize': 20,
  'total': 1
},
  'msg': '成功'
}
```

1.2. 员工考勤记录日历模式查询

1.2.1. 基本信息

Path： /v1/api/attendance/record/calendar

Method： POST

接口描述：用于日历模式展示数据，以自然月为单位展示。

1.2.2. 请求参数

Headers

参数名称	参数值	是否必须	备注
Content-Type	application/json	是	

Body

名称	类型	是否必须	备注
uuid	string	必须	考勤统计记录uuid，通过考勤统计记录查询考勤记录

1.2.3. 返回数据

名称	类型	是否必须	备注
	object []	非必须	
└─ uuid	string	非必须	考勤记录uuid
└─ personName	string	非必须	员工姓名
└─ personCode	string	非必须	员工编号
└─ attendanceDate	string	非必须	考勤日期，时间戳
└─ attTimeResult	object []	非必须	考勤时段结果：打卡时间&考勤结果
order └─	integer	非必须	考勤时段编号，详情请查看数据字典“考勤时段编号”
inPunch └─	string	非必须	上班打卡时间eg:09:00:00，需要精确到秒
outPunch └─	string	非必须	下班打卡时间eg:11:00:00，需要精确到秒
result └─	integer	非必须	考勤结果，详情请查看数据字典“考勤结果”
└─ status	integer	非必须	考勤状态，详情请查看数据字典“考勤状态”

1.2.4. HTTP请求示例

请求 path

```
/v1/api/attendance/record/calendar
```

请求 body

```
{
  'uuid': '0d03c5c7eb804a8d8e9f873f56c0053f'
}
```

1.2.5. HTTP响应示例

响应 200

```
{
  'code': 0,
  'data': [{
    'attTimeResult': [{
      'inPunch': '15:49:41',
      'order': 1,
      'outPunch': '--',
      'result': 6
    }
  ],
  'attendanceDate': 1622390400000,
  'personCode': '118347',
  'personName': '考勤记录备用员工01',
  'status': 4,
  'uuid': '196d09c65707480f94a5e6876374c099'
},
  'msg': '成功'
}
```

1.3. 考勤记录分页列表查询

1.3.1. 基本信息

Path: /v1/api/attendance/record/list

Method: POST

接口描述: 分页获取考勤记录。

1.3.2. 请求参数

Headers

参数名称	参数值	是否必须	备注
Content-Type	application/json	是	

Body

名称	类型	是否必须	备注
pageNum	integer	非必须	当前页码,不传默认1
pageSize	integer	非必须	分页大小, 不传默认20
personName	string	非必须	员工姓名, 支持模糊查询
personCode	string	非必须	员工编号, 支持模糊查询
startTime	string	非必须	开始时间, 毫秒级时间戳
endTime	string	非必须	截止时间, 毫秒级时间戳

1.3.3. 返回数据

名称	类型	是否必须	备注
pageNum	integer	非必须	当前页码
pageSize	integer	非必须	分页大小
total	integer	非必须	总数
list	object []	非必须	
└─ uuid	string	非必须	考勤记录uuid
└─ personName	string	非必须	员工姓名
└─ personCode	string	非必须	员工编号
└─ attendanceDate	string	非必须	考勤日期，时间戳
└─ attTimeResult	object []	非必须	考勤时段结果：打卡时间&考勤结果
└─ order	integer	非必须	考勤时段编号，详情请查看数据字典“考勤时段编号”
└─ inPunch	string	非必须	上班打卡时间eg:09:00:00，需要精确到秒
└─ outPunch	string	非必须	下班打卡时间eg:11:00:00，需要精确到秒
└─ result	integer	非必须	考勤结果，详情请查看数据字典“考勤结果”
└─ status	integer	非必须	考勤状态，详情请查看数据字典“考勤状态”

1.3.4. HTTP请求示例

请求 path

```
/v1/api/attendance/record/list
```

请求 body

```
{
  "personName": "",
  "startTime": 1622217600000,
  "endTime": 1624895999000,
  "pageNum": 1,
  "pageSize": 20
}
```

1.3.5. HTTP响应示例

响应 200

```

{
  'code': 0,
  'data': {
    'list': [{
      'attTimeResult': [{
        'inPunch': '--',
        'order': 1,
        'outPunch': '--',
        'result': 2
      }, {
        'inPunch': '16:19:04',
        'order': 2,
        'outPunch': '--',
        'result': 0
      }
    ],
    'attendanceDate': 1622390400000,
    'personCode': '994466',
    'personName': '考勤记录备用员工04',
    'status': 4,
    'uuid': '010a8b2f13b84bab8427b69af1a3003a'
  }, {
    'attTimeResult': [{
      'inPunch': '15:52:46',
      'order': 1,
      'outPunch': '--',
      'result': 6
    }
  ],
    'attendanceDate': 1622390400000,
    'personCode': '089162',
    'personName': '考勤记录备用员工01',
    'status': 4,
    'uuid': 'a0b542af2d4046be8169e6ff3f6c30c9'
  }, {
    'attTimeResult': [{
      'inPunch': '--',
      'order': 1,
      'outPunch': '--',
      'result': 2
    }, {
      'inPunch': '--',
      'order': 2,
      'outPunch': '--',
      'result': 2
    }, {
      'inPunch': '15:49:37',
      'order': 3,
      'outPunch': '--',
      'result': 0
    }
  ],
    'attendanceDate': 1622390400000,
    'personCode': '708594',
    'personName': '考勤记录备用员工03',
    'status': 4,
    'uuid': 'abb26b51297542288ad1b5bbd3b8e4ea'
  }, {
    'attTimeResult': [{
      'inPunch': '--',
      'order': 1,
      'outPunch': '--',
      'result': 2
    }, {
      'inPunch': '15:49:42',

```

```
        'order': 2,
        'outPunch': '--',
        'result': 0
      }
    ],
    'attendanceDate': 1622390400000,
    'personCode': '716629',
    'personName': '考勤记录备用员工02',
    'status': 4,
    'uuid': '91127128a4634a4dac4fa5d7453400ae'
  }
],
'pageNum': 1,
'pageSize': 20,
'total': 3
},
'msg': '成功'
}
```

1.4. 考勤记录详情分页列表查询

1.4.1. 基本信息

Path: /v1/api/attendance/record/detail/list

Method: POST

接口描述: 获取考勤记录详细信息，展示考勤时间区间内通行的详细信息。

1.4.2. 请求参数

Headers

参数名称	参数值	是否必须	备注
Content-Type	application/json	是	

Body

名称	类型	是否必须	备注
pageNum	integer	非必须	当前页码,不传默认1
pageSize	integer	非必须	分页大小，不传默认20
uuid	string	必须	考勤记录uuid

1.4.3. 返回数据

名称	类型	是否必须	备注
pageNum	integer	非必须	当前页码
pageSize	integer	非必须	分页大小
total	integer	非必须	总数
list	object []	非必须	
└─ personUuid	string	非必须	员工uuid
└─ personName	string	非必须	员工姓名
└─ personCode	string	非必须	员工编号
└─ deviceName	string	非必须	设备名称
└─ deviceLocation	string	非必须	设备位置
└─ snapshotUri	string	非必须	抓拍照片url
└─ personImageUri	string	非必须	底库照片url
└─ passTime	string	非必须	通行时间，毫秒级时间戳

1.4.4. HTTP请求示例

请求 path

```
/v1/api/attendance/record/detail/list
```

请求 body

```
{
  'pageNum': 1,
  'pageSize': 20,
  'uuid': '010a8b2f13b84bab8427b69af1a3003a'
}
```

1.4.5. HTTP响应示例

响应 200

```
{
  'code': 0,
  'data': {
    'list': [{
      'deviceLocation': 'OpenAPI测试',
      'deviceName': '网络摄像机-MegEye-C4R-222-人脸识别',
      'passTime': 1622449196901,
      'personCode': '994466',
      'personImageUri': 'http://10.171.4.130/pub/_ZzEwMF9mb3JldmVyQn',
      'personName': '考勤记录备用员工04',
      'personUuid': '87a3e7456cc34303af431b79476936a7',
      'snapshotUri': 'http://10.171.4.130/pub/_ZzEwMF82bQ==_1c63b1b7',
    }, {
      'deviceLocation': 'OpenAPI测试',
      'deviceName': '网络摄像机-MegEye-C4R-322-人脸识别',
      'passTime': 1622449194902,
      'personCode': '994466',
      'personImageUri': 'http://10.171.4.130/pub/_ZzEwMF9mb3JldmVyQn',
      'personName': '考勤记录备用员工04',
      'personUuid': '87a3e7456cc34303af431b79476936a7',
      'snapshotUri': 'http://10.171.4.130/pub/_ZzEwMF82bQ==_0d49b7c3',
    }
  ],
  'pageNum': 1,
  'pageSize': 20,
  'total': 4
},
'msg': '成功'
}
```

1.5. 考勤记录调整

1.5.1. 基本信息

Path: /v1/api/attendance/record/adjust

Method: POST

接口描述: 以天为时间单位，针对考勤记录做调整，考勤结果为正常或已经是出勤调整的无法调整，考勤结果和考勤统计实时变更。

1.5.2. 请求参数

Headers

参数名称	参数值	是否必须	备注
Content-Type	application/json	是	

Body

名称	类型	是否必须	备注
uuid	string	必须	考勤记录uuid
adjustType	integer	必须	调整类型，详情请查看数据字典“调整类型”
adjustReason	string	非必须	调整原因，注：调整原因长度为0-50个字符

1.5.3. 返回数据

名称	类型	是否必须	备注
adjustRecordUuid	string	非必须	出勤调整记录uuid，调整成功时必有值
uuid	string	非必须	考勤记录uuid
adjustType	integer	非必须	调整类型，详情请查看数据字典“调整类型”
adjustReason	string	非必须	调整原因

1.5.4. HTTP请求示例

请求 path

```
/v1/api/attendance/record/adjust
```

请求 body

```
{
  'uuid': '34b68646a6634fbea214535685df198c',
  'adjustType': 1,
  'adjustReason': '出勤调整'
}
```

1.5.5. HTTP响应示例

响应 200

```
{
  'code': 0,
  'data': {
    'adjustReason': '出勤调整',
    'adjustRecordUuid': '341e150d24f7484384ba24471abebe55',
    'adjustType': 1,
    'uuid': '34b68646a6634fbea214535685df198c'
  },
  'msg': '成功'
}
```

1.6. 考勤记录回调数据结构

名称	类型	是否必须	备注
uuid	string	非必须	考勤记录uuid
personName	string	非必须	员工姓名
personCode	string	非必须	员工编号
attendanceDate	string	非必须	考勤日期，时间戳
attTimeResult	object []	非必须	考勤时段结果：打卡时间&考勤结果
└─ order	integer	非必须	考勤时段编号，详情请查看数据字典“考勤时段编号”
└─ inPunch	string	非必须	上班打卡时间eg:09:00:00，需要精确到秒
└─ outPunch	string	非必须	下班打卡时间eg:11:00:00，需要精确到秒
└─ result	integer	非必须	考勤结果，详情请查看数据字典“考勤结果”
status	integer	非必须	考勤状态，注：：0-暂无结果（‘-’表示），1-正常，2-旷工，3-出勤调整，4-异常

1.6.1. 回调数据示例

响应 200

```
{
  "attTimeResult": [{
    "adjust": -1,
    "inPunch": "09:00:00",
    "order": 1,
    "outPunch": "10:00:00",
    "period": 2,
    "result": 4
  }],
  "attendanceDate": 1624896000000,
  "personCode": "5677978",
  "personName": "彭斯亮",
  "personUuid": "2a12603fc2db4ebebe6dd8d6004e0c93",
  "status": 4,
  "uuid": "26862b1abeb64c4d869add1d225e5ebe"
}
```

- 1. 考勤统计管理
 - 1.1. 考勤统计记录分页列表查询
 - 1.1.1. 基本信息
 - 1.1.2. 请求参数
 - 1.1.3. 返回数据
 - 1.1.4. HTTP请求示例
 - 1.1.5. HTTP响应示例

1. 考勤统计管理

1.1. 考勤统计记录分页列表查询

1.1.1. 基本信息

Path： /v1/api/attendance/statistic/list

Method： POST

接口描述：统计考勤人员一个月内的考勤状况，分别以考勤时间段、考勤天为时间维度，细粒度的展示了员工的月出勤、日出勤和时段出勤情况以及请假等。

1.1.2. 请求参数

Headers

参数名称	参数值	是否必须	备注
Content-Type	application/json	是	

Body

名称	类型	是否必须	备注
pageNum	integer	非必须	当前页码,不传默认1
pageSize	integer	非必须	分页大小，不传默认20
name	string	非必须	员工姓名 支持模糊查询
groupUuid	string	非必须	考勤组uuid
startTime	string	非必须	查询开始时间，毫秒级时间戳
endTime	string	非必须	查询截止时间，毫秒级时间戳

1.1.3. 返回数据

名称	类型	是否必须	备注
pageNum	integer	非必须	当前页码
pageSize	integer	非必须	分页大小
total	integer	非必须	总数
list	object []	非必须	
└─ uuid	string	非必须	考勤统计记录 uuid
└─ personName	string	非必须	员工姓名
└─ personCode	string	非必须	员工编号
└─ groupName	string	非必须	考勤组名称
└─ workDays	integer	非必须	应出勤天数
└─ normalDays	integer	非必须	正常天数
└─ adjustDays	integer	非必须	调整天数
└─ absentDays	integer	非必须	旷工天数
└─ lateTimes	integer	非必须	迟到次数
└─ leaveEarlyTimes	integer	非必须	早退次数
└─ lateLeaveEarlyTimes	integer	非必须	迟到早退次数
└─ omitPunchTimes	integer	非必须	漏打卡次数
└─ absentTimes	integer	非必须	缺勤次数

1.1.4. HTTP请求示例

请求 path

```
/v1/api/attendance/statistic/list
```

请求 body

```
{
  "name": "",
  "startTime": 1622476800000,
  "endTime": 1625068799000,
  "pageNum": 1,
  "pageSize": 10
}
```


1.1.5. HTTP响应示例

响应 200

```
{
  'code': 0,
  'data': {
    'list': [{
      'absentDays': 0,
      'absentTimes': 0,
      'adjustDays': 0,
      'groupName': '考勤组记录备用1a',
      'lateLeaveEarlyTimes': 0,
      'lateTimes': 0,
      'leaveEarlyTimes': 0,
      'normalDays': 0,
      'omitPunchTimes': 1,
      'personCode': '118347',
      'personName': '考勤记录备用员工01',
      'uuid': '0d03c5c7eb804a8d8e9f873f56c0053f',
      'workDays': 1
    }, {
      'absentDays': 0,
      'absentTimes': 0,
      'adjustDays': 0,
      'groupName': '考勤组记录备用1a',
      'lateLeaveEarlyTimes': 0,
      'lateTimes': 0,
      'leaveEarlyTimes': 0,
      'normalDays': 0,
      'omitPunchTimes': 1,
      'personCode': '089162',
      'personName': '考勤记录备用员工01',
      'uuid': '59418e8ab0ca43579b920ddc70ac3f9d',
      'workDays': 1
    }, {
      'absentDays': 0,
      'absentTimes': 1,
      'adjustDays': 0,
      'groupName': '考勤组记录备用2b',
      'lateLeaveEarlyTimes': 0,
      'lateTimes': 0,
      'leaveEarlyTimes': 0,
      'normalDays': 0,
      'omitPunchTimes': 0,
      'personCode': '279983',
      'personName': '考勤记录备用员工04',
      'uuid': '4cd3bfff54d614ad390704bdff5ad555d',
      'workDays': 1
    }
  ],
  'pageNum': 1,
  'pageSize': 10,
  'total': 3
},
  'msg': '成功'
}
```

- 1. 出勤调整管理
 - 1.1. 出勤调整
 - 1.1.1. 基本信息
 - 1.1.2. 请求参数
 - 1.1.3. 返回数据
 - 1.1.4. HTTP请求示例
 - 1.1.5. HTTP响应示例
 - 1.2. 出勤调整记录分页列表查询
 - 1.2.1. 基本信息
 - 1.2.2. 请求参数
 - 1.2.3. 返回数据
 - 1.2.4. HTTP请求示例
 - 1.2.5. HTTP响应示例
 - 1.3. 更新出勤调整记录
 - 1.3.1. 基本信息
 - 1.3.2. 请求参数
 - 1.3.3. 返回数据
 - 1.3.4. HTTP请求示例
 - 1.3.5. HTTP响应示例

1. 出勤调整管理

1.1. 出勤调整

1.1.1. 基本信息

Path： /v1/api/attendance/adjust/add

Method： POST

接口描述： 出勤调整主要对考勤员工异常的考勤记录进行正常状态修正，且考勤组有该员工时可操作；可修正的状态有补卡、请假、调休与调班；调整的时间范围是自然月5号（包含5号）以前可以修改上一个月异常的考勤记录，5号以后仅修改本月的考勤记录；也可对未来将要生成的考勤结果调整，出勤调整的优先级比正常打卡高；调整实时生效，考勤统计实时更新。

1.1.2. 请求参数

Headers

参数名称	参数值	是否必须	备注
Content-Type	application/json	是	

Body

名称	类型	是否必须	备注
personUuid	string	必须	员工uuid
adjustType	integer	必须	调整类型：1-补卡、2-请假、3-调休、4-调班
adjustTimeBegin	string	必须	调整时段开始时间，毫秒级时间戳
adjustTimeEnd	string	必须	调整时段截止时间，毫秒级时间戳
adjustReason	string	非必须	调整原因，注：调整原因长度为0-50个字符

1.1.3. 返回数据

名称	类型	是否必须	备注
personUuid	string	非必须	员工uuid
adjustType	integer	非必须	调整类型：1-补卡、2-请假、3-调休、4-调班
adjustTimeBegin	string	非必须	调整时段开始时间，毫秒级时间戳
adjustTimeEnd	string	非必须	调整时段截止时间，毫秒级时间戳
adjustReason	string	非必须	调整原因，注：调整事由长度为0-50个字符
uuid	string	非必须	考勤调整记录uuid，出勤调整成功时必有

1.1.4. HTTP请求示例

请求 path

```
/v1/api/attendance/adjust/add
```

请求 body

```
{
  'personUuid': '0b0dc7d3ff5b45e687140301d192957d',
  'adjustType': 1,
  'adjustTimeBegin': 1622434654000,
  'adjustTimeEnd': 1622535454000,
  'adjustReason': 'adjustReason01'
}
```

1.1.5. HTTP响应示例

响应 200

```
{
  'code': 0,
  'data': {
    'adjustReason': 'adjustReason01',
    'adjustTimeBegin': 1622434654000,
    'adjustTimeEnd': 1622535454000,
    'adjustType': 1,
    'personUuid': '0b0dc7d3ff5b45e687140301d192957d',
    'uuid': 'affc2c30d6ea43728730fc06ccfee28c'
  },
  'msg': '成功'
}
```

1.2. 出勤调整记录分页列表查询

1.2.1. 基本信息

Path： /v1/api/attendance/adjust/list

Method： POST

接口描述： 获取调整记录列表。

1.2.2. 请求参数


Headers

参数名称	参数值	是否必须	备注
Content-Type	application/json	是	

Body

名称	类型	是否必须	备注
pageNum	integer	非必须	当前页码,不传默认1
pageSize	integer	非必须	分页大小, 不传默认20
personName	string	非必须	姓名 支持模糊查询
adjustType	integer	非必须	调整类型: 1-补卡、2-请假、3-调休、4-调班
startTime	string	非必须	查询开始时间, 毫秒级时间戳
endTime	string	非必须	查询截止时间, 毫秒级时间戳

1.2.3. 返回数据

名称	类型	是否必须	备注
pageNum	integer	非必须	当前页码
pageSize	integer	非必须	分页大小
total	integer	非必须	总数
list	object []	非必须	
 personName	string	非必须	员工姓名
 personCode	string	非必须	员工编号
 groupName	string	非必须	考勤组名称
 adjustTimeBegin	string	非必须	调整时段开始时间，毫秒级时间戳
 adjustTimeEnd	string	非必须	调整时段截止时间，毫秒级时间戳
 adjustType	integer	非必须	调整类型，详情请查看数据字典“调整类型”
 adjustReason	string	非必须	调整原因

1.2.4. HTTP请求示例

请求 path

```
/v1/api/attendance/adjust/list
```

请求 body

```
{
  "personName": "考勤备用员工",
  "adjustType": 1,
  "pageNum": 1,
  "pageSize": 10,
  "startTime": 1622217600742,
  "endTime": 1624895999742
}
```

1.2.5. HTTP响应示例

响应 200

```
{
  'code': 0,
  'data': {
    'list': [{
      'adjustReason': 'adjustReason04',
      'adjustTimeBegin': 1622434654000,
      'adjustTimeEnd': 1622535454000,
      'adjustType': 4,
      'groupName': '考勤组备用1a',
      'personCode': '323183',
      'personName': '考勤备用员工01'
    }, {
      'adjustReason': 'adjustReason03',
      'adjustTimeBegin': 1622434654000,
      'adjustTimeEnd': 1622535454000,
      'adjustType': 3,
      'groupName': '考勤组备用3c',
      'personCode': '636041',
      'personName': '考勤备用员工03'
    }, {
      'adjustReason': 'adjustReason02',
      'adjustTimeBegin': 1622434654000,
      'adjustTimeEnd': 1622535454000,
      'adjustType': 2,
      'groupName': '考勤组备用2b',
      'personCode': '134634',
      'personName': '考勤备用员工02'
    }
  ],
  'pageNum': 1,
  'pageSize': 10,
  'total': 3
},
  'msg': '成功'
}
```

1.3. 更新出勤调整记录

1.3.1. 基本信息

Path: /v1/api/attendance/adjust/update

Method: POST

接口描述: 对已经调整的调整记录进行修改，此操作同出勤调整添加操作，但是对已经生效的出勤调整产生影响，从而导致考勤结果和考勤统计记录的变化；具体表现为两次调整区间相交部分，相交的时间区间内的考勤解果不会有影响，不相交区间内的考勤结果的若是被第一次考勤调整则将被还原成调整前的状态，未被第一调整的时间区间内的考勤结果将被调整。

1.3.2. 请求参数

Headers

参数名称	参数值	是否必须	备注
Content-Type	application/json	是	

Body

名称	类型	是否必须	备注
uuid	string	必须	出行调整记录主键uuid
adjustType	integer	必须	调整类型，详情请查看数据字典“调整类型”
adjustTimeBegin	string	必须	调整时段开始时间
adjustTimeEnd	string	必须	调整时段截止时间
adjustReason	string	非必须	调整事由

1.3.3. 返回数据

名称	类型	是否必须	备注
code	string	非必须	消息提示编码
msg	string	非必须	消息提示

1.3.4. HTTP请求示例

请求 path

```
/v1/api/attendance/adjust/update
```

请求 body

```
{
  'personUuid': '2fd98a319ea04fe497b786bc1df68002',
  'adjustType': 1,
  'adjustTimeBegin': 1622434655000,
  'adjustTimeEnd': 1622535455000,
  'adjustReason': 'adjustReason01'
}
```

1.3.5. HTTP响应示例

响应 200

通行记录回调

```
{  
  'code': 0,  
  'msg': '成功'  
}
```

- 1. 功能配置
 - 1.1. 获取一脸通配置
 - 1.1.1. 基本信息
 - 1.1.2. 请求参数
 - 1.1.3. 返回数据
 - 1.1.4. HTTP请求示例
 - 1.1.5. HTTP响应示例
 - 1.2. 重置一脸通配置
 - 1.2.1. 基本信息
 - 1.2.2. 请求参数
 - 1.2.3. 返回数据
 - 1.2.4. HTTP请求示例
 - 1.2.5. HTTP响应示例
 - 1.3. 一脸通配置保存
 - 1.3.1. 基本信息
 - 1.3.2. 请求参数
 - 1.3.3. 返回数据
 - 1.3.4. HTTP请求示例
 - 1.3.5. HTTP响应示例
 - 1.4. 获取二维码配置
 - 1.4.1. 基本信息
 - 1.4.2. 请求参数
 - 1.4.3. 返回数据
 - 1.4.4. HTTP请求示例
 - 1.4.5. HTTP响应示例
 - 1.5. 保存二维码配置
 - 1.5.1. 基本信息
 - 1.5.2. 请求参数
 - 1.5.3. 返回数据
 - 1.5.4. HTTP请求示例
 - 1.5.5. HTTP响应示例
 - 1.6. 查看入库标准
 - 1.6.1. 基本信息
 - 1.6.2. 请求参数
 - 1.6.3. 返回数据
 - 1.6.4. HTTP请求示例
 - 1.6.5. HTTP响应示例
 - 1.7. 保存入库标准
 - 1.7.1. 基本信息
 - 1.7.2. 请求参数
 - 1.7.3. 返回数据
 - 1.7.4. HTTP请求示例
 - 1.7.5. HTTP响应示例
 - 1.8. 重置入库标准
 - 1.8.1. 基本信息
 - 1.8.2. 请求参数
 - 1.8.3. 返回数据
 - 1.8.4. HTTP请求示例

- 1.8.5. HTTP响应示例
- 1.9. 测温、口罩、健康码开关查询
 - 1.9.1. 基本信息
 - 1.9.2. 请求参数
 - 1.9.3. 返回数据
 - 1.9.4. HTTP请求示例
 - 1.9.5. HTTP响应示例

1. 功能配置

1.1. 获取一脸通配置

1.1.1. 基本信息

Path: /v1/api/sys/config/queryAFaceConfig

Method: POST

接口描述: 获取考勤记录回调地址、通行记录回调地址、测温开关、口罩开关、健康码开关配置

1.1.2. 请求参数

Headers

参数名称	参数值	是否必须	示例	备注
Content-Type	application/json	是		

1.1.3. 返回数据

名称	类型	是否必须	备注
attendanceRecordCallbackUrl	string	必须	考勤管理-考勤记录会回调地址
identifyRecordCallbackUrl	string	必须	门禁参数-识别记录回调地址
temperatureSwitch	integer	必须	门禁参数-测温开关-v1.0.1新增, 0: 关闭, 1: 打开
maskAlertSwitch	integer	必须	口罩提示开关, 0: 关闭, 1: 打开
healthKitSwitch	integer	必须	健康码检测开关, 0: 关闭, 1: 打开

1.1.4. HTTP请求示例

请求 path

```
/v1/api/sys/config/queryAFaceConfig
```

请求 body

```
{}
```

1.1.5. HTTP响应示例

响应 200

```
{
  "code": 0,
  "data": {
    "identifyRecordCallbackUrl": "https://www.megvii.com/",
    "attendanceRecordCallbackUrl": "https://www.megvii.com/",
    "temperatureSwitch": 1,
    "maskAlertSwitch": 1,
    "healthKitSwitch": 1
  },
  "msg": "成功"
}
```

1.2. 重置一脸通配置

1.2.1. 基本信息

Path: /v1/api/sys/config/resetAFaceConfig

Method: POST

接口描述: 重置考勤记录回调地址、通行记录回调地址、测温开关、口罩开关、健康码开关配置至默认状态

1.2.2. 请求参数

Headers

参数名称	参数值	是否必须	示例	备注
Content-Type	application/json	是		

1.2.3. 返回数据

名称	类型	是否必须	备注
attendanceRecordCallbackUrl	string	必须	考勤管理-考勤记录会回调地址
identifyRecordCallbackUrl	string	必须	门禁参数-识别记录回调地址
temperatureSwitch	integer	必须	门禁参数-测温开关-v1.0.1新增, 0: 关闭, 1: 打开
maskAlertSwitch	integer	必须	口罩提示开关, 0: 关闭, 1: 打开
healthKitSwitch	integer	必须	健康码检测开关, 0: 关闭, 1: 打开

1.2.4. HTTP请求示例

请求 path

```
/v1/api/sys/config/resetAFaceConfig
```

请求 body

```
{}
```

1.2.5. HTTP响应示例

响应 200

```
{
  "code": 0,
  "data": {
    "attendanceRecordCallbackUrl": "",
    "healthKitSwitch": 0,
    "identifyRecordCallbackUrl": "",
    "maskAlertSwitch": 0,
    "temperatureSwitch": 0
  },
  "msg": "成功"
}
```

1.3. 一脸通配置保存

1.3.1. 基本信息

Path: /v1/api/sys/config/saveAFaceConfig**Method:** POST

接口描述: 一脸通配置保存, 可进行访客管理、考勤管理、门禁管理、防疫测温参数配置;

1.3.2. 请求参数

Headers

参数名称	参数值	是否必须	示例	备注
Content-Type	application/json	是		

Body

名称	类型	是否必须	备注
attendanceRecordCallbackUrl	string	非必须	考勤管理-考勤记录会回调地址
identifyRecordCallbackUrl	string	非必须	门禁参数-识别记录回调地址
temperatureSwitch	integer	必须	门禁参数-测温开关-v1.0.1新增, 0: 关闭, 1: 打开
maskAlertSwitch	integer	必须	口罩提示开关, 0: 关闭, 1: 打开
healthKitSwitch	integer	必须	健康码检测开关, 0: 关闭, 1: 打开, 默认为关闭

1.3.3. 返回数据

名称	类型	是否必须	备注
attendanceRecordCallbackUrl	string	必须	考勤管理-考勤记录会回调地址
identifyRecordCallbackUrl	string	必须	门禁参数-识别记录回调地址
temperatureSwitch	integer	必须	门禁参数-测温开关-v1.0.1新增，0：关闭，1：打开
maskAlertSwitch	integer	必须	口罩提示开关，0：关闭，1：打开
healthKitSwitch	integer	必须	健康码检测开关，0：关闭，1：打开

1.3.4. HTTP请求示例

请求 path

```
/v1/api/sys/config/saveAFaceConfig
```

请求 body

```
{
  "identifyRecordCallbackUrl":"https://www.megvii.com/",
  "attendanceRecordCallbackUrl":"https://www.megvii.com/",
  "temperatureSwitch":1,
  "maskAlertSwitch":1,
  "healthKitSwitch":1
}
```

1.3.5. HTTP响应示例

响应 200

```
{
  "code":0,
  "data":{
    "identifyRecordCallbackUrl":"https://www.megvii.com/",
    "attendanceRecordCallbackUrl":"https://www.megvii.com/",
    "temperatureSwitch":1,
    "maskAlertSwitch":1,
    "healthKitSwitch":1
  },
  "msg":"成功"
}
```

1.4. 获取二维码配置

1.4.1. 基本信息

Path: /v1/api/sys/queryQrcodeExpireTimeConfigInfo

Method: POST

接口描述: 获取人员配置中员工、访客二维码有效时长配置

1.4.2. 请求参数

Headers

参数名称	参数值	是否必须	示例	备注
Content-Type	application/json	是		

1.4.3. 返回数据

名称	类型	是否必须	备注
visitorQrcodeExpireTime	integer	非必须	访客二维码失效时间，单位分钟，默认10
staffQrcodeExpireTime	integer	非必须	员工二维码失效时间，单位分钟，默认10

1.4.4. HTTP请求示例

请求 path

```
/v1/api/sys/queryQrcodeExpireTimeConfigInfo
```

请求 body

```
{}
```

1.4.5. HTTP响应示例

响应 200


```
{
  "code":0,
  "data":{
    "visitorQrCodeExpireTime":10,
    "staffQrCodeExpireTime":10
  },
  "msg":"成功"
}
```

1.5. 保存二维码配置

1.5.1. 基本信息

Path： /v1/api/sys/saveQrcodeExpireTimeConfigInfo

Method： POST

接口描述： 保存人员配置中员工、访客二维码有效时长配置

1.5.2. 请求参数

Headers

参数名称	参数值	是否必须	示例	备注
Content-Type	application/json	是		

Body

名称	类型	是否必须	备注
visitorQrCodeExpireTime	integer	非必须	访客二维码失效时间，单位分钟，默认10，范围是5-1440之间的整数
staffQrCodeExpireTime	integer	非必须	员工二维码失效时间，单位分钟，默认10，范围是5-1440之间的整数

1.5.3. 返回数据

OK

1.5.4. HTTP请求示例

请求 path

/v1/api/sys/saveQrcodeExpireTimeConfigInfo

请求 body

```
{
  "visitorQrCodeExpireTime":10,
  "staffQrCodeExpireTime":10
}
```

1.5.5. HTTP响应示例

响应 200

```
{ "code":0, "msg":"成功"}
```

1.6. 查看入库标准

1.6.1. 基本信息

Path： /v1/api/sys/storageCriteria/info

Method： POST

接口描述： 查看人脸入库标准配置，自定义配置中可查看模糊度、垂直角度、旋转角度等配置；

1.6.2. 请求参数

Headers

参数名称	参数值	是否必须	示例	备注
Content-Type	application/json	是		

1.6.3. 返回数据

名称	类型	是否必须	备注
level	integer	必须	人脸入库登记等级，详情请查看数据字典“人脸入库登记等级”
customConfig	object	非必须	
└─ blur	number	非必须	模糊度（自定义时必有），模糊度为0-1之间的数
└─ pitch	number	非必须	垂直角度（自定义时必有），垂直角度为1-100之间的数
└─ roll	number	非必须	旋转角度（自定义时必有），旋转角度为1-100之间的数
└─ yaw	number	非必须	水平角度（自定义时必有），水平角度为1-100之间的数
└─ brightnessMin	number	非必须	最小平均亮度（自定义时必有），最小平均亮度为1-1000之间的整数
└─ size	number	非必须	最小人脸（自定义时必有），最小人脸为1-500之间的整数
└─ sdtDeviation	number	非必须	亮度标准差（自定义时必有），亮度标准差为1-1000之间的整数
└─ brightnessMax	number	非必须	最大平均亮度（自定义时必有），最大平均亮度为1-1000之间的整数

1.6.4. HTTP请求示例

请求 path

```
/v1/api/sys/storageCriteria/info
```

请求 body

```
{}
```

1.6.5. HTTP响应示例

响应 200

```
{
  "code": 0,
  "data": {
    "customConfig": {
      "blur": 0.9,
      "brightnessMax": 200.0,
      "brightnessMin": 10.0,
      "pitch": 100.0,
      "roll": 100.0,
      "sdtDeviation": 200.0,
      "size": 30.0,
      "yaw": 100.0
    },
    "level": 4
  },
  "msg": "成功"
}
```

1.7. 保存入库标准

1.7.1. 基本信息

Path： /v1/api/sys/storageCriteria/save

Method： POST

接口描述： 保存人脸入库标准配置，可配置高、低、自定义配置

1.7.2. 请求参数

Headers

参数名称	参数值	是否必须	示例	备注
Content-Type	application/json	是		

Body

名称	类型	是否必须	备注
level	integer	必须	人脸入库登记等级，详情请查看数据字典“人脸入库登记等级”
customConfig	object	非必须	
└─ blur	number	非必须	模糊度（自定义时必有），模糊度为0-1之间的数
└─ pitch	number	非必须	垂直角度（自定义时必有），垂直角度为1-100之间的数
└─ roll	number	非必须	旋转角度（自定义时必有），旋转角度为1-100之间的数
└─ yaw	number	非必须	水平角度（自定义时必有），水平角度为1-100之间的数
└─ brightnessMin	number	非必须	最小平均亮度（自定义时必有），最小平均亮度为1-1000之间的整数
└─ size	number	非必须	最小人脸（自定义时必有），最小人脸为1-500之间的整数
└─ sdtDeviation	number	非必须	亮度标准差（自定义时必有），亮度标准差为1-1000之间的整数
└─ brightnessMax	number	非必须	最大平均亮度（自定义时必有），最大平均亮度为1-1000之间的整数

1.7.3. 返回数据

名称	类型	是否必须	备注
level	integer	必须	人脸入库登记等级，详情请查看数据字典“人脸入库登记等级”
customConfig	object	非必须	
└─ blur	number	非必须	模糊度（自定义时必有），模糊度为0-1之间的数
└─ pitch	number	非必须	垂直角度（自定义时必有），垂直角度为1-100之间的数
└─ roll	number	非必须	旋转角度（自定义时必有），旋转角度为1-100之间的数
└─ yaw	number	非必须	水平角度（自定义时必有），水平角度为1-100之间的数
└─ brightnessMin	number	非必须	最小平均亮度（自定义时必有），最小平均亮度为1-1000之间的整数
└─ size	number	非必须	最小人脸（自定义时必有），最小人脸为1-500之间的整数
└─ sdtDeviation	number	非必须	亮度标准差（自定义时必有），亮度标准差为1-1000之间的整数
└─ brightnessMax	number	非必须	最大平均亮度（自定义时必有），最大平均亮度为1-1000之间的整数

1.7.4. HTTP请求示例

请求 path

```
/v1/api/sys/storageCriteria/save
```

请求 body

```
{
  "level": 4,
  "customConfig": {
    "blur": 1,
    "pitch": 90,
    "roll": 90,
    "yaw": 90,
    "brightnessMin": 10,
    "size": 20,
    "sdtDeviation": 500,
    "brightnessMax": 500
  }
}
```

1.7.5. HTTP响应示例

响应 200

```
{
  "code": 0,
  "data": {
    "customConfig": {
      "blur": 1.0,
      "brightnessMax": 500.0,
      "brightnessMin": 10.0,
      "pitch": 90.0,
      "roll": 90.0,
      "sdtDeviation": 500.0,
      "size": 20.0,
      "yaw": 90.0
    },
    "level": 4
  },
  "msg": "成功"
}
```

1.8. 重置入库标准

1.8.1. 基本信息

Path： /v1/api/sys/storageCriteria/setDefault

Method： POST

接口描述： 重置人脸入库标准配置至默认配置（高）；

1.8.2. 请求参数

Headers

参数名称	参数值	是否必须	示例	备注
Content-Type	application/json	是		

1.8.3. 返回数据

名称	类型	是否必须	备注
level	integer	必须	等级：3-高

1.8.4. HTTP请求示例

请求 path

```
/v1/api/sys/storageCriteria/setDefault
```

请求 body

```
{}
```

1.8.5. HTTP响应示例

响应 200

```
{
  "code": 0,
  "data": {
    "level": 3
  },
  "msg": "成功"
}
```

1.9. 测温、口罩、健康码开关查询

1.9.1. 基本信息

Path: /v1/api/sys/config/temperatureConfig

Method: POST

接口描述: 查询测温、口罩、健康码功能开关状态；

1.9.2. 请求参数

Headers

参数名称	参数值	是否必须	示例	备注
Content-Type	application/json	是		

1.9.3. 返回数据

名称	类型	是否必须	备注
temperatureSwitch	integer	必须	测温模式开关, 0: 关闭, 1: 打开
maskAlertSwitch	integer	必须	口罩提示开关, 0: 关闭, 1: 打开
healthKitSwitch	integer	必须	健康码提示开关, 0: 关闭, 1: 打开

1.9.4. HTTP请求示例

请求 path

```
/v1/api/sys/config/temperatureConfig
```

请求 body

```
{}
```

1.9.5. HTTP响应示例

响应 200

```
{
  "code": 0,
  "data": {
    "maskAlertSwitch": 1,
    "temperatureSwitch": 1,
    "healthKitSwitch": 1
  },
  "msg": "成功"
}
```

- 1. 事件记录回调
 - 1.1. 事件记录回调说明
 - 1.2. 事件记录回调数据结构
 - 1.3. 回调数据示例
 - 1.3.1. 通行记录回调数据示例

1. 事件记录回调

1.1. 事件记录回调说明

在【系统管理】-【功能配置】-【事件配置】页面，填入事件订阅服务器地址后，配置事件规则-联动反馈时，选择联动事件订阅，产生的事件记录就会推送到该服务器地址。

1.2. 事件记录回调数据结构

字段	字段类型	是否必须	默认值	备注
id	string	非必须		事件记录id
trackId	string	非必须		事件记录的trackId
uuid	string	非必须		事件记录的uuid
captureImg	string	非必须		抓拍图
groupImg	string	非必须		底库图
fullImg	string	非必须		全景图
eventName	string	非必须		事件类型名
eventTypeid	integer	非必须		事件类型id
deviceId	integer	非必须		设备id
deviceName	string	非必须		设备名
deviceUuid	string	非必须		设备uuid
zoneId	integer	非必须		区域ID
zoneUuid	String	非必须		区域uuid

字段	字段类型	是否必须	默认值	备注
zoneName	String	非必须		区域名
timestamp	integer	非必须		事件时间
eventLevelId	integer	非必须		事件等级ID, 详情请查看数据字典“事件等级”
eventLevelColor	string	非必须		事件等级颜色
eventLevelName	string	非必须		事件等级名
status	integer	非必须		事件记录状态, 详情请查看数据字典“事件状态”
nvrId	integer	非必须		nvr设备id
nvrChannelNo	integer	非必须		nvr channel 号
gmtCreate	integer	非必须		创建时间
gmtModified	integer	非必须		修改时间
ext	string	非必须		业务扩展数据

ext 通行记录

字段	字段类型	是否必须	默认值	备注
personName	String	非必须		用户名

1.3. 回调数据示例

1.3.1. 通行记录回调数据示例

响应 200

```
{
  "captureImg": "http://10.122.94.244/pub/_ZzEwMF9mb3JldmVyQnVja2V0_132e997",
  "deviceId": 351,
  "deviceName": "C3S123-视频流-B3R1012-126-通",
  "eventLevelColor": "#D9001B",
  "eventLevelId": 1,
  "eventLevelName": "高",
  "eventType": 501,
  "eventName": "重点人员出入事件",
  "ext": {
    "personName": "杨亮"
  },
  "gmtCreate": 1625123076976,
  "gmtModified": 1625123076976,
  "groupImg": "http://10.122.94.244/pub/_ZzEwMF9mb3JldmVyQnVja2V0_7568ee9cf",
  "id": "8-LiYHoBG4rJ_r3I01B0",
  "status": 0,
  "timestamp": 1625123075478,
  "uuid": "64bfa27726e44db181c22dc4e7c97ab4",
  "zoneId": 1,
  "zoneName": "一级区域"
}
```

- 1.1. 通行记录回调数据结构
 - 1.1.1. 回调数据示例

1.1. 通行记录回调数据结构

字段	字段类型	是否必须	备注
recordType	integer	非必须	记录类型。1 - 通行;
personId	long	非必须	人员id
personUuid	string	非必须	人员UUID
personName	string	非必须	人员姓名
personType	integer	非必须	人员类型，详情请查看数据字典“人员类型”
personCode	string	非必须	员工编码
personPhone	string	非必须	员工手机
personIdentifyNum	string	非必须	员工身份证
personEmail	string	非必须	员工邮箱
personUniquelyIdentify	string	非必须	员工唯一标识
personExt	string	非必须	员工备注
visitType	integer	非必须	访客类型，详情请查看数据字典“访客类型”
visitPeronUuid	string	非必须	访客人员UUID

字段	字段类型	是否必须	备注
visitPersonType	integer	非必须	访问人员的类型，详情请查看数据字典“人员类型”
visitPersonCode	string	非必须	访问人员的员工编码
visitPersonName	string	非必须	访问人员的姓名
visitReason	string	非必须	来访原因
visitFirm	string	非必须	访客单位
visitNumPlate	string	非必须	访客车牌
orgId	long	非必须	组织id
orgUuid	string	非必须	组织uuid
orgType	integer	非必须	组织类型。1 - 非顶层组织; 2 - 区域顶层组织
orgFullName	string	非必须	组织名称
orgNickName	string	非必须	组织别名
orgParentId	long	非必须	父组织id
zoneId	long	非必须	所属区域id

字段	字段类型	是否必须	备注
zoneUuid	string	非必须	所属区域uuid
zoneName	string	非必须	所属区域名称
deviceId	long	非必须	抓拍设备的id
deviceUuid	string	非必须	抓拍设备的uuid
deviceName	string	非必须	抓拍设备的名称
deviceLocation	string	非必须	抓拍设备的位置
deviceType	integer	非必须	抓拍设备的类型：1 - 智能摄像头 2 - 智能面板机 4 - 网络摄像机
deviceModel	integer	非必须	抓拍设备的型号：设备型号（详细型号描述请查看数据字典）与"设备的类型"相对应
deviceStatus	integer	非必须	抓拍设备在抓拍时的状态。3 - 在线; 4 - 离线; (还有其他4种状态，客户无需关注 0 - 未激活; 1 - 在线未绑定; 2 - 离线未绑定; 5 - 解绑中)
recognitionType	integer	非必须	识别人员类型。1 - 员工; 2 - 访客; 3 - 重点人员（黑名单）; 4 - 无权限人员; 5 - 未识别; 6 - VIP访客

字段	字段类型	是否必须	备注
recognitionTypeDesc	string	非必须	识别人员类型文字描述
verificationMode	integer	非必须	验证类型。0 - 人脸; 1 - 人脸或刷卡; 2 - 人脸及刷卡; 3 - 人脸及密码; 4 - 远程开门
verificationModeDesc	string	非必须	验证类型文字描述
vistorRecordType	integer	非必须	验证类型, 详情请查看数据字典“通行认证方式”
passType	integer	非必须	通行类型。0 - 未通行; 1 - 通行; 2 - 复合认证未通过
passTypeDesc	string	非必须	通行类型文字描述
livenessType	integer	非必须	活体结果。0 - 非活体; 1 - 活体 ;2 - 未知
recognitionScore	double	非必须	识别分数
livenessScore	double	非必须	活体分数
passTime	OffsetDateTime	非必须	抓拍时间。ISO 8601 的精确到秒的格式化时间。形如2020-04-15T17:16:20+08:00
playbackSupport	boolean	非必须	是否支持录像回放
nvrUuid	string	非必须	录像回放NVR id

字段	字段类型	是否必须	备注
channelNo	integer	非必须	录像回放channel number
id	string	非必须	本条识别记录的id
personImageUrl	string	非必须	人员底库图片url
snapshotUrl	string	非必须	抓拍图url
frameUrl	string	非必须	全景图url
passTimestamp	long	非必须	抓拍时间。unix时间戳，毫秒
passTimeFormat	string	非必须	格式化为string类型的抓拍时间
temperature	double	非必须	人体温度
highThreshold	double	非必须	高温阈值
temperatureType	integer	非必须	人体高温状态，0：未检测，1：正常，2：高温
maskType	integer	非必须	口罩状态，0：未知，1：戴口罩，2：未戴口罩
temperatureTypeDesc	string	非必须	人体高温状态文字描述
maskTypeDesc	string	非必须	口罩状态文字描述 /

字段	字段类型	是否必须	备注
autograph	string	非必须	个性签名

1.1.1. 回调数据示例

响应 200

```

{
  "birthdayString": "生日快乐",
  "channelNo": 1,
  "deviceId": 1,
  "deviceLocation": "未知",
  "deviceName": "设备",
  "deviceStatus": 3,
  "deviceType": 1,
  "deviceUuid": "c27689ef42d446c686ee58b14249ce7f",

  "highThreshold": 38.5,
  "id": "313123",
  "livenessScore": "88.0",
  "livenessType": 1,
  "maskType": 1,
  "maskTypeDesc": "戴口罩",
  "orgFullName": "组织",
  "orgId": 1,
  "orgNickName": "",
  "orgParentId": 1,
  "orgType": 1,
  "orgUuid": "7348499c84ba4cb8a0e16cba33080be4",
  "passTime": "2021-05-24T14:13:10.298+08:00",
  "passTimeFormat": "2021-05-24 14:08:39",
  "passTimestamp": 1621836519111,
  "passType": 1,
  "passTypeDesc": "未通行",
  "personCode": "123123",
  "personExt": "新员工",
  "personId": 1,
  "personIdentifyNum": "1423221990020222112",
  "personImageUrl": "http://10.122.101.181/pub/_ZzEwMF9mb3JldmVyQnVja2V0_05ea",
  "personName": "张三",
  "personPhone": "13513502132",
  "personType": 1,
  "personUniqueIdentify": "21312312",
  "personUuid": "1fe6039e266049acb1221388980b7a14",
  "recognitionScore": "89.0",
  "recognitionType": 1,
  "recordType": 1,
  "snapshotUrl": "http://10.122.101.181/pub/_ZzEwMF9mb3JldmVyQnVja2V0_05eafd1",
  "temperature": 36.5,
  "temperatureType": 0,
  "temperatureTypeDesc": "未检测",
  "verificationMode": 1,
  "verificationModeDesc": "人脸",
  "visitFirm": "交通部",
  "visitNumPlate": "京p88888",
  "visitPeronUuid": "259c090d4d5945e39b1eee835df5b768",
  "visitPersonId": 2,
  "visitPersonName": "李四",
  "visitPersonType": 1,
  "visitReason": "面试",
  "visitType": 1,
  "zoneId": 1,
  "zoneName": "一级区域",
  "zoneUuid": "91790090b6a94dd4b2a9fa46f6aeeed33",
  "autograph": "每天都要精彩"
}

```

- 1.1. 考勤记录回调数据结构
 - 1.1.1. 回调数据示例

1.1. 考勤记录回调数据结构

名称	类型	是否必须	备注
uuid	string	非必须	考勤记录uuid
personName	string	非必须	员工姓名
personCode	string	非必须	员工编号
attendanceDate	string	非必须	考勤日期，时间戳
attTimeResult	object []	非必须	考勤时段结果：打卡时间&考勤结果
└─ order	integer	非必须	考勤时段编号，详情请查看数据字典“考勤时段编号”
└─ inPunch	string	非必须	上班打卡时间eg:09:00:00，需要精确到秒
└─ outPunch	string	非必须	下班打卡时间eg:11:00:00，需要精确到秒
└─ result	integer	非必须	考勤结果，详情请查看数据字典“考勤结果”
status	integer	非必须	考勤状态，注：：0-暂无结果（'--'表示），1-正常，2-旷工，3-出勤调整，4-异常

1.1.1. 回调数据示例

响应 200

```
{
  "attTimeResult": [{
    "adjust": -1,
    "inPunch": "09:00:00",
    "order": 1,
    "outPunch": "10:00:00",
    "period": 2,
    "result": 4
  }],
  "attendanceDate": 1624896000000,
  "personCode": "5677978",
  "personName": "彭斯亮",
  "personUuid": "2a12603fc2db4ebebe6dd8d6004e0c93",
  "status": 4,
  "uuid": "26862b1abeb64c4d869add1d225e5ebe"
}
```

- 1. 订阅接口
 - 1.1. 业务数据记录订阅
 - 1.1.1. 基本信息
 - 1.1.2. 请求参数
 - 1.1.3. 返回数据
 - 1.1.4. HTTP请求示例
 - 1.1.5. HTTP响应示例
 - 1.2. 查询业务数据记录订阅
 - 1.2.1. 基本信息
 - 1.2.2. 请求参数
 - 1.2.3. 返回数据
 - 1.2.4. HTTP请求示例
 - 1.2.5. HTTP响应示例
 - 1.3. 解除业务数据记录订阅
 - 1.3.1. 基本信息
 - 1.3.2. 请求参数
 - 1.3.3. 返回数据
 - 1.3.4. HTTP请求示例
 - 1.3.5. HTTP响应示例

1. 订阅接口

1.1. 业务数据记录订阅

1.1.1. 基本信息

Path： /v1/api/pubsub/busi/record/add

Method： POST

接口描述： 第三方订阅不同类型的业务数据

1.1.2. 请求参数

Headers

参数名称	参数值	是否必须	示例	备注
Content-Type	application/json	是		

Body

名称	类型	是否必须	备注
appKey	string	必须	用户appKey，如果不传入，就取当前请求携带的cappKey
pushUrl	string	必须	用户接受数据的url，产生的业务数据会推送到这个地址
busiTypes	integer []	必须	要订阅的业务类型:5-通行记录
└─		非必须	

1.1.3. 返回数据

```
OK
```

1.1.4. HTTP请求示例

请求path

```
/v1/api/pubsub/busi/record/add
```

请求body

```
{
  "appKey": "sdfajk3242324fa!dj7",
  "pushUrl": "http://10.122.101.34:9999/event/record/push",
  "busiTypes": [
    1,
    2,
    3,
    5,
    12
  ]
}
```

1.1.5. HTTP响应示例

响应 200

```
{
  "code": 0,
  "msg": "成功"
}
```

1.2. 查询业务数据记录订阅

1.2.1. 基本信息

Path: /v1/api/pubsub/busi/record/query

Method: POST

接口描述: 查询订阅的信息

1.2.2. 请求参数

Headers

参数名称	参数值	是否必须	示例	备注
Content-Type	application/json	是		

Body

名称	类型	是否必须	备注
appKey	string	非必须	用户appKey，优先从请求头取当前请求携带的cappKey，如果请求头没有，则必传

1.2.3. 返回数据

名称	类型	是否必须	备注
appKey	string	非必须	用户appKey
pushUrl	string	非必须	用户接受数据的url，产生的业务数据会推送到这个地址
busiTypes	integer []	非必须	业务订阅类型，详情请查看数据字典“订阅的业务类型”

1.2.4. HTTP请求示例

请求path

```
/v1/api/pubsub/busi/record/query
```

请求body

```
{
  "appKey": "sdfajk3242324fa!dj7"
}
```

1.2.5. HTTP响应示例

响应 200

```
{
  "code": 0,
  "data": {
    "appKey": "appkey1",
    "busiTypes": [
      5
    ],
    "pushUrl": "http://10.122.101.34:9999/event/record/push"
  },
  "msg": "成功"
}
```

1.3. 解除业务数据记录订阅

1.3.1. 基本信息

Path: /v1/api/pubsub/busi/record/unbind

Method: POST

接口描述: 解除订阅

1.3.2. 请求参数

Headers

参数名称	参数值	是否必须	示例	备注
Content-Type	application/json	是		

Body

名称	类型	是否必须	备注
appKey	string	非必须	用户appKey，优先从请求头取当前请求携带的cappKey，如果请求头没有，则必传

1.3.3. 返回数据

```
OK
```

1.3.4. HTTP请求示例

请求path

```
/v1/api/pubsub/busi/record/unbind
```

请求body

```
{  
  "appKey": "sdfajk3242324fa!d"  
}
```

1.3.5. HTTP响应示例

响应 200

```
{  
  "code": 0,  
  "msg": "成功"  
}
```

- [1. 业务推送数据结构](#)

1. 业务推送数据结构

字段	字段类型	是否必须	备注
busiType	string	非必须	业务类型，详情请查看数据字典“订阅的业务类型”
data	string	非必须	对应业务类型的数据（格式json字符串），不同的业务类型产生的数据各不相同，下面会一一说明

- [1.1. busiType=5 通行识别记录推送](#)
 - [1.1.1. 回调数据示例](#)

1.1. busiType=5 通行识别记录推送

字段	字段类型	是否必须	默认值	备注
personId	long	非必须		人员id
personUuid	string	非必须		人员UUID
personName	string	非必须		人员姓名
personType	integer	非必须		人员类型, 详情请查看数据字典“人员类型”
personCode	string	非必须		员工编码
personPhone	string	非必须		员工手机
personIdentifyNum	string	非必须		员工身份证
personEmail	string	非必须		员工邮箱
personUniqueIdentify	string	非必须		员工唯一标识
personExt	string	非必须		员工备注
visitType	integer	非必须		访客类型, 详情请查看数据字典“访客类型”
visitPeronUuid	string	非必须		访客人员UUID
visitPersonType	integer	非必须		访问人员的类型, 详情请查看数据字典“人员类型”

字段	字段类型	是否必须	默认值	备注
visitPersonCode	string	非必须		访问人员的员工编号
visitPersonName	string	非必须		访问人员的姓名
visitReason	string	非必须		来访原因
visitFirm	string	非必须		访客单位
visitNumPlate	string	非必须		访客车牌
orgId	long	非必须		组织id
orgUuid	string	非必须		组织uuid
orgType	integer	非必须		组织类型。1 - 非层组织; 2 - 区域层组织
orgFullName	string	非必须		组织名称
orgNickName	string	非必须		组织别名
orgParentId	long	非必须		父组织id
zoneId	long	非必须		所属区域id
zoneUuid	string	非必须		所属区域uuid

字段	字段类型	是否必须	默认值	备注
zoneName	string	非必须		所属区域名称
deviceId	long	非必须		抓拍设备的id
deviceUuid	string	非必须		抓拍设备的uuid
deviceName	string	非必须		抓拍设备的名称
deviceLocation	string	非必须		抓拍设备的位置
deviceType	integer	非必须		抓拍设备的类型 - 智能摄像头 2 - 智能面板机 4 - 网络摄像机
deviceModel	integer	非必须		抓拍设备的型号 设备型号（详细型号描述请查看数据字典）与"设备的型号"相对应
deviceStatus	integer	非必须		抓拍设备在抓拍时的状态。3 - 在线; 4 - 离线; (还有其他4种状态, 客户不关注 0 - 未激活; 1 - 在线未绑定; 2 - 离线未绑定; 5 - 解绑中)
recognitionType	integer	非必须		识别人员类型。1 - 员工; 2 - 访客; 3 - 重点人员 (黑名单); 4 - 无权限人员; 5 - 未识别; 6 - VIP访客
recognitionTypeDesc	string	非必须		识别人员类型文本描述

字段	字段类型	是否必须	默认值	备注
verificationMode	integer	非必须		验证类型，详情查看数据字典“通认证方式”
verificationModeDesc	string	非必须		验证类型文字描述
vistorRecordType	integer	非必须		验证类型，详情查看数据字典“通认证方式”
passType	integer	非必须		通行类型。0 - 未通行; 1 - 通行; 2 - 复合认证未通过
passTypeDesc	string	非必须		通行类型文字描述
livenessType	integer	非必须		活体结果。0 - 非活体; 1 - 活体; 2 - 未知
recognitionScore	double	非必须		识别分数
livenessScore	double	非必须		活体分数
passTime	OffsetDateTime	非必须		抓拍时间。ISO 8601的精确到秒格式化时间。形如2020-04-15T17:16:20+08
playbackSupport	boolean	非必须		是否支持录像回放
nvrUuid	string	非必须		录像回放NVR id
channelNo	integer	非必须		录像回放channel number

字段	字段类型	是否必须	默认值	备注
id	string	非必须		本条识别记录的id
personImageUrl	string	非必须		人员底库图片url
snapshotUrl	string	非必须		抓拍图url
frameUrl	string	非必须		全景图url
passTimestamp	long	非必须		抓拍时间。unix时间戳，毫秒
passTimeFormat	string	非必须		格式化为string类的抓拍时间
temperature	double	非必须		人体温度
highThreshold	double	非必须		高温阈值
temperatureType	integer	非必须		人体高温状态，0：未检测，1：正常，2：高温
maskType	integer	非必须		口罩状态，0：未知，1：戴口罩，2：未戴口罩
temperatureTypeDesc	string	非必须	人体高温状态文字描述	

字段	字段类型	是否必须	默认值	备注
maskTypeDesc	string	非必须	口罩状态文字描述	
autograph	string	非必须	个性签名	

1.1.1. 回调数据示例

响应 200

```

{
  "passTypeDesc": "通行",
  "temperatureType": 0,
  "recognitionScore": 86.82751,
  "passTime": "2021-08-13T18:11:27.177+08:00",
  "zoneUuid": "7ca8da7d8f424fabb891c99e20251404",
  "deviceId": 661,
  "deviceName": "c3v-32j",
  "orgUuid": "7ca8da7d8f424fabb891c99e20251404",
  "maskTypeDesc": "未检测",
  "orgType": 2,
  "passTimestamp": 1628849487177,
  "passType": 1,
  "personIdentifyNum": "017589",
  "verificationMode": 0,
  "zoneId": 1,
  "id": "30eb7cb982694641b64c334087849aec",
  "visitPersonName": "备用员工1",
  "deviceType": 4,
  "deviceUuid": "c95557d72d394ece943f90e0d975fdf2",
  "deviceSetUuid": "c8984a39228848f1960656f23faeae16",
  "maskType": 0,
  "orgParentId": 0,
  "orgFullName": "一级区域",
  "highThreshold": 37.2,
  "visitType": 2,
  "vistorType": 0,
  "isAttendance": false,
  "recognitionType": 6,
  "visitPeronUuid": "0b4214b64bdd499b8ae2feca5f84d0af",
  "personId": 816,
  "highTemperatureOff": false,
  "deviceSetId": 475,
  "deviceLocation": "openapi",
  "livenessScore": 81.48996,
  "visitPersonCode": "52294059",
  "personUuid": "24b73fcd65f34147b0473ff00200efdd",
  "passTimeFormat": "2021-08-13 18:11:27",
  "orgId": 1,
  "deviceStatus": 3,
  "visitReason": "vip拜访",
  "personPhone": "15168346187",
  "personExt": "vip访客备注",
  "temperature": 0.0,
  "zoneName": "一级区域",
  "personType": 2,
  "recordSource": 1,
  "visitPersonId": 814,
  "visitPersonType": 1,
  "temperatureTypeDesc": "未检测",
  "verificationModeDesc": "人脸",
  "personImageUrl": "http://10.171.4.194/pub/_Y2FjdHVzMTAwX2ZvcvV2ZXJCdWNRZXQ=_9274d8a96bc343108059e",
  "recordType": 1,
  "personImageUri": "_Y2FjdHVzMTAwX2ZvcvV2ZXJCdWNRZXQ=_9274d8a96bc343108059e",
  "personName": "备用vip访客1",
  "livenessType": 2,
  "personEmail": "anQ9gpsYQG@126.com",
  "playbackSupport": false,
  "snapshotUrl": "http://10.171.4.194/pub/_Y2FjdHVzMTAwXzNt_b659c59340894abca",
  "deviceModel": 21,
  "snapshotUri": "_Y2FjdHVzMTAwXzNt_b659c59340894abcaaa91a1fbeb8a6d2",
  "recognitionTypeDesc": "VIP访客",
  "autograph": "每天都要精彩"
}

```


- 1. 通用模块
 - 1.1. 上传文件（可用于人脸比对）
 - 1.1.1. 基本信息
 - 1.1.2. 请求参数
 - 1.1.3. 返回数据
 - 1.1.4. HTTP请求示例
 - 1.1.5. HTTP响应示例

1. 通用模块

1.1. 上传文件（可用于人脸比对）

1.1.1. 基本信息

Path： /v1/api/misc/file/upload

Method： POST

接口描述： 可以进行文件上传

1.1.2. 请求参数

Headers

参数名称	参数值	是否必须	示例	备注
Content-Type	multipart/form-data	是		

Body

参数名称	参数类型	是否必须	备注
file	file	是	上传文件,此接口适用于传输非人脸以外的文件类型
type	integer	是	1:导入文件，指各类批量导入，临时上传的文件,2:人脸（或其他）图片，大小限制为10M，可用于人脸比对场景（不做质量判断）

1.1.3. 返回数据

名称	类型	是否必须	备注
uri	string	非必须	上传返回的uri

1.1.4. HTTP请求示例

请求 path

```
/v1/api/misc/file/upload
```

请求 body

```
upload file stream (OMITTED)
```

1.1.5. HTTP响应示例

响应 200

```
{
  "code": 0,
  "data": {
    "uri": "_ZzEwMF9mb3JldmVyQnVja2V0_9a00bafda96b4829a9ff6cd470b1c675"
  },
  "msg": "成功"
}
```


- 1. 运维管理
 - 1.1. 查询设备信息
 - 1.1.1. 基本信息
 - 1.1.2. 请求参数
 - 1.1.3. 返回数据
 - 1.1.4. HTTP请求示例
 - 1.1.5. HTTP响应示例

1. 运维管理

1.1. 查询设备信息

1.1.1. 基本信息

Path： /v1/api/device/info

Method： GET

接口描述： 查询设备信息，设备型号、软件版本。

1.1.2. 请求参数

无

1.1.3. 返回数据

名称	类型	是否必须	备注
name	string	必须	设备名称
type	string	必须	设备型号
version	string	必须	系统版本

1.1.4. HTTP请求示例

请求 path

/v1/api/device/info

请求参数

无

1.1.5. HTTP响应示例

响应 200

```
{
  "code": 0,
  "msg": "成功",
  "time_used": 0.008513603,
  "data": {
    "name": "AIoT视图综合应用一体机",
    "type": "MegCube-HAL3004-4S2",
    "version": "V1.0.2.0"
  }
}
```

- 1. 人脸比对
 - 1.1. 1:1查询
 - 1.1.1. 基本信息
 - 1.1.2. 请求参数
 - 1.1.3. 返回数据
 - 1.2. 1: N列表查询
 - 1.2.1. 基本信息
 - 1.2.2. 请求参数
 - 1.2.3. 返回数据
 - 1.2.4. HTTP请求示例
 - 1.2.5. HTTP响应示例

1. 人脸比对

1.1. 1:1查询

1.1.1. 基本信息

Path: /v1/api/photo/searchOneToOnePhoto

Method: POST

接口描述: 上传两张人脸图片进行人脸相似度比分

1.1.2. 请求参数

Headers

参数名称	参数值	是否必须	示例	备注
Content-Type	application/json	是		

Body

名称	类型	是否必须	备注
image1	string	必须	1、支持图片1 Base64编码 2、支持efs地址，使用该接口上传图片获取efs地址：/v1/api/person/uploadImage 3、必须同时使用Base64或者esf类型对比，不可以混合使用（建议使用efs方式）
image2	string	必须	1、支持图片2 Base64编码 2、支持efs地址，使用该接口上传图片获取efs地址：/v1/api/person/uploadImage 3、必须同时使用Base64或者esf类型对比，不可以混合使用（建议使用efs方式）
imageType	integer	必须	类型：1 efs地址，2 Base64编码 默认1

1.1.3. 返回数据

名称	类型	是否必须	备注
score	number	非必须	比较分数

HTTP请求示例

1、base64请求实例

```
{
  "image1": "图片base64处理字符串",
  "image2": "图片base64处理字符串",
  "imageType": 2
}
```

2、EFS图片请求实例

URI获取 调用下面【上传人员图片，不卡质量】接口 /v1/api/person/uploadImage

```
{
  "image1": "_ZzEwMF82bQ==_8125493cca594b55999767dec7a12640",
  "image2": "_ZzEwMF82bQ==_8125493cca594b55999767dec7a12640",
  "imageType": 1
}
```

HTTP响应示例

响应200

```
{
  "code": 0,
  "data": {
    "score": 99.98237
  },
  "msg": "成功"
}
```

1.2. 1： N列表查询

1.2.1. 基本信息

Path： /v1/api/photo/search

Method： POST

接口描述： 上传图片进行比分获取底库最相似人

1.2.2. 请求参数

Headers

名称	参数值	是否必须	示例	备注
Content-Type	application/json	是		

Body

名称	类型	是否必须	备注
uri	string	必须	图片寻址uri，通过【上传人员图片接口】获取uri
groupTypes	integer []	必须	底库来源 1-员工库；2-访客库；3-重点人员库
└─		非必须	
threshold	integer	必须	阈值大小, 数值为正整数，阈值区间 (1, 99) 包括1和99
pageNum	integer	非必须	当前页, 默认值1
pageSize	integer	非必须	每页条数, 默认值0

1.2.3. 返回数据

名称	类型	是否必须	备注
pageNum	integer	非必须	当前页码
pageSize	integer	非必须	分页大小
total	integer	非必须	总数
list	object []	非必须	
└─url	string	非必须	底库图
└─personUuid	string	非必须	人员uuid
└─profileId	string	非必须	档案id
└─name	string	非必须	姓名
└─groupType	integer	非必须	人员身份类型， 1-员工;2-访客;3-黑名单;4-陌生人
└─groupTypeDesc	string	非必须	人员身份
└─score	number	非必须	比对分数
└─count	integer	非必须	返回详情数量

1.2.4. HTTP请求示例

请求 path

```
/v1/api/photo/search
```

请求 body

```
{
  "uri": "_ZzEwMF9mb3JldmVyQnVja2V0_1e381bd5932846519d0c605a99d65752",
  "groupTypes": [
    3
  ],
  "threshold": 70,
  "pageNum": 1,
  "pageSize": 10
}
```

1.2.5. HTTP响应示例

响应 200

```
{
  "code": 0,
  "data": {
    "list": [
      {
        "count": 8,
        "groupType": 3,
        "groupTypeDesc": "黑名单",
        "name": "备用黑名单1",
        "personUuid": "224eb049f12a47e58f518f5ae34b143d",
        "score": 99.98075,
        "url": "http://10.171.4.130/pub/_ZzEwMF9mb3JldmVyQnVja2V0_aed08"
      }
    ],
    "pageNum": 1,
    "pageSize": 10,
    "total": 1
  },
  "msg": "成功"
}
```