loonflow

发布 *1.0.0*

blackholll

2020 年 05 月 01 日

关于 loonflow

1 loonflow 是什么 **3**

2 前言 **5**

3 操作系统支持 **7**

4 如何获取代码 **9**

5 如何运行 **11**

5.1 开发环境 11

5.2 生产环境 12

5.3 生产环境 docker compose 方式部署 13

5.4 演示环境 docker compose 方式运行 14

6 版本升级 **17**

6.1 r0.1.x-r.2.x 17

6.2 r0.2.x-r.3.x 18

6.3 r0.3.x-r1.0.x 18

7 相关术语 **21**

8 基础架构 **23**

9 代码结构 **25**

10 登录管理后台 **27**

11 同步用户信息 **29**

12 工作流配置 **31**

12.1 自定义通知 31

12.2 自定义执行脚本 . . . 32
12.3 工作流 . . . 33
12.4 状态 . . . 35
12.5 流转 . . . 37

13 用户管理 **39**

14 角色管理 **41**

15 部门管理 **43**

16 调用权限 **45**

17 接口调用鉴权 **47**

18 接口调用逻辑 **49**

19 工作流相关接口 **51**
19.1 获取工作流列表 . . . 51
19.2 获取工作流初始状态 . . . 52
19.3 获取工作流状态详情 . . . 55
19.4 获取工作流状态列表 . . . 56

20 工单相关接口 **61**
20.1 获取工单列表 . . . 61
20.2 新建工单 . . . 63
20.3 获取工单详情 . . . 64
20.4 获取工单可以做的操作 . . . 68
20.5 接单 . . . 70
20.6 转交 . . . 70
20.7 加签 . . . 71
20.8 加签处理完成 . . . 72
20.9 处理工单 . . . 72
20.10 获取工单流转记录 . . . 73
20.11 工单处理步骤记录 . . . 79
20.12 修改工单状态 . . . 91
20.13 批量获取工单状态 . . . 92
20.14 修改工单字段的值 . . . 93
20.15 重试工单脚本/hook 任务 . . . 93
20.16 新增工单评论/注释 . . . 94
20.17 工单 hook 回调 . . . 94
20.18 工单当前的参与人详情 . . . 95
20.19 强制关闭工单 . . . 97
20.20 撤回工单 . . . 97

21 相关常量 **99**
21.1 状态类型 . . . 99
21.2 分配方式 . . . 99
21.3 处理人类型 . . . 100
21.4 流转类型 . . . 100
21.5 自定义字段类型 . . . 100
21.6 字段属性 . . . 101
21.7 工单权限类别 . . . 101

22 常见问题 **103**

23 欢迎捐助 **107**

24 发布说明 **109**
24.1 r1.0.4 . . . 109
24.2 r1.0.3 . . . 109
24.3 r1.0.2 . . . 109
24.4 r1.0.1 . . . 110
24.5 r1.0.0 . . . 110
24.6 r0.x.x . . . 111

r1.x 之前版本见 https://github.com/blackholll/loonflow/wiki

CHAPTER 1

loonflow 是什么

a workflow engine base django 基于 django 的工作流引擎系统, 通过 http 接口调用。可以作为企业内部统一的工作流引擎，提供诸如权限申请、资源申请、发布申请、请假、报销、it 服务等所有工作流场景的服务。如果有一定的开发能力建议只使用后端引擎功能, 前端根据场景定制开发可分散于各个内部后台管理系统 (如人事、运维、监控、cmdb 等等)

CHAPTER 2

前言

本人 2011 年开始接触工作流，2013 年开始开发工作流第一版本，至今经历了多个版本。目前着手开发一个开源版本，致力于提供企业统一工作流引擎方案欢迎加入 qq 群一起交流工作流相关技术: 558788490

CHAPTER 3

操作系统支持

建议使用 Centos,Redhat,Ubuntu 这类 linux 操作系统因为 celery4 以后不支持 windows, 所以状态脚本和通知脚本执行会无法使用。可以参考此文档兼容下: 参考文档

CHAPTER 4

如何获取代码

推荐使用最新的版本. 可以直接通过此链接下载 release 版本, 或者使用以下命令

```
# loonflow
git clone git@github.com:blackholll/loonflow.git
git checkout vx.x.x  #(具体的版本号, 如 v0.3.15) 拉取代码

# shutongflow: vue.js 版本前端 +django 后端的调用方 demo
https://github.com/youshutong2080/shutongFlow

# workflowdemo:  bootstrap 版本前后端 (前后端为分离的调用方 demo)
https://github.com/jimmy201602/workflowdemo
# 其他版本 demo: 欢迎有兴趣的同学提供 react 版本调用方 demo

# loonflow-helper: 一些脚本的 demo, 如通知脚本、执行脚本、用户同步脚本, 欢迎大家一起完善 (直接提pr)
https://github.com/blackholll/loonflow-helper
```

CHAPTER 5

如何运行

5.1 开发环境

- 将 settings/dev.py.simple 在 settings 目录下复制一份并重命名为 config.py
- 创建数据库并修改 settings/config.py 中相应配置 (数据库配置、redis 地址配置、日志路径配置等等)
- 创建 python 虚拟环境: python3.6.x(python3.6 最新稳定版)
- 安装依赖包: pip install -r requirements/dev.txt
- 启动 redis(用于生成唯一的工单流水号 +celery 异步任务 [执行脚本、状态 hook、通知 hook])
- 初始化数据库

```
python manage.py makemigrations
python manage.py migrate
# 如果只是本地测试，无需二次开发，也可以直接参考生产环境部署中直接导入初始 sql(初始 sql 中包含 admin
用户)
```

- 创建初始账户: python manage.py createsuperuser
- 启动开发环境: python manage.py runserver 如果需要启动在其他端口:python manage.py runserver 8888
- 启动 celery 任务: celery -A tasks worker -l info -Q loonflow (用于执行任务脚本、触发任务 hook、通知 hook。本地开发二次开发如果不需要这些功能时可以不启动)

5.2 生产环境

- 生产环境建议使用 nginx+uwsgi 的方式部署 (nginx 及 uwsgi 配置文件可参考 https://github.com/blackholll/loonflow-helper/tree/master/deploy)
- 将 settings/pro.py.simple 在 settings 目录下复制一份并重命名为 config.py
- 创建数据库并修改 settings/pro.py 中相应配置 (数据库配置、redis 地址配置、日志路径配置等等)
- 创建 python 虚拟环境: python3.6.x(python3.6 最新稳定版)
- 安装依赖包: pip install -r requirements/pro.txt
- 启动 redis(用于生成唯一的工单流水号 +celery 异步任务 [执行脚本、状态 hook、通知 hook]，centos7 下 redis service 配置文件可参考 https://github.com/blackholll/loonflow-helper/tree/master/deploy)
- 初始化数据库，导入初始化 sql，命令如下

```
mysql -uroot -p  loonflow_1_0 < loonflow_init.sql # 生产环境不建议使用 migrate. 用户名及数据库需要根据你的实际情况也即 config.py 中的配置做相应修改
```

- 初始 admin 账号密码为 admin/loonflow123 (用于登录管理后台，管理用户及配置工作流等)
- 启动 celery 任务: celery multi start -A tasks worker -l info -c 8 -Q loonflow –logfile=xxx.log –pidfile=xxx.pid # -c 参数为启动的 celery 进程数，注意 logfile 和 pidfile 前面是两个-，logfile 为日志文件路径, pidfile 为 pid 文件路径，可自行视情况调整
- 如需优雅停止 celery 服务:

```
celery multi stopwait -A tasks worker -l info -c 8 -Q loonflow --logfile=xxx.log --
↪pidfile=xxx.pid
```

- 如需优雅重启 celery 服务:

```
celery multi restart -A tasks worker -l info -c 8 -Q loonflow --logfile=xxx.log --
↪pidfile=xxx.pid
```

- 启动 uwsgi
- 启动 nginx

5.3 生产环境 docker compose 方式部署

为了方便的数据的持久化以及升级操作, 此方式不会启动数据库, 请事先准备好数据库。(注意创建库的使用使用 utf-8 字符集)。只部署 loonflow (包括 nginx、redis), 启动时将连接到你提供的数据库 (保证持久化数据)。

- 准备工作:

```
准备一台 linux 服务器
安装好 python3(请自行百度或者 google)
安装好 docker-compose(请自行百度或者 google)
配置容器镜像加速 (请自行百度或者 google)
```

- 准备好数据库，授予权限

```
# 进入 mysql 后创建数据库并授权
mysql> create database loonflow character set utf8;  # 注意要使用 utf8 字符集
mysql> grant all privileges on loonflow.* to loonflow@'%' identified by '123456';
```

- 启动方式

确保已经安装了 python3 后。cd 到 docker_compose_deploy/loonflow_only 目录后, 执行以下命令

```
# 修改 run.py 中数据库相关配置为你准备好的数据库信息
db_host = ''  # loonflow 使用的数据库的 ip
db_port = ''  # loonflow 使用的数据库的端口
db_name = ''  # loonflow 使用的数据库的名称
db_user = ''  # loonflow 使用的数据库的用户
db_password = ''  # loonflow 使用的数据库的用户密码

ddl_db_user = ''  # 可以执行 ddl(拥有修改表结构权限) 的用户
ddl_db_password = ''  # 可以执行 ddl(拥有修改表结构权限) 的用户的密码

# 安装并启动服务
python3 run.py install # 此命令后修改 dockerfile 中的数据库配置, 然后启动

# 启动服务
python3 run.py start  # 此命令直接启动服务, 请保证之前 install 过 (也就是 dockerfile 中数据库配置已被修改)

# 停止服务, 这种方式对于 celery task 任务非优雅停止, 可以使用 flower(celery 的监控系统), 将任务消费停止, 并且等待所有认为都结束后再执行
python3 run.py stop
```

5.4 演示环境 docker compose 方式运行

提供者: 逆寒刀 (children1987@qq.com)

5.4.1 安装前必读

- 本方式会同时安装 loonflow 及 shutongflow(looflow 的调用方 demo, https://github.com/youshutong2080/shutongFlow), shutongflow 功能不够完善，所以仅供大家开发调用方程序时参考
- 强烈建议基于一台新装的 CentOS 7 安装。因为其它场景可能会触发一些未被测到的问题
- 这只是一个为了方便快速展示代码的 demo，考虑到安全、性能等因素，请勿直接用于生产
- 至少需要 2G 内存，推荐 4G

5.4.2 安装前准备

- 关闭 firewalld

```
# 关闭防火墙
systemctl stop firewalld.service
# 检查防火墙状态
systemctl status firewalld
```

- 关闭 selinux

  建议永久关闭，而非临时关闭, 详见 https://blog.csdn.net/zhoushengtao12/article/details/95346903

- 保证以下端口未被使用

```
3306 (mysql)
6060 (loonflow)
6061 (shutongFlow_frontend)
6062 (shutongFlow_backend)
6379 (redis)
```

5.4.3 开始安装

```
cd /opt && yum install -y git && git clone -b v1.0.3 https://gitee.com/shihowcom/
↪loonflow_ro loonflow
# 在如下文件完成必要配置，重点是 ip
vi loonflow/docker_compose_deploy/loonflow_shutongflow/config.json
cd loonflow && python ./docker_compose_deploy/loonflow_shutongflow/setup_all.py
```

(下页继续)

(续上页)

```
# 各参数含义如下
{
"ip": "117.33.233.74",  # 你的服务器的地址
"is_need_setup_mysql": true,  # 是否希望程序同时安装 mysql,  如果需要则设置为 true。 建议事先准备好 mysql(此处设置为 false)
"mysql": {
  "root_password": "mySql12#4,.De",  # 如果事先准备好了 mysql, 此处设置你准备的 mysql 的密码, 否则保持不变
  "host": "127.0.0.1",  # 如果事先准备好了 mysql, 此处设置你准备的 mysql 的地址, 否则保持不变
  "port": "3306"   # 如果事先准备好了 mysql, 此处设置你准备的 mysql 的端口, 否则保持不变
}
```

}

5.4.4 访问

docker 容器们启动成功后，就可以通过以下方式访问了：

- loonflow 管理后台

```
访问地址: http://ip:6060/   ip 为你的 centos7 服务器的 ip 地址
账号/密码: admin/loonflow123
```

- shutongflow

```
访问地址: http://ip:6061/  ip 为你的 centos7 服务器的 ip 地址
账号/密码: admin/yxuqtr
```

CHAPTER 6

版本升级

6.1 r0.1.x-r.2.x

从 r0.1.x-r.2.x 升级。需要一些 DDL 操作

- workflow.models.Transition 新增字段 timer, 新增字段 attribute_type_id,condition_expression
- ticket.modles.TicketRecord 新增 script_run_last_result 字段, 新增 is_end 字段, 新增 is_rejected 字段, 新增 multi_all_person 字段
- 删除 ticket.modles.TicketStateLastMan
- workfow.models.State 新增 remember_last_man_enable 字段
- account.models.AppToken 新增字段 workflow_ids 字段, 用于给每个 app 授权可以访问的工作流资源 (对应工作及对应的工单，升级后需要修改此配置). 新增 ticket_sn_prefix 字段，用于支持配置工单流水号前缀
- workflow.models.Workflow 新增字段 limit_expression，用于新建工单权限的限制
- workflow.models.workflow 新增字段 notices，用于关联通知方式
- workflow.models 新增表 CustomNotice 用于支持自定义通知方式
- workflow.models.CustomField 新增 label 字段用于调用方自行扩展

6.2 r0.2.x-r.3.x

- 因为 v0.3 版本中 username 参数改成从 header 中获取，所以接口调用时需要将 username 通过 header 方式传递
- 为了脚本安全考虑, 当状态的参与人类型为脚本时，参与人需要设置为脚本记录的 id。迁移时需要将这些状态的参与人从脚本名称改成脚本记录的 id

6.3 r0.3.x-r1.0.x

0. 如果你对升级过程不熟悉，强烈建议你将生产环境数据 (数据库) 导入到测试环境，按照下面的操作演练过且没问题后再在生产环境操作
1. 准备好通知的 hook 服务端 (如果你当前有用到通知脚本，你需要改成 hook 方式, 如果没用到通知功能，可忽略此步)

准备好通知的 hook 的服务端 (可以在服务端提供短信、钉钉、企业微信、邮件等通知服务)，服务端给 loonflow 分配一个 token 用于生签名，服务器端以此 token 使用相同的算法生成签名用于校验 loonflow。校验通过后根据 hook 请求的数据来发送通知消息。

2. 下载 loonflow v1.0.x 版本到新的服务器或者新的目录
3. 将 0.3.x 版本的 media 目录下目录及文件全部复制到 v1.0.x 版本的 media 目录下
4. 创建新的 python3.6 虚拟环境，并安装好 requirement/pro.txt 中的依赖
5. 停止调用方服务
6. 停止 loonflow 0.3 版本服务 (包括 web 服务及 task 任务，task 任务可优雅停止，命令:xxxx)

```
celery multi stopwait -A tasks worker -l info -c 8 -Q loonflow --logfile=xxx.log --
↪pidfile=xxx.pid
```

7. 备份好 0.3.x 版本数据库 (为了在发现问题时快速回退)
8. 执行升级 sql

```
https://github.com/blackholll/loonflow-helper/tree/master/update/0.3.xto1.0.x/ddl.sql
https://github.com/blackholll/loonflow-helper/tree/master/update/0.3.xto1.0.x/dml.py ␣
↪ ## 将文件中的数据库配置修改为你的 0.3.x 版本使用的数据库
```

9. 将代码中 settings/pro.py 中复制并重命名为 settings/config.py, 将 config.py 中数据库就 redis 配置修改为当前使用的地址，临时修改 config.py 中的 DEBUG 参数 =True，进入新的虚拟环境尝试使用 python manage.py runserver 0.0.0.0:9999 启动 loonflow 1.0.x，观察是否有报错，排查错误
10. 访问 http://$serverip:9999, 在“工作流管理”-“通知管理”中新增好需要用到的通知

11. 访问 http://$serverip:9999，在“工作流管理”–“工作流配置”中逐个编辑需要发送通知的工作流，选择对应的通知，并设置标题模板和通知模板
12. 修改 config.py 中的 DEBUG 参数 =False, 使用 uwsgi+nginx 启动 loonflow r1.0.x,
13. 启动 task 服务
14. 启动调用方服务

CHAPTER 7

相关术语

工单：具体的待处理事项，用户新建的是工单，工单按照工作流的设计来实现不同状态不同处理人之间的流转

工作流：即工作流的设计，定义了工单的审批链、各状态的处理人、各状态可以执行的操作（提交、保存，处理完成，退回，关闭等等）、每个状态下显示哪些字段、哪些字段可以在哪些编辑

子工单: 主要用于工单流转存在子集的情况，如在项目开发周期中存在项目周期和应用周期两个层级，当项目处于开发中时，项目的多个涉及应用在项目开发中可能正处于不同的阶段 (代码编写、静态扫描、单元测试、完成开发等状态)。当应用状态都完成开发时将触发项目的状态到提测中。在这个场景中应用的工单即为项目工单的子工单。应用工单的父状态即为项目的“开发中”

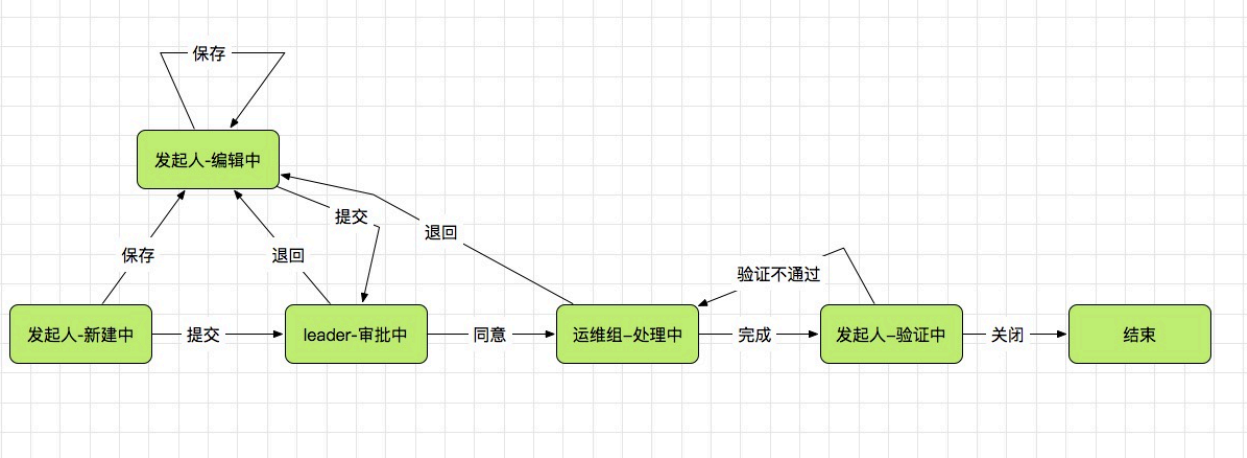
子工作流: 工作流的父子层级不体现在工作流记录中，而体现在状态记录中。在配置工作流时，可以给某个工作流的某个状态设置一个子工作流。可以在工作流的不同状态设置不同的子工作流。

流程图: 为了方便用户了解工作流的流转规则，可以通过流程图的方式展示给用户，如下

转交: 正常情况下工单的流转都是按照其对应工作流设定的规则来流转 (状态、处理人类型、处理人等). 在实际操作中，比如 A 提交了个工单，到达运维处理中状态，B 接单处理，B 在处理过程中发现自己其实处理不了，需要 C 才能处理。于是将工单转交给 C。

加签: 加签与转交不同。正常情况下工单的流转都是按照其对应工作流设定的规则来流转 (状态、处理人类型、处理人等). 在实际操作中, 比如 A 提交了个工单，到达运维处理中状态，B 接单处理，B 在处理过程中发现需要 C 做些操作或者提供些信息，才能处理，于是将工单加签给 C. C 处理完成后工单处理人会回到 B. 于是 B 可以继续处理

工单自定义字段与工作流自定义字段的区别: workflow 里面自定义字段规定工作流有哪些自定义的字段。比如配置一个请假的工作流。需要有请假天数这个字段。工单里面的自定义字段存的是自定义字段具体的值。



比如现在用于新建了一个请假工单，填写了请假天数。那么工单的自定义字段表中会保存这个值

工作流处理过程可以理解为工单状态的变化，如一个工作流处理过程中可以有：发起人新建中、发起人编辑中、部门经理审核中、技术人员处理中、发起人验证中、结束等状态，每个状态对应相应的处理人（如部门经理审核中这个状态下只有部门经理才可以处理该工单）。如用户在新建工单的时候处于“发起人新建中”，（用户）提交后工单处于“部门经理审核中”，部门经理（即“部门经理审核中”状态的处理人）审批通过后，工单的状态变更为“技术人员处理中”。注意:”转交”和”加签”使用场景不同，使用时前端需要做必要的说明，避免用户使用错误

CHAPTER 8

基础架构

LOONFLOW 分为两部分:

- 工作流配置的管理后台
- 提供 http api 供各个系统 (如果 oa、cmdb、运维系统、客服系统) 调用以完成各自系统定制化的工单需求

CHAPTER 9

代码结构

```
.
├── apps
|   ├── account # 用户应用
|   ├── manage  # 管理后台应用
|   ├── ticket  # 工单应用
|   └── workflow # 工作流应用
├── docs # 文档目录，后续会将文档全部迁移到 wiki 中
|   ├── apis # 接口文档
|   ├── images  # 相关图片
|   └── specs  # 代码规范
├── loonflow
|   └── __init__.py
|   └── url.py # url 路由主入口
|   └── wsgi.py  # wsgi 配置
├── media # 静态文件目录
|   ├── flowchart  # 工作流流程图，用户上次的流程图，后续将弃用
|   ├── notice_script  # 通知脚本目录
|   └── workflow_script  # 工作流执行脚本目录
├── requirements # 依赖文件目录
|   ├── common.txt  # 通用依赖
|   ├── dev.txt  # 开发环境依赖
|   ├── pro.txt  # 生产环境依赖
|   ├── test.txt  # 测试环境依赖
├── service # 服务层
```

(下页继续)

(续上页)

```
|   ├── account  # 用户相关服务
|   ├── common  # 通用服务
|   ├── manage  # 管理后台相关服务
|   ├── permission   # 权限相关服务
|   ├── ticket  # 工单相关服务
|   └── workflow  # 工作流相关服务
├── settings # 配置文件目录
|   └──  __init__.py
|   └──  common.py  # 通用配置
|   └──  dev.py  # 开发环境配置
|   └──  prod.py  # 生产环境配置
|   └──  test.py  # 测试环境配置
├── static # 静态文件，管理后台页面使用
|   ├── bower_components
|   ├── dist
|   └── plugins
├── templates  # 模板文件，管理后台页面使用，因为管理后台未前后端分离，所以有模板文件
|   ├── admin
|   ├── doc
|   ├── user_and_permission
|   └── workflow
└── tests  # 单元测试目录
    ├── test_models # model 层测试
    ├── test_services # service 层测试
    └── test_views  # view 层测试
```

CHAPTER 10

登录管理后台

使用部署过程中创建的 (python manage.py creatsuperuser) 用户名密码登录 http://host_ip:port

CHAPTER 11

同步用户信息

同步账户中用户、角色、用户角色 (用户具有的角色)、部门信息。loonflow 中工单的流转过程中需要根据用户的相关信息来确定新的处理人，因为不同公司用户组织架构信息保存方式各不一样 (如 ldap、AD 或者直接保存在企业微信、钉钉等等)，需要你自己编写用户组织信息的脚本，定时将你司的最新用户组织信息同步到 loonflow 中。可参考从 AD 中同步。非常欢迎大家将自己的脚本 pr 到 https://github.com/blackholll/loonflow-helper。

注意:

- loonflow 中的用户只是用于流转的时候确定新的处理人，无需要用户登录 loonflow 的管理后台，所以同步脚本中往 loonflow 插入用户记录时，密码随便插入。当然超级管理员除外 (超级管理员可以通过 python manage.py createsuperuser 命令来创建)
- 用户表中 dept_id 为 loonflow 的部门表中主键 id，非你司用户信息中的部门 id。同步部门时可以将你司部门 id 保存在 loonflow 部门表中的 label 字段中来关联.label 字段建议使用字段的 json 格式，方便以后扩展。如 {“source_dept_id”:11}

CHAPTER 12

工作流配置

12.1 自定义通知

自定义通知的管理权限仅限 loonflow 管理员，即用户表中 is_admin 为 true 的用户。通知 hook 添加后，工作流管理员在配置工作流时可以选择这些通知。其中 hook_url 为提供通知服务的地址，token 作为计算签名的密钥，当工作流选择了通知，那么通过此工作流创建的工单状态发送变化时都会触发 hook。loonflow 将会对这个 hook_url 发送一个 post 请求，请求头中包括签名，计算方式方式同 loonflow 校验 api 请求的签名算法。请求内容中包括工单最新的处理人信息，被请求方可以根据这个处理人信息发送通知消息。hook 服务方建议校验请求头中的签名后再响应请求。

签名算法

```
def gen_signature_by_token(cls, token: str)->tuple:
  md5_key = token
  timestamp = str(int(time.time()))
  ori_str = timestamp + md5_key
  tar_str = hashlib.md5(ori_str.encode(encoding='utf-8')).hexdigest()
  return True, dict(signature=tar_str, timestamp=timestamp)
```

loonflow 发送 hook 请求时，header 头中将包含 signature 和 timestamp。

请求内容中将包含以下参数

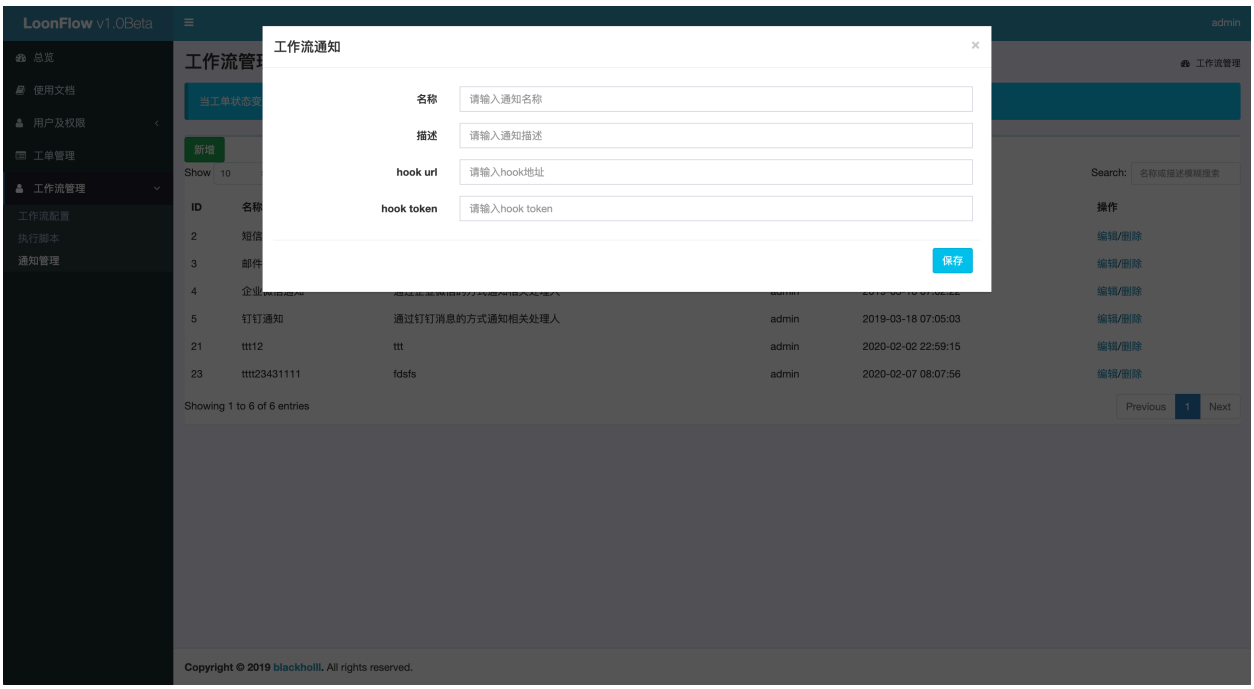
```
{
  'title_result': title_result, # 通知标题，在工作流配置中会配置标题模板，此处将根据工单的信息生成实际的标题
```

(下页继续)

(续上页)

```
  'content_result': content_result, # 通知内容，在工作流配置中会配置内容模板，此处将根据工单的信息生成实际的内容
  'participant_type_id': ticket_obj.participant_type_id, # 当前参与人类型，类型定义见文档中" 常量说明", hook 服务方需根据类型决定是否发送消息
  'ticket_value_info': ticket_value_info,  # 该字段中将包含工单所有字段的值，一般情况下发送消息时候根据标题和内容发送即可，
  'last_flow_log': last_flow_log,  # 工单的最近一条处理记录，发送通知消息时可以根据自己需要决定消息中是否包含这个信息
  'participant_info_list': participant_info_list  # 工单的当前处理人信息列表， 是一个数组，每个记录中都包含处理人的 username, alias, email, phone
 }
```

新建自定义通知



12.2 自定义执行脚本

0.3 版本开始 loonflow 已经支持配置状态时设置 hook 方式, 此方式可以替代执行脚本方式, 不建议使用执行脚本的方式。执行脚本存在以下缺陷

1. 脚本需要上传到 loonflow 服务端，且保存在 media 目录，虽然被重命名了，但是只要支持文件名就可以直接访问，而执行脚本中通常会包含密码等敏感信息，有安全风险

2. 因为需要保存脚本文件到 media 目录，loonflow 迁移时需要迁移此部分文件

3. 容器集群中部署时，还需要挂载分布式文件存储以保证容器迁移时可以继续访问磁盘内容，维护成本高

4.loonflow 是工作流引擎，执行脚本内容无法控制，很容易引入一些耗时 (如轮询) 或者耗资源的运算，这些运算将拖慢 loonflow 的性能

新建自定义脚本

12.3 工作流

创建一个工作流包括四个部分，创建工作流基础信息、添加工作流的自定义字段、添加状态、添加流转。在创建和编辑工作流时可以指定对应的管理员，只有工作流对应的管理员、工作流创建人、loonflow 超级管理员才有工作流的编辑权限。拥有工作流管理员权限的用户可以新增和维护自己相关 (自己创建的、作为管理员的) 的工作流

选择通知：loonflow 超级管理员在通知管理中添加通知后，工作流管理员即可在新建或编辑工作流时候选择这些通知。

工单查看权限校验：开启后，以工单的关联人 (创建人、曾经的处理人) 的名义调用工单详情接口才可以返回详情信息

限制表达式：默认”{}”，限制周期 ({“period”:24} 24 小时), 限制次数 ({“count”:1} 在限制周期内只允许提交 1 次), 限制级别 ({“level”:1} 针对 (1 单个用户 2 全局) 限制周期限制次数, 默认特定用户); 允许特定人员提交 ({“allow_persons”:” zhangsan,lisi” } 只允许张三提交工单, { “allow_depts”:” 1,2” } 只允许部门 id 为 1

和 2 的用户提交工单，{ “allow_roles” :” 1,2” } 只允许角色 id 为 1 和 2 的用户提交工单)。符合限制表达式规则时调用创建工单接口会返回 code=-1, 同时 msg 中将有相应的说明

标题模板：用于发送通知时，确定通知的标题

内容模板：用于发送通知，确定通知的内容

管理员：工作流的创建人和这里的管理员有权限修改配置此工作流，另外 loonflow 的超级管理员拥有对所有工作流的配置查看权限

每个类型的工作流有各自不同的字段，通过自定义字段可以为工作流新增独有的字段

字段标识：自定义字段需要有个标识，需要为英文字母 + 下划线，如 reason、problem_description

字段名称：自定义字段的名称，如请假理由、问题描述等

字段类型：支持字符串、整形、浮点型、布尔、日期、日期时间、单选框、多选框、下拉列表、多选下拉列表、文本域、用户名、多选用户名、附件。其中单选、多选、下拉、多选下拉默认支持给定可选范围的情况，如果你需要可选项通过接口获取，请使用字符串类型，然后在” 标签 “中填写前后端约定好的信息。由前端针对这个字段特殊处理。附件类型需要调用方自己处理上传逻辑后将附件路径随其他字段一起提交，loonflow 只保存附件路径。关于用户名类型字段，调用方前端可以显示一个搜索用户的可选框，然后提交被选择的 username 或者逗号隔开的多个被选择 username(多选用户名类型)。

顺序 ID：用于指定工单详情界面中字段的布局顺序，调用方前端需要根据这个 id 来布局字段

默认值：调用方前端当用户未填写内容时，可以根据这个信息给定默认值显示

布尔显示定义：当字段类型是布尔类型时，可以通过定义来显示不同的文字，如 { “1” :” 是” ,” 0” :” 否” } 或 { “1” :” 需要” ,” 0” :” 不需要” }，注意数字也需要引号

选项：用于指定单选、多选、下拉框、多选下拉框字段的可选项目，格式为 json 如:{“1”:”中国”，“2”:”美国”},{“ch”:”中国”，“us”:”美国”} 注意数字也需要引号

标签：用于特殊字段，通过与调用方约定标签内容，可以实现任意的字段表现形式，格式为 json, 如 {“table”:”http://xxx.com/table”} 可以表示从 http://xxx.com/table 获取信息，然后显示为一个表格

模板：用于文本域类型字段前端根据此模板显示默认值

12.4 状态

工单会存在各种状态，工单的处理过程也就是工单的状态的变化，loonflow 对于工作流的状态支持各种灵活的配置

名称：状态的名称，如创建中、编辑中、tl 审批中、结束等等

是否隐藏：开启后，获取工单步骤接口将不返回此状态 (工单当前处于隐藏状态除外)。因为 step 图是线性的，而 loonflow 工作流状态是支持环状关联的，所以在 step 图需要将一些旁支状态隐藏掉

顺序 id：用于指定“获取工单步骤”接口中返回的状态 id，因为 step 图是线性的，而 loonflow 工作流状态是支持环状关联的，所以在 step 图显示时需要给定每个状态的显示顺序

状态类型：状态分为三个状态，初始状态、普通状态、结束状态，一个工作必须包含一个初始状态、一个结束状态、0-N 个普通状态，初始状态用于创建工单时根据这个状态及状态关联的流转来确定可以做的提交操作 (提交、新建、保存等等)。结束状态表示工单到此状态后将无法被继续处理，建议只设置一个结束状态，否则“强制关闭工单”接口将无法正常关闭工单 (无法确定结束状态是哪个)

是否记忆最后处理人：开启后，到达此状态时会先检查之前是否有人在此状态处理过，如果有则处理人为最后一次处理的人，使用场景如：运维处理中状态下处理人有 A、B、C，其中 A 处理了工单，然后到达发起人

确认状态时，发起人发现处理的有问题，那么发起人可能是希望将工单退回到之前处理的 A。而不是 A、B、C 都收到工单

参与人类型：参与人类型包括个人、多人、部门、角色、脚本 (建议使用 hook 替代)、工单字段、父工单字段、hook、无。注意：如果需要在此状态创建子工单，需要将参与人类型设置为个人，参与人使用 loonrobot

参与人：参与人信息根据参与人类型不同而不同，如果参与人类型是个人，那么参与人需要填写用户的 username。如果参与人类型是多人，那么参与人需要填写多个 username, 用逗号隔开，如 zhansan,lisi。如果参与人类型是部门，则参与人填写 loonflow 部门记录中的部门 id(需要多个部门处理时，逗号隔开部门的 id，如 1,3)。如果参与人类型是角色，则参与人填写 loonflow 角色记录中的 id. 如果参与人类型是脚本（建议使用 hook 方式替代），则填写“工作流配置”-“执行脚本”中的脚本 id。如果参与人类型是工单字段，则参与人填写工作流基础字段或者此工作流定义的自定义字段的字段标识如 username, agent。如果参与人类型是父工单字段，则填写工单的对应字段的标识。如果参会人类型填写无，那么处理人信息留空。如果参与人类型是 hook，那么参与人填写 { “hook_url” :” http://xxx.com/xxx” , “hook_token” :” xxxx” , “wait” :true, “extra_info” :” xxx” }，其中 hook_url 是要触发 hook 的目标地址，hook_token 用于签名计算, 你的 hook 服务端需要保存此 hook_token，hook 签名算法如下方代码区, 触发 hook 请求时候会将 timestamp 和 signature 添加到请求头中，hook 服务端应该在收到请求后按照相同的签名算法先校验签名的有效性然后才响应 hook 请求。wait 的值可以是 true 或者 false, 如果 wait 的值是 false 那么工单触发 hook 成功 (hook 服务端返回 json 类型数据，其中 code=0) 后直接进入新的状态 (触发失败的话即 code=-1 或者服务端无响应或者 http status 非 200 工单会标记 script_run_last_result 为 False，你可以调用“重试工单脚本/任务”重新触发 hook)，如果 wait 的值是 true 那么工单触发 hook 后会停留在当前状态，直到 hook 方回调 (回调逻辑见文档中“工单相关接口”-“工单 hook 回调”)loonflow 成功 (请求参数中 result=True) 后工单的状态才继续流转。extra_info(非必填) 可以用于传一些额外的信息，loonflow 会将这个信息连同工单信息传给 hook 服务端。

```
import time
timestamp = str(time.time())[:10]
ori_str = timestamp + token
signature = hashlib.md5(ori_str.encode(encoding='utf-8')).hexdigest()
```

hook 触发时 loonflow 向 hook_url 服务端 post 请求时带的数据如下：

```
{
  "id":11,
  "sn":"loonflow202002020001",
  "title":"xxx",
  "state_id":1,
  ...., //等等工单的所有字段的值
  "extra_info": "xxxx", // 此处如果你配置 hook 的时候指定了 extra_info 那么会有这个字段，如果没配置就没这个信息
}
```

分配方式：分配方式包括直接处理、主动接单、随机分配、全部处理。如果设置为直接处理，工单的当前处理人可以直接点击配置的流转 (如同意、拒绝、完成) 来处理。如果设置为主动接单，则当前处理人需要先接单，然后才可以按照配置的流转来处理 (表现形式为获取用户可执行的操作接口只会返回接单这个流转，具体参考关于接单接口的纤细描述)。如果设置为随机分配，那么系统会自动将工单处理人设置为有处理权限

的人之间的一个，只有这个人可以处理。如果设置为全部处理，则需要此状态的所有参与人都处理完且处理操作一致，才会改变工单的状态。

表单字段：状态的表单字段用于定义当用户有对该工单处理权限时，当前状态工单详情中应该显示哪些字段以及这些字段是否可编辑。其中 1 表示只读，2 表示必填，3 表示可选，示例：{ “gmt_created” :1,” title” :2, “sn” :1}，内置特殊字段 participant_info.participant_name: 当前处理人信息 (部门名称、角色名称)，state.state_name: 当前状态的状态名，workflow.workflow_name: 工作流名称，

状态标签：json 格式，可以使用此配置实现不同状态各种定制化需求，如在服务器申请工单的 tl 审批阶段显示发起人拥有的所有服务器权限列表，那么状态标签可以设置为 { “display_server” :1}，标签具体内容与调用方约定好即可，

12.5 流转

流转用于定义每个状态之间的变化途径。

名称：定义流转的名称，表现为工单详情中用户可以点击的操作，如 “提交”、“保存”、“同意”、“拒绝”、“完成”、“关闭” 等等

流转类型：流转类型包括常规流转和定时器流转。如果选择定时器流转，需要设定定时器的时间。

定时器：单位是秒，如果流转配置了定时器，当工单处于这个流转的源状态停留对应定时器的时间后未做任何操作，那么定时器会触发工单自动流转到下个状态。使用场景如：客服类工单超过 SLA 无人处理，自动流转到客服团队 leader 处理

源状态：流转的源状态，即这个流转是在某个状态下可以执行

目标状态：触发流转后工单的状态将改变为这个目标状态。当流转类型为条件流转时，这里的目标状态将无效，以条件表达式中设定的条件目标状态来流转

条件表达式：流转条件表达式，根据表达式中的条件来确定流转的下个状态，格式为 [{ “expression” :” {days} > 3 and {days} ≤10”，“target_state_id” :11},{ “expression” :” {days} >10”，“target_state_id” :12}] 其中 {} 用于填充工单的字段 key, 运算时会换算成实际的值，当符合条件下个状态将变为 target_state_id 中的值, 表达式只支持简单的运算或 datetime/time 运算. loonflow 会以首次匹配成功的条件为准，所以多个条件不要有冲突

属性类型：因为别的审批系统中对于每个操作可能只有同意还是拒绝，所以此处加个属性用于与其他审批系统对接，另外也根据根据这个属性来判断工单是否被拒绝 (工单列表查询支持是否被拒绝这个条件)

是否校验必填项: 默认在用户点击操作的时候需要校验工单表单的必填项, 如果设置为否则不检查。用于如”退回”属性的操作，不需要填写表单内容

点击弹窗提示：可以用于当用户在处理工单时做特定操作时，弹窗提示信息。如用户点击”拒绝”时弹窗提示要求用户确认是否真的拒绝，避免点错

弹窗内容: 当启用”点击弹窗提示“时可以设置弹窗中的内容

用户及权限相关管理只有超级管理员用户才有权限查看或者编辑

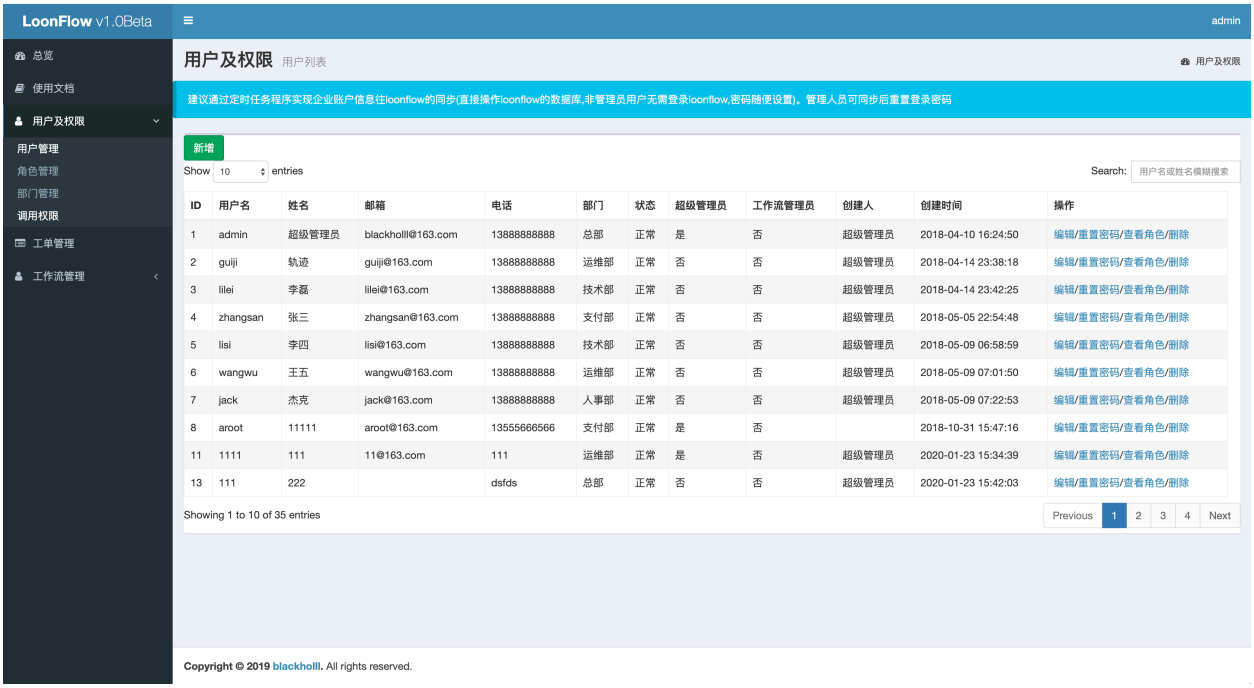
CHAPTER 13

用户管理

loonflow 管理后台提供的用户管理功能适用于开始的功能测试，或者账户信息微调。建议通过定时任务程序实现企业账户信息往 loonflow 的同步 (直接操作 loonflow 的数据库, 非管理员用户无需登录 loonflow, 密码随便设置)。管理人员可同步后重置登录密码。管理分为两类：超级管理员、工作流管理员

超级管理员：拥有 loonflow 管理后台所有功能的权限, 包括工作流管理员的所有权限

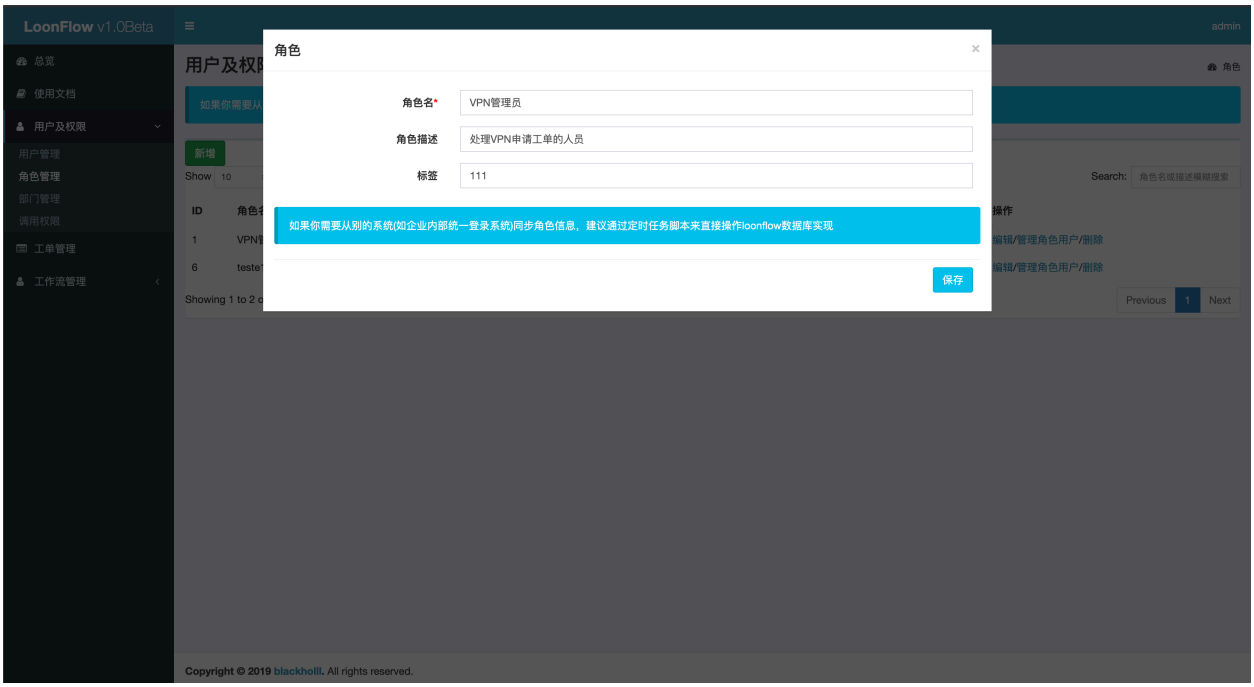
工作流管理员：允许登录 loonflow 管理后台拥有工作流配置及对应有权限工作流的相关管理功能



CHAPTER 14

角色管理

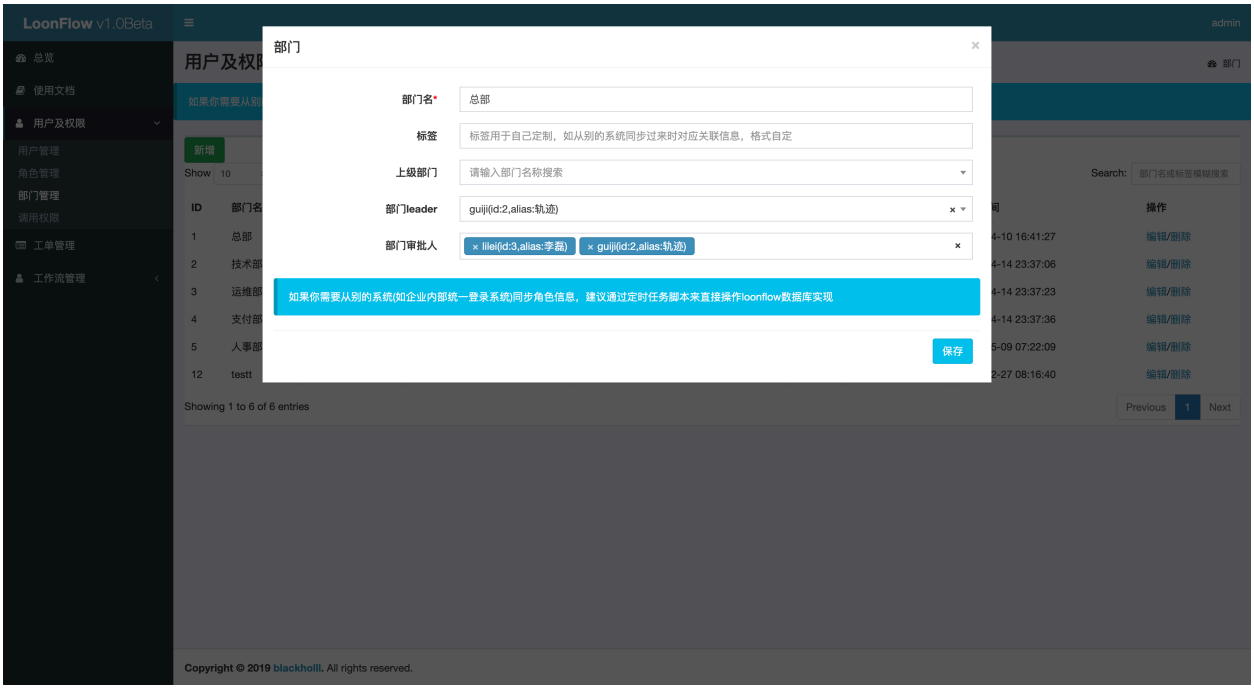
此处角色管理可以定义一系列角色来作为工作流配置中状态的处理人。如可以定义一线处理人、pc 故障维修人等等角色。如果你需要这个角色与公司的用户体系一致，建议通过定时任务脚本直接操作 loonflow 的数据库来实现



CHAPTER 15

部门管理

此处部门信息用于工作流配置是状态处理人为部门、变量 (创建人 tl) 的情况。如果你需要这个部门信息与公司用户体系一致，建议通过定时任务脚本直接操作 loonflow 的数据库来实现



CHAPTER 16

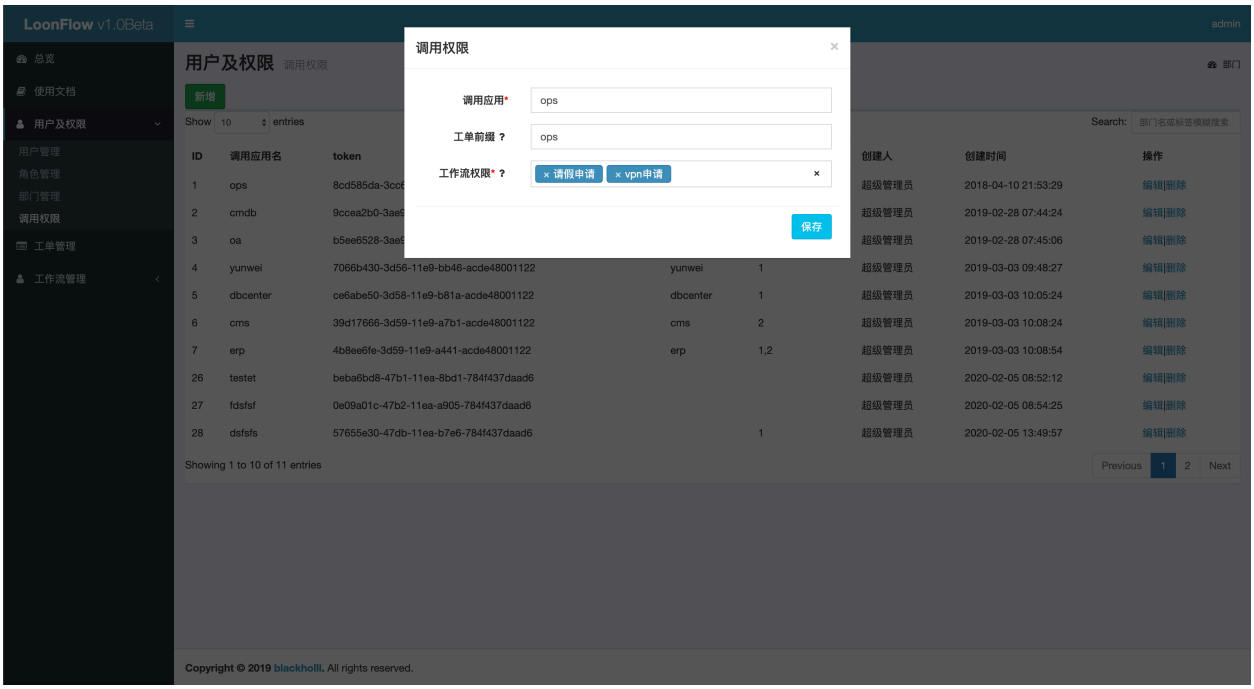
调用权限

loonflow 给每个调用方应用分配一个 token, 用于生成调用时请求头中的签名。同时可以限定每个应用可以操作的工作流列表，以及定义每个应用创建工单时，工单流水号的规则。

调用应用：建议用英文字母，如客服系统调用可以用 kefu,oa 系统调用可以用 oa

工单前缀: 用于指定生成工单的流水号规则，如设定为 kefu, 则生成的流水号为 kefu_202003080001

工作流权限: 指定使用此应用调用 loonflow 接口时，可以操作的工作流列表，创建工单时只能选择这些工作流，查看工单列表时也只能选择这些类型的工单


LoonFlow v1.0Beta
admin
总览
使用文档
用户及权限
用户管理
角色管理
部门管理
调用权限
工单管理
工作流管理
用户及权限 调用权限
部门
新增
Show 10 entries
Search:
部门名或标签模糊搜索
ID 调用应用名 token 创建人 创建时间 操作
1 ops 8cd585da-3cc6 超级管理员 2018-04-10 21:53:29 编辑|删除
2 cmdb 9ccea2b0-3ae9 超级管理员 2019-02-28 07:44:24 编辑|删除
3 oa b5ee6528-3ae5 超级管理员 2019-02-28 07:45:06 编辑|删除
4 yunwei 7066b430-3d56-11e9-bb46-acde48001122 yunwei 1 超级管理员 2019-03-03 09:48:27 编辑|删除
5 dbcenter ce6abe50-3d58-11e9-b81a-acde48001122 dbcenter 1 超级管理员 2019-03-03 10:05:24 编辑|删除
6 cms 39d17666-3d59-11e9-a7b1-acde48001122 cms 2 超级管理员 2019-03-03 10:08:24 编辑|删除
7 erp 4b8ee6fe-3d59-11e9-a441-acde48001122 erp 1,2 超级管理员 2019-03-03 10:08:54 编辑|删除
26 testet beba6bd8-47b1-11ea-8bd1-784f437daad6 超级管理员 2020-02-05 08:52:12 编辑|删除
27 fdsfsf 0e09a01c-47b2-11ea-a905-784f437daad6 超级管理员 2020-02-05 08:54:25 编辑|删除
28 dsfsfs 57655e30-47db-11ea-b7e6-784f437daad6 1 超级管理员 2020-02-05 13:49:57 编辑|删除
Showing 1 to 10 of 11 entries
Previous 1 2 Next
Copyright © 2019 blackholll. All rights reserved.
调用权限
调用应用*
ops
工单前缀 ?
ops
工作流权限* ?
× 请假申请 × vpn申请
保存

CHAPTER 17

接口调用鉴权

Loonflow 作为工作流引擎，正确是的使用姿势是各个系统的后端通过 http api 调用按照各自的需求来完成工单展示、工单新建、工单处理逻辑在 loonflow 的管理后台中”账户-调用 token”中新新增记录. 填写调用方 app_name 新增后会生成一个签名 token. 调用方将签名信息写到 http header 中来调用具体的 api 签名算法如下:

```
import time
timestamp = str(time.time())[:10]
ori_str = timestamp + token
signature = hashlib.md5(ori_str.encode(encoding='utf-8')).hexdigest()
```

api 调用:

```
import requests

headers = dict(signature=signature, timestamp=timestamp, appname=app_name,␣
↪username=username)

# get
get_data = dict(per_page=20, category='all')
r = requests.get('http://127.0.0.1:8000/api/v1.0/tickets', headers=headers,␣
↪params=get_data)
result = r.json()

# post
data = dict(target_username='lisi', suggestion='请协助提供更多信息')
```

(下页继续)

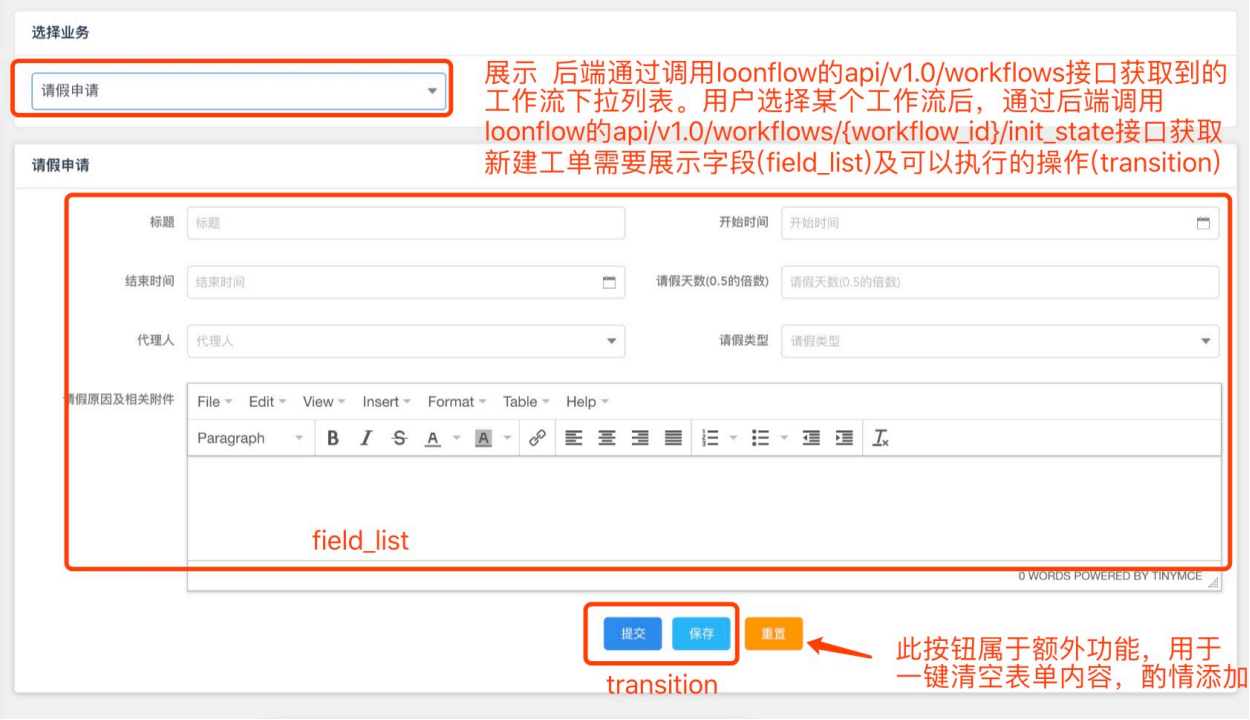
(续上页)

```
r = requests.post('http://127.0.0.1:8000/api/v1.0/tickets/{ticket_id}/add_node',␣
↪headers=headers, json=data)
result = r.json()

# patch
requsts.patch, 传参同 post

# put
requests.put, 传参同 post
```

注意: 开发阶段如果需要在 postman 中测试接口，避免每次都需要重新生成签名，可以将 service.permission.api_permission.ApiPermissionCheck 中签名有效期改长些

CHAPTER 18

接口调用逻辑




流程图
新建中
2 TL审批中
3 人事部门-处理中
4 结束
后端调用loonflow的api/v1.0/tickets/{ticket_id}/flowsteps接口获取step
请假申请 详情
流水号 loonflow_201805280002
标题 请假申请
开始时间 2018-05-29 00:00:00
结束时间 2018-05-30 00:00:00
请假天数(0.5的倍数) 1
代理人 admin
请假类型 调休
创建人 webb
创建时间 2018-05-28 16:27:02
请假原因及相关附件 因事请假一天，望批准，谢谢！
后端调用loonflow的api/v1.0/tickets/{ticket_id}接口获取工单详情，其中field_list信息中包含了需要显示的字段以及每个字端的类型、属性(只读、必填、可选)
处理意见
请输入处理意见
同意
拒绝
后端调用loonflow的api/v1.0/tickets/{ticket_id}/transitions获取用户可以执行的操作
用户填写表单中必要的信息后点击 操作按钮，前端提交数据到调用方的后端。 调用方后端调用loonflow的api/v1.0/tickets/{ticket_id} —patch method来完成工单处理
工单操作日志
节点名称
参与者
操作时间
处理意见
新建中
webb
2018-05-28 16:27:02
后端调用loonflow的api/v1.0/tickets/{ticket_id}/flowlogs

CHAPTER 19

工作流相关接口

19.1 获取工作流列表

- url

api/v1.0/workflows

- method

get

- 使用场景

获取到工作流列表后，用户选择对应的工作流来新建对应的工单。如果需要多级类型，可以在调用方系统保存对应关系。如调用方的“权限申请-VPN 权限申请“对应 loonflow 中 id 为 1 的 workflow, 调用方的“权限申请-服务器权限申请“对应 loonflow 中 id 为 2 的 workflow

- 请求参数

参数名	类型	必填	说明
page	int	否	页码，默认 1
per_page	int	否	每页个数，默认 10
name	varchar	否	支持根据 workflow name 模糊查询

- 返回数据

```
{
  "code": 0,
  "data": {
    "total": 2,
    "page": 1,
    "per_page": 10,
    "value": [{
      "name": "请假申请",
      "creator": "admin",
      "description": "请假申请",
      "gmt_created": "2018-04-23 20:49:32"
    }, {
      "name": "vpn 申请",
      "creator": "admin",
      "description": "vpn 权限申请",
      "gmt_created": "2018-05-06 12:32:36"
    }]
  },
  "msg": ""
}
```


19.2 获取工作流初始状态

- url

api/v1.0/workflows/{workflow_id}/init_state

- method

get

- 请求参数

无

- 使用场景

用于获取创建工单时对应工作流的初始状态信息，返回内容包括创建工单时需要填写的表单内容，可以执行的提交操作

- 返回数据


```
{
  "msg": "",
  "code": 0,
  "data": {
```


(下页继续)

(续上页)


```
    "order_id": 0,
    "workflow_id": 1,
    "name": "新建中",
    "participant_type_id": 1,
    "distribute_type_id": 1,
    "participant": "wangfei",
    "is_hidden": false,
    "type_id": 1,
    "gmt_created": "2018-04-23 20:53:33",
    "id": 1,
    "transition": [{
      "transition_id": 1,
      "transition_name": "提交"
    }, {
      "transition_id": 2,
      "transition_name": "保存"
    }],
    "sub_workflow_id": 0,
    "creator": "admin",
    "label": {},
    "field_list": [{
      "order_id": 20,
      "field_key": "title",
      "field_attribute": 2,
      "value": null,
      "name": "标题",
      "field_type_id": 5
    }, {
      "order_id": 35,
      "field_key": "leave_proxy",
      "field_attribute": 2,
      "field_type_id": 60,
      "field_value": null,
      "field_name": "代理人",
      "field_choice": {}
    }, {
      "order_id": 25,
      "field_key": "leave_end",
      "field_attribute": 2,
      "field_type_id": 30,
      "field_value": null,
      "field_name": "结束时间",
      "field_choice": {}
```


(下页继续)

(续上页)


```
    }, {
      "order_id": 20,
      "field_key": "leave_start",
      "field_attribute": 2,
      "field_type_id": 30,
      "field_value": null,
      "field_name": "开始时间",
      "field_choice": {}
    }, {
      "order_id": 40,
      "field_key": "leave_type",
      "field_attribute": 2,
      "field_type_id": 40,
      "field_value": null,
      "field_name": "请假类型",
      "field_choice": {
        "1": "年假",
        "2": "调休",
        "3": "病假",
        "4": "婚假"
      }
    }, {
      "order_id": 45,
      "field_key": "leave_reason",
      "field_attribute": 2,
      "field_type_id": 55,
      "field_value": null,
      "field_name": "请假原因及相关附件",
      "field_choice": {}
    }, {
      "order_id": 30,
      "field_key": "leave_days",
      "field_attribute": 2,
      "field_type_id": 5,
      "field_value": null,
      "field_name": "请假天数 (0.5 的倍数)",
      "field_choice": {}
    }]
  }
}
```

19.3 获取工作流状态详情

- url

api/v1.0/workflows/states/{state_id}

- method

get

- 请求参数

无

- 使用场景

略

- 返回数据


```
{
  "code": 0,
  "data": {
    "id": 1,
    "name": "\u65b0\u5efa\u4e2d",
    "workflow_id": 1,
    "sub_workflow_id": 0,
    "distribute_type_id": 1,
    "is_hidden": false,
    "order_id": 0,
    "type_id": 1,
    "participant_type_id": 1,
    "participant": "wangfei",
    "state_field": {
      "title": 2,
      "leave_start": 2,
      "leave_end": 2,
      "leave_days": 2,
      "leave_proxy": 2,
      "leave_type": 2,
      "leave_reason": 2
    },
    "label": {},
    "creator": "admin",
    "gmt_created": "2018-04-23 20:53:33"
  },
  "msg": ""
}
```

19.4 获取工作流状态列表

- url

api/v1.0/workflows/{workflow_id}/states

- method

get

- 使用场景

可用于用户查询工单列表时选择工作流类型后，显示该工作流类型拥有的状态，然后可以再根据工单当前状态来查询。另外可用于管理员干预工单强制修改状态时允许选择的目标状态

- 返回数据


```
{
  "code": 0,
  "data": {
    "value": [{
      "id": 1,
      "creator": "admin",
      "gmt_created": "2018-04-23 20:53:33",
      "gmt_modified": "2018-05-13 11:42:11",
      "is_deleted": false,
      "name": "\u65b0\u5efa\u4e2d",
      "workflow_id": 1,
      "sub_workflow_id": 0,
      "is_hidden": false,
      "order_id": 0,
      "type_id": 1,
      "remember_last_man_enable": false,
      "participant_type_id": 1,
      "participant": "wangfei",
      "distribute_type_id": 1,
      "state_field_str": {
        "title": 2,
        "leave_start": 2,
        "leave_end": 2,
        "leave_days": 2,
        "leave_proxy": 2,
        "leave_type": 2,
        "leave_reason": 2
      },
      "label": {},
      "participant_info": {
```


(下页继续)

(续上页)


```
        "participant": "wangfei",
        "participant_name": "wangfei",
        "participant_type_id": 1,
        "participant_type_name": "\u4e2a\u4eba",
        "participant_alias": "wangfei"
      }
    }, {
      "id": 2,
      "creator": "admin",
      "gmt_created": "2018-04-30 15:45:48",
      "gmt_modified": "2018-05-14 06:44:10",
      "is_deleted": false,
      "name": "\u53d1\u8d77\u4eba-\u7f16\u8f91\u4e2d1",
      "workflow_id": 1,
      "sub_workflow_id": 2,
      "is_hidden": true,
      "order_id": 2,
      "type_id": 0,
      "remember_last_man_enable": false,
      "participant_type_id": 5,
      "participant": "creator",
      "distribute_type_id": 1,
      "state_field_str": {
        "leave_end": 3,
        "leave_days": 3,
        "sn": 1,
        "state.state_name": 1,
        "leave_proxy": 3,
        "title": 3,
        "gmt_created": 1,
        "creator": 1,
        "leave_start": 3,
        "leave_reason": 3,
        "leave_type": 3
      },
      "label": {},
      "participant_info": {
        "participant": "creator",
        "participant_name": "creator",
        "participant_type_id": 5,
        "participant_type_name": "\u53d8\u91cf",
        "participant_alias": "\u5de5\u5355\u521b\u5efa\u4eba"
      }
```


(下页继续)

(续上页)


```
    }, {
      "id": 3,
      "creator": "admin",
      "gmt_created": "2018-04-30 15:46:42",
      "gmt_modified": "2018-11-27 07:20:33",
      "is_deleted": false,
      "name": "TL\u5ba1\u6279\u4e2d",
      "workflow_id": 1,
      "sub_workflow_id": 0,
      "is_hidden": false,
      "order_id": 3,
      "type_id": 0,
      "remember_last_man_enable": true,
      "participant_type_id": 5,
      "participant": "creator_tl",
      "distribute_type_id": 3,
      "state_field_str": {
        "leave_reason": 1,
        "leave_start": 1,
        "leave_type": 1,
        "creator": 1,
        "gmt_created": 1,
        "title": 1,
        "leave_proxy": 1,
        "sn": 1,
        "leave_end": 1,
        "leave_days": 1
      },
      "label": {
        "tech_er_in": "qa"
      },
      "participant_info": {
        "participant": "creator_tl",
        "participant_name": "creator_tl",
        "participant_type_id": 5,
        "participant_type_name": "\u53d8\u91cf",
        "participant_alias": "\u5de5\u5355\u521b\u5efa\u4eba\u7684tl"
      }
    }, {
      "id": 4,
      "creator": "admin",
      "gmt_created": "2018-04-30 15:47:58",
      "gmt_modified": "2018-05-13 11:42:59",
```


(下页继续)

(续上页)


```
    "is_deleted": false,
    "name": "\u4eba\u4e8b\u90e8\u95e8-\u5904\u7406\u4e2d",
    "workflow_id": 1,
    "sub_workflow_id": 0,
    "is_hidden": false,
    "order_id": 4,
    "type_id": 0,
    "remember_last_man_enable": false,
    "participant_type_id": 1,
    "participant": "admin",
    "distribute_type_id": 1,
    "state_field_str": {
      "sn": 1,
      "title": 1,
      "leave_start": 1,
      "leave_end": 1,
      "leave_days": 1,
      "leave_proxy": 1,
      "leave_type": 1,
      "creator": 1,
      "gmt_created": 1,
      "leave_reason": 1
    },
    "label": {},
    "participant_info": {
      "participant": "admin",
      "participant_name": "admin",
      "participant_type_id": 1,
      "participant_type_name": "\u4e2a\u4eba",
      "participant_alias": "\u8d85\u7ea7\u7ba1\u7406\u5458"
    }
  }, {
    "id": 5,
    "creator": "admin",
    "gmt_created": "2018-04-30 15:51:41",
    "gmt_modified": "2018-05-11 06:52:39",
    "is_deleted": false,
    "name": "\u7ed3\u675f",
    "workflow_id": 1,
    "sub_workflow_id": 0,
    "is_hidden": false,
    "order_id": 6,
    "type_id": 2,
```


(下页继续)

(续上页)


```
      "remember_last_man_enable": false,
      "participant_type_id": 0,
      "participant": "",
      "distribute_type_id": 1,
      "state_field_str": {},
      "label": {},
      "participant_info": {
        "participant": "",
        "participant_name": "",
        "participant_type_id": 0,
        "participant_type_name": "",
        "participant_alias": ""
      }
    }],
    "per_page": 10,
    "page": 1,
    "total": 5
  },
  "msg": ""
}
```

CHAPTER 20

工单相关接口

20.1 获取工单列表

- url

api/v1.0/tickets

- method

get

- 请求参数

参数名	类型	必填	说明
sn	varchar	否	流水号，支持根据 sn 的前几位模糊查询
title	varchar	否	工单标题，模糊查询
create_start	varchar	否	创建时间起, 如果创建时间起和创建时间止都不提供，则只返回最近一年的数据
create_end	varchar	否	创建时间止, 如果创建时间起和创建时间止都不提供，则只返回最近一年的数据
workflow_ids	varchar	否	工作流 ids，逗号隔开多个工作流 id, 如”1,2,3”
state_ids	varchar	否	状态 ids, 逗号隔开多个状态 id, 如”1,2,3”
ticket_ids	varchar	否	工单 ids, 逗号隔开多个 id，如”1,2,3”
reverse	varchar	否	是否按照创建时间倒序,”0”或者”1”，默认倒序
page	int	否	页码，默认 1
per_page	varchar	否	每页个数，默认 10
act_state_id	int	否	工单的进行状态, 0 草稿中 (也就是初始状态)、1 进行中 2 被退回 3 被撤回 4 已完成 5 已关闭
category	varchar	是	类型 (‘all’: 所有工单，‘owner’: 我创建的工单，‘duty’: 我的待处理工单，‘relation’: 我的关联工单 [包括我新建的、我处理过的、曾经需要我处理过的工单。注意这里只考虑历史状态，工单将来状态的处理人不考虑]，‘worked’: 我处理过的工单)

- 返回数据


```
{
  "msg": "",
  "code": 0,
  "data": {
    "value": [{
      "participant_info": {
        "participant_type_id": 1,
        "participant": "1",
        "participant_name": "zhangsan",
        "participant_type_name": "个人",
        "participant_alias": "张三"
      },
```


(下页继续)

(续上页)


```
      "gmt_created": "2018-05-15 07:16:38",
      "parent_ticket_state_id": 0,
      "state": {
        "state_name": "发起人-确认中",
        "state_id": 10
      },
      "creator": "lilei",
      "parent_ticket_id": 0,
      "title": "vpn 申请",
      "gmt_modified": "2018-05-22 07:26:54",
      "workflow": {
        "workflow_name": "vpn 申请",
        "workflow_id": 2
      },
      "sn": "loonflow_201805150001",
      "id": 17
    }],
    "total": 1,
    "page": 1,
    "per_page": 10
  }
}
```


20.2 新建工单

- url

api/v1.0/tickets

- method

post

- 请求参数

参数名	类型	必填	说明
workflow_id	int	是	工作流 id(工单关联的工作流的 id)
transition_id	int	是	新建工单时候的流转 id（通过 workflow/{id}/init_state 接口可以获取新建工单时允许的 transition）
parent_ticket_id	int	否	父工单的 id(用于子工单的逻辑，如果新建的工单是某个工单的子工单需要填写父工单的 id)
parent_ticket_state_id	int	否	父工单的状态（子工单是和父工单的某个状态关联的），如果提供 parent_ticket_id，则此参数也必须
suggestion	varchar	否	处理意见（与处理工单类型，用户在处理工单的时候点击了按钮操作可以填写附加的一些意见如: 麻烦尽快处理）
其他必填字段	N/A	否	其他必填字段或可选字段（在配置工作流过程中, 会配置工作流初始状态必填和可选的字段。在新建工单时候必须提供必填字段。如请假申请工单，配置了自定义字段请假天数 days，工单初始状态也设置了 days 为必填，那么新建此类工单时候就必选提供 days)

- 返回数据

```
{
  "msg": "",
  "code": 0,
  "data": {
    "ticket_id": 1
  }
}
```

20.3 获取工单详情

- url

api/v1.0/tickets/{ticket_id}

- method

get

- 请求参数

无

- 返回数据

```
{
  "code": 0,
  "msg": "",
  "data": {
    "value": {
      "workflow_id": 2,
      "in_add_node": true,
      "gmt_created": "2018-05-15 07:16:38",
      "id": 17,
      "relation": "guiji,wangwu,lilei",
      "title": "vpn\u7533\u8bf72",
      "sn": "loonflow_201805150001",
      "parent_ticket_id": 0,
      "creator": "lilei",
      "script_run_last_result": true,
      "gmt_modified": "2018-05-22 07:26:54",
      "act_state_id": 1,
      "multi_all_person": "{}",
      "creator_info": {
        "email": "lilei@163.com",
        "alias": "\u674e\u78ca",
        "dept_info": {
          "creator_info": {
            "creator_id": 1,
            "creator_alias": "\u8d85\u7ea7\u7ba1\u7406\u5458"
          },
          "leader": "lilei",
          "parent_dept_info": {
            "parent_dept_name": "\u603b\u90e8",
            "parent_dept_id": 1
          },
          "approver_info": [],
          "parent_dept_id": 1,
          "name": "\u6280\u672f\u90e8",
          "is_deleted": false,
          "creator": "admin",
          "gmt_modified": "2018-05-09 06:45:27",
          "label": "",
          "id": 2,
          "approver": "",
          "gmt_created": "2018-04-14 23:37:06",
          "leader_info": {
            "leader_alias": "\u674e\u78ca",
            "leader_username": "lilei"
```


(下页继续)

(续上页)


```
        }
      },
      "username": "lilei",
      "phone": "13888888888",
      "is_active": true
    },
    "participant_type_id": 3,
    "state_id": 10,
    "is_end": false,
    "is_deleted": false,
    "field_list": [{
      "field_value": "loonflow_201805150001",
      "label": {},
      "boolean_field_display": {},
      "field_type_id": 5,
      "field_template": "",
      "field_choice": {},
      "field_key": "sn",
      "field_attribute": 1,
      "description": "\u5de5\u5355\u7684\u6d41\u6c34\u53f7",
      "default_value": null,
      "order_id": 10,
      "field_name": "\u6d41\u6c34\u53f7"
    }, {
      "field_value": "\u53d1\u8d77\u4eba-\u786e\u8ba4\u4e2d",
      "label": {},
      "boolean_field_display": {},
      "field_type_id": 5,
      "field_template": "",
      "field_choice": {},
      "field_key": "state.state_name",
      "field_attribute": 1,
      "description": "\u5de5\u5355\u5f53\u524d\u72b6\u6001\u7684\u540d\u79f0",
      "default_value": null,
      "order_id": 41,
      "field_name": "\u72b6\u6001\u540d"
    }, {
      "field_value": "\u603b\u90e8",
      "label": {},
      "boolean_field_display": {},
      "field_type_id": 5,
      "field_template": "",
      "field_choice": {},
```


(下页继续)

(续上页)


```
        "field_key": "participant_info.participant_name",
        "field_attribute": 1,
        "description": "\u5de5\u5355\u7684\u5f53\u524d\u5904\u7406\u4eba",
        "default_value": null,
        "order_id": 50,
        "field_name": "\u5f53\u524d\u5904\u7406\u4eba"
      }, {
        "field_value": "vpn\u7533\u8bf7",
        "label": {},
        "boolean_field_display": {},
        "field_type_id": 5,
        "field_template": "",
        "field_choice": {},
        "field_key": "workflow.workflow_name",
        "field_attribute": 1,
        "description": "\u5de5\u5355\u6240\u5c5e\u5de5\u4f5c\u6d41\u7684\u540d\u79f0",
        "default_value": null,
        "order_id": 60,
        "field_name": "\u5de5\u4f5c\u6d41\u540d\u79f0"
      }],
      "parent_ticket_state_id": 0,
      "add_node_man": "zhangsan",
      "participant": "1",
      "state_info": {
        "id": 10,
        "creator": "admin",
        "gmt_created": "2018-04-30 15:47:58",
        "gmt_modified": "2018-05-13 11:42:59",
        "is_deleted": false,
        "name": "\u4eba\u4e8b\u90e8\u95e8-\u5904\u7406\u4e2d",
        "workflow_id": 1,
        "is_hidden": false,
        "order_id": 4,
        "type_id": 0,
        "enable_retreat": false,
        "remember_last_man_enable": false,
        "participant_type_id": 1,
        "participant": "admin",
        "distribute_type_id": 1,
        "state_field_str": {
          "sn": 1,
          "title": 1,
          "leave_start": 1,
```


(下页继续)

(续上页)

```
            "leave_end": 1,
            "leave_days": 1,
            "leave_proxy": 1,
            "leave_type": 1,
            "creator": 1,
            "gmt_created": 1,
            "leave_reason": 1
          },
          "label": {}
        }
      }
    }
}
```

20.4 获取工单可以做的操作

- url

api/v1.0/tickets/{ticket_id}/transitions

- method

get

- 请求参数

无

- 返回数据

```
{
  "msg": "",
  "data": {
    "value": [
      {
        "transition_name": "提交",
        "field_require_check": true,  # 默认为 ture, 如果此为否时， 不校验表单必填内容
        "transition_id": 1,
        "is_accept": false, # 不是接单,
        "in_add_node": false, # 不处于加签状态下
        "enable_alert": false,  # 是否弹窗告警, 可用于当用户点击此操作的时确定是否弹窗信息
        "alert_text": "" # 弹窗中的消息内容
      },
      {
        "transition_name": "保存",
```

(下页继续)

(续上页)

```
        "field_require_check": true,  # 默认为 ture, 如果此为否时， 不校验表单必填内容
        "transition_id": 2,
        "is_accept": false, # 不是接单,
        "in_add_node": false, # 不处于加签状态下
        "enable_alert": false,  # 是否弹窗告警，可用于当用户点击此操作的时确定是否弹窗信息
        "alert_text": "" # 弹窗中的消息内容
      }
    ]
    },
  "code": 0
}
```

如果当前处理人超过一个人 (处理人类型为多人，部门、角色都有可能实际为多个人)，且当前状态的分配方式为主动接单，则会要求先接单, 返回数据如下。处理时需要处理人先接单 (点击接单按钮时调用接单接口).

```
{
  "msg": "",
  "code": 0,
  "data": {
    "value": [
      {
        "transition_id": 0,
        "transition_name": "接单",
        "is_accept": true,  # 接单,
        "in_add_node": false,
        "field_require_check": false
      }
    ]
  }
}
```

当工单当前处于加签状态下，返回格式如下。则用户点击“完成”按钮时，需要调用完成加签操作接口

```
{
  "msg": "",
  "code": 0,
  "data": {
    "value": [
      {
        "transition_id": 0,
        "transition_name": "完成",
        "is_accept": false,
        "in_add_node": true, # 处于加签状态
        "field_require_check": false
```

(下页继续)

(续上页)

```
        }
      ]
    }
}
```

20.5 接单

- url

api/v1.0/tickets/{ticket_id}/accept

- method

post

- 请求参数

无

- 使用场景

使用接口获取工单当前可以做的的操作后，如果 data.value.is_accept==true, 则需要用户先接单才能处理，即页面显示接单按钮，用户点击后调用接单接口，将工单的当前处理人设置该用户

- 返回数据

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

20.6 转交

- url

api/v1.0/tickets/{ticket_id}/deliver

- method

post

- 请求参数

参数名	类型	必填	说明
target_username	varchar	是	转交对象的用户名
suggestion	varchar	否	转交意见
from_admin	boolß	否	是否管理员强制转交，此参数用于对应工作流管理员或者超级管理员强制转交工单，传了 from_admin,loonflow 会校验用户是否是超级管理员或者该工作流的管理员

- 使用场景

在工单处理界面可以显示一个按钮“转交”，当用户认为当前工单自己处理不了时，可以将工单转交给合适的人处理。另外作为管理员可以强制 (即非工单当前处理人的情况下) 将工单转交给别人 ß

- 返回数据

```
{
  "data": true,
  "code": 0,
  "msg": ""
}
```

20.7 加签

- url

api/v1.0/tickets/{ticket_id}/add_node

- method

post

- 请求参数

参数名	类型	必填	说明
target_username	varchar	是	加签对象的用户名
suggestion	varchar	否	加签意见

- 使用场景

当用户 A 提交了一个权限申请工单，达到运维人员处理人中状态，作为运维人员的 B 在处理过程中发现需要 C 先处理或者提供一些必要的信息，B 才能处理。那么 B 在处理工单界面可以点击”加签“按钮，弹窗中选择 C。系统调用 loonflow 的加签接口将工单加签给 C。C 处理完后点击”完成“按钮，系统调用 loonflow 的加签完成接口，工单处理人将回到 B. 那么 B 就可以按照之前既定流程正常流转下去

- 返回数据

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

20.8 加签处理完成

- url

api/v1.0/tickets/{ticket_id}/add_node_end

- method

post

- 请求参数

参数名	类型	必填	说明
suggestion	varchar	否	加签完成意见

- 使用场景

使用场景当 A 将工单加签给 B.B 在处理工单时候，界面将只显示“完成“按钮，点击后后端调用此接口，将工单基础表中的 is_add_node 设置为 false

- 返回数据

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

20.9 处理工单

- url

api/v1.0/tickets/{ticket_id}

- method

patch

- 请求参数

参数名	类型	必填	说明
transition_id	int	是	流转 id
suggestion	varchar	否	处理意见（与处理工单类型，用户在处理工单的时候点击了按钮操作可以填写附加的一些意见如: 麻烦尽快处理）
其他必填字段	N/A	否	其他必填字段或可选字段（在配置工作流过程中, 会配置工作流每个状态的必填和可选的字段。在处理工单时候必须提供必填字段。如请假申请工单，配置了自定义字段请假天数 days，工单初始状态也设置了 days 为必填，那么处理此类工单时候就必选提供 days)。工单详情接口中有当前处理是时必选的字段

- 返回数据

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

20.10 获取工单流转记录

- url

api/v1.0/tickets/{ticket_id}/flowlogs

- method

get

- 请求参数

参数名	类型	必填	说明
ticket_data	int	否	是否返回每个操作时工单的所有字段信息，默认否

- 返回数据（ticket_data 未传或 ticket_data 传 0）

```
{
  "msg": "",
```

(下页继续)

(续上页)


```
"data": {
  "total": 4,
  "value": [
    {
      "state": {
        "state_name": "发起人-确认中",
        "state_id": 5
      },
      "transition": {
        "transition_name": "确认完成",
        "transition_id": 5,
        "attribute_type_id": 3
      },
      "ticket_id": 1,
      "participant_info": {
        "participant_email": "lilei@163.com",
        "participant_alias": "李磊",
        "participant_phone": "13888888888",
        "participant": "lilei",
        "participant_type_id": 1
      },
      "gmt_modified": "2018-04-30 15:57:26",
      "gmt_created": "2018-04-30 15:56:02",
      "suggestion": "已经生效，感谢"
    },
    {
    "state": {
      "state_name": "技术人员-处理中",
      "state_id": 4
      },
    "transition": {
      "transition_name": "处理完成",
      "transition_id": 4
    },
    "ticket_id": 1,
    "participant_info": {
        "participant_email": "lilei@163.com",
        "participant_alias": "李磊",
        "participant_phone": "13888888888",
        "participant": "lilei",
        "participant_type_id": 1
      },
    "gmt_modified": "2018-04-30 15:57:14",
```


(下页继续)

(续上页)


```
      "gmt_created": "2018-04-30 15:55:32",
      "suggestion": "处理完成"
      },
      {
      "state": {
        "state_name": "TL 审批中",
        "state_id": 3
      },
      "transition": {
        "transition_name": "同意",
        "transition_id": 3
      },
      "ticket_id": 1,
      "participant_info": {
          "participant_email": "lilei@163.com",
          "participant_alias": "李磊",
          "participant_phone": "13888888888",
          "participant": "lilei",
          "participant_type_id": 1
        },
      "gmt_modified": "2018-04-30 15:57:00",
      "gmt_created": "2018-04-30 15:53:19",
      "suggestion": "同意处理"
      },
      {
      "state": {
        "state_name": "新建中",
        "state_id": 1
      },
      "transition": {
        "transition_name": "提交",
        "transition_id": 1
      },
      "ticket_id": 1,
      "gmt_modified": "2018-04-30 15:52:35",
      "gmt_created": "2018-04-10 17:39:33",
      "suggestion": "请尽快处理，谢谢"
      }],
    "page": 1,
    "per_page": 10
    },
  "code": 0
}
```

- 返回数据（ticket_data 传 1）


```
{
        "msg": "",
        "data": {
                "total": 4,
                "value": [{
                                "state": {
                                        "state_name": "发起人-确认中",
                                        "state_id": 5
                                },
                                "transition": {
                                        "transition_name": "确认完成",
                                        "transition_id": 5,
                                        "attribute_type_id": 3
                                },
                                "ticket_id": 1,
                                "participant_info": {
                                        "participant_email": "lilei@163.com",
                                        "participant_alias": "李磊",
                                        "participant_phone": "13888888888",
                                        "participant": "lilei",
                                        "participant_type_id": 1
                                },
                                "gmt_modified": "2018-04-30 15:57:26",
                                "gmt_created": "2018-04-30 15:56:02",
                                "suggestion": "已经生效，感谢",
                                "ticket_data": {
                                        "title": "xxx",
                                        "sn": "xxxxx",
                                        "state_id": 1,
                                        "ticket_id": 1,
                                        "gmt_modified": "2018-04-30 15:57:26",
                                        "gmt_created": "2018-04-30 15:56:02",
                                        "xxxx": "....."
                                }
                        },
                        {
                                "state": {
                                        "state_name": "技术人员-处理中",
                                        "state_id": 4
                                },
                                "transition": {
                                        "transition_name": "处理完成",
                                        "transition_id": 4
```


(下页继续)

(续上页)


```
                        },
                        "ticket_id": 1,
                        "participant_info": {
                                "participant_email": "lilei@163.com",
                                "participant_alias": "李磊",
                                "participant_phone": "13888888888",
                                "participant": "lilei",
                                "participant_type_id": 1
                        },
                        "gmt_modified": "2018-04-30 15:57:14",
                        "gmt_created": "2018-04-30 15:55:32",
                        "suggestion": "处理完成",
                        "ticket_data": {
                                "title": "xxx",
                                "sn": "xxxxx",
                                "state_id": 1,
                                "ticket_id": 1,
                                "gmt_modified": "2018-04-30 15:57:26",
                                "gmt_created": "2018-04-30 15:56:02",
                                "xxxx": "....."
                        }
                },
                {
                        "state": {
                                "state_name": "TL 审批中",
                                "state_id": 3
                        },
                        "transition": {
                                "transition_name": "同意",
                                "transition_id": 3
                        },
                        "ticket_id": 1,
                        "participant_info": {
                                "participant_email": "lilei@163.com",
                                "participant_alias": "李磊",
                                "participant_phone": "13888888888",
                                "participant": "lilei",
                                "participant_type_id": 1
                        },
                        "gmt_modified": "2018-04-30 15:57:00",
                        "gmt_created": "2018-04-30 15:53:19",
                        "suggestion": "同意处理",
                        "ticket_data": {
```


(下页继续)

(续上页)


```
                                "title": "xxx",
                                "sn": "xxxxx",
                                "state_id": 1,
                                "ticket_id": 1,
                                "gmt_modified": "2018-04-30 15:57:26",
                                "gmt_created": "2018-04-30 15:56:02",
                                "xxxx": "....."
                        }
                },
                {
                        "state": {
                                "state_name": "新建中",
                                "state_id": 1
                        },
                        "transition": {
                                "transition_name": "提交",
                                "transition_id": 1
                        },
                        "ticket_id": 1,
                        "gmt_modified": "2018-04-30 15:52:35",
                        "gmt_created": "2018-04-10 17:39:33",
                        "suggestion": "请尽快处理，谢谢",
                        "ticket_data": {
                                "title": "xxx",
                                "sn": "xxxxx",
                                "state_id": 1,
                                "ticket_id": 1,
                                "gmt_modified": "2018-04-30 15:57:26",
                                "gmt_created": "2018-04-30 15:56:02",
                                "xxxx": "....."
                        }
                }
        ],
        "page": 1,
        "per_page": 10
    },
    "code": 0
```


}

20.11 工单处理步骤记录

- url

api/v1.0/tickets/{ticket_id}/flowsteps

- method

get

- 请求参数

无

- 返回数据

```
{
  "data": {
    "current_state_id": 2  //工单当前状态 id
    "value": [{
      "state_id": 17,
      "state_flow_log_list": [],
      "order_id": 0,
      "state_name": "test11111"
    }, {
      "state_id": 18,
      "state_flow_log_list": [],
      "order_id": 0,
      "state_name": "2233222"
    }, {
      "state_id": 6,
      "state_flow_log_list": [{
        "gmt_created": "2018-05-15 07:16:38",
        "participant_info": {
          "participant_alias": "李磊",
          "participant_type_id": 1,
          "participant": "lilei",
          "participant_phone": "13888888888",
          "participant_email": "lilei@163.com"
        },
        "suggestion": "",
        "participant": "lilei",
        "state_id": 6,
        "participant_type_id": 1,
        "transition": {
          "transition_name": "提交",
          "transition_id": 7
```

(下页继续)

(续上页)


```
        },
        "id": 32,
        "intervene_type_id": 0
      }],
      "order_id": 1,
      "state_name": "发起人-新建中"
    }, {
      "state_id": 7,
      "state_flow_log_list": [{
        "gmt_created": "2018-05-15 07:20:40",
        "participant_info": {
          "participant_alias": "李磊",
          "participant_type_id": 1,
          "participant": "lilei",
          "participant_phone": "13888888888",
          "participant_email": "lilei@163.com"
        },
        "suggestion": "同意申请",
        "participant": "lilei",
        "state_id": 7,
        "participant_type_id": 1,
        "transition": {
          "transition_name": "同意",
          "transition_id": 8
        },
        "id": 33,
        "intervene_type_id": 0
      }],
      "order_id": 2,
      "state_name": "发起人 tl-审批中"
    }, {
      "state_id": 8,
      "state_flow_log_list": [{
        "gmt_created": "2018-05-16 06:42:00",
        "participant_info": {
          "participant_alias": "轨迹",
          "participant_type_id": 1,
          "participant": "guiji",
          "participant_phone": "13888888888",
          "participant_email": "guiji@163.com"
        },
        "suggestion": "接单处理",
        "participant": "guiji",
```


(下页继续)

(续上页)


```
        "state_id": 8,
        "participant_type_id": 1,
        "transition": {
          "transition_name": "未知操作",
          "transition_id": 0
        },
        "id": 36,
        "intervene_type_id": 0
      }, {
        "gmt_created": "2018-05-16 06:49:55",
        "participant_info": {
          "participant_alias": "轨迹",
          "participant_type_id": 1,
          "participant": "guiji",
          "participant_phone": "13888888888",
          "participant_email": "guiji@163.com"
        },
        "suggestion": "同意",
        "participant": "guiji",
        "state_id": 8,
        "participant_type_id": 1,
        "transition": {
          "transition_name": "同意",
          "transition_id": 9
        },
        "id": 37,
        "intervene_type_id": 0
      }, {
        "gmt_created": "2018-05-16 06:57:31",
        "participant_info": {
          "participant_alias": "轨迹",
          "participant_type_id": 1,
          "participant": "guiji",
          "participant_phone": "13888888888",
          "participant_email": "guiji@163.com"
        },
        "suggestion": "接单处理",
        "participant": "guiji",
        "state_id": 8,
        "participant_type_id": 1,
        "transition": {
          "transition_name": "未知操作",
          "transition_id": 0
```


(下页继续)

(续上页)


```
        },
        "id": 38,
        "intervene_type_id": 0
      }, {
        "gmt_created": "2018-05-16 06:57:36",
        "participant_info": {
          "participant_alias": "轨迹",
          "participant_type_id": 1,
          "participant": "guiji",
          "participant_phone": "13888888888",
          "participant_email": "guiji@163.com"
        },
        "suggestion": "同意",
        "participant": "guiji",
        "state_id": 8,
        "participant_type_id": 1,
        "transition": {
          "transition_name": "同意",
          "transition_id": 9
        },
        "id": 39,
        "intervene_type_id": 0
      }, {
        "gmt_created": "2018-05-16 06:58:41",
        "participant_info": {
          "participant_alias": "轨迹",
          "participant_type_id": 1,
          "participant": "guiji",
          "participant_phone": "13888888888",
          "participant_email": "guiji@163.com"
        },
        "suggestion": "同意",
        "participant": "guiji",
        "state_id": 8,
        "participant_type_id": 1,
        "transition": {
          "transition_name": "同意",
          "transition_id": 9
        },
        "id": 40,
        "intervene_type_id": 0
      }, {
        "gmt_created": "2018-05-16 07:01:53",
```


(下页继续)

(续上页)


```
        "participant_info": {
          "participant_alias": "轨迹",
          "participant_type_id": 1,
          "participant": "guiji",
          "participant_phone": "13888888888",
          "participant_email": "guiji@163.com"
        },
        "suggestion": "同意",
        "participant": "guiji",
        "state_id": 8,
        "participant_type_id": 1,
        "transition": {
          "transition_name": "同意",
          "transition_id": 9
        },
        "id": 41,
        "intervene_type_id": 0
      }, {
        "gmt_created": "2018-05-16 07:03:34",
        "participant_info": {
          "participant_alias": "轨迹",
          "participant_type_id": 1,
          "participant": "guiji",
          "participant_phone": "13888888888",
          "participant_email": "guiji@163.com"
        },
        "suggestion": "同意",
        "participant": "guiji",
        "state_id": 8,
        "participant_type_id": 1,
        "transition": {
          "transition_name": "同意",
          "transition_id": 9
        },
        "id": 43,
        "intervene_type_id": 0
      }, {
        "gmt_created": "2018-05-16 07:04:45",
        "participant_info": {
          "participant_alias": "轨迹",
          "participant_type_id": 1,
          "participant": "guiji",
          "participant_phone": "13888888888",
```


(下页继续)

(续上页)


```
            "participant_email": "guiji@163.com"
        },
        "suggestion": "同意",
        "participant": "guiji",
        "state_id": 8,
        "participant_type_id": 1,
        "transition": {
            "transition_name": "同意",
            "transition_id": 9
        },
        "id": 45,
        "intervene_type_id": 0
    }, {
        "gmt_created": "2018-05-16 07:31:29",
        "participant_info": {
            "participant_alias": "轨迹",
            "participant_type_id": 1,
            "participant": "guiji",
            "participant_phone": "13888888888",
            "participant_email": "guiji@163.com"
        },
        "suggestion": "同意",
        "participant": "guiji",
        "state_id": 8,
        "participant_type_id": 1,
        "transition": {
            "transition_name": "同意",
            "transition_id": 9
        },
        "id": 47,
        "intervene_type_id": 0
    }, {
        "gmt_created": "2018-05-16 23:21:00",
        "participant_info": {
            "participant_alias": "轨迹",
            "participant_type_id": 1,
            "participant": "guiji",
            "participant_phone": "13888888888",
            "participant_email": "guiji@163.com"
        },
        "suggestion": "同意",
        "participant": "guiji",
        "state_id": 8,
```


(下页继续)

(续上页)


```
        "participant_type_id": 1,
        "transition": {
          "transition_name": "同意",
          "transition_id": 9
        },
        "id": 49,
        "intervene_type_id": 0
      }, {
        "gmt_created": "2018-05-16 23:24:03",
        "participant_info": {
          "participant_alias": "轨迹",
          "participant_type_id": 1,
          "participant": "guiji",
          "participant_phone": "13888888888",
          "participant_email": "guiji@163.com"
        },
        "suggestion": "同意",
        "participant": "guiji",
        "state_id": 8,
        "participant_type_id": 1,
        "transition": {
          "transition_name": "同意",
          "transition_id": 9
        },
        "id": 51,
        "intervene_type_id": 0
      }, {
        "gmt_created": "2018-05-16 23:24:44",
        "participant_info": {
          "participant_alias": "轨迹",
          "participant_type_id": 1,
          "participant": "guiji",
          "participant_phone": "13888888888",
          "participant_email": "guiji@163.com"
        },
        "suggestion": "同意",
        "participant": "guiji",
        "state_id": 8,
        "participant_type_id": 1,
        "transition": {
          "transition_name": "同意",
          "transition_id": 9
        },
```


(下页继续)

(续上页)


```
        "id": 53,
        "intervene_type_id": 0
      }, {
        "gmt_created": "2018-05-16 23:33:26",
        "participant_info": {
          "participant_alias": "轨迹",
          "participant_type_id": 1,
          "participant": "guiji",
          "participant_phone": "13888888888",
          "participant_email": "guiji@163.com"
        },
        "suggestion": "同意",
        "participant": "guiji",
        "state_id": 8,
        "participant_type_id": 1,
        "transition": {
          "transition_name": "同意",
          "transition_id": 9
        },
        "id": 55,
        "intervene_type_id": 0
      }],
      "order_id": 3,
      "state_name": "运维人员-审批中"
    }, {
      "state_id": 9,
      "state_flow_log_list": [{
        "gmt_created": "2018-05-16 07:01:54",
        "participant_info": {
          "participant_phone": "",
          "participant_alias": "demo_script.py",
          "participant_email": "",
          "participant_type_id": 6,
          "participant": "demo_script.py"
        },
        "suggestion": "False\n",
        "participant": "demo_script.py",
        "state_id": 9,
        "participant_type_id": 6,
        "transition": {
          "transition_name": "脚本执行完成",
          "transition_id": 10
        },
```


(下页继续)

(续上页)


```
        "id": 42,
        "intervene_type_id": 0
      }, {
        "gmt_created": "2018-05-16 07:03:34",
        "participant_info": {
          "participant_phone": "",
          "participant_alias": "demo_script.py",
          "participant_email": "",
          "participant_type_id": 6,
          "participant": "demo_script.py"
        },
        "suggestion": "False\n",
        "participant": "demo_script.py",
        "state_id": 9,
        "participant_type_id": 6,
        "transition": {
          "transition_name": "脚本执行完成",
          "transition_id": 10
        },
        "id": 44,
        "intervene_type_id": 0
      }, {
        "gmt_created": "2018-05-16 07:04:45",
        "participant_info": {
          "participant_phone": "",
          "participant_alias": "demo_script.py",
          "participant_email": "",
          "participant_type_id": 6,
          "participant": "demo_script.py"
        },
        "suggestion": "False\n",
        "participant": "demo_script.py",
        "state_id": 9,
        "participant_type_id": 6,
        "transition": {
          "transition_name": "脚本执行完成",
          "transition_id": 10
        },
        "id": 46,
        "intervene_type_id": 0
      }, {
        "gmt_created": "2018-05-16 07:31:29",
        "participant_info": {
```


(下页继续)

(续上页)


```
            "participant_phone": "",
            "participant_alias": "demo_script.py",
            "participant_email": "",
            "participant_type_id": 6,
            "participant": "demo_script.py"
        },
        "suggestion": "lilei\n",
        "participant": "demo_script.py",
        "state_id": 9,
        "participant_type_id": 6,
        "transition": {
            "transition_name": "脚本执行完成",
            "transition_id": 10
        },
        "id": 48,
        "intervene_type_id": 0
    }, {
        "gmt_created": "2018-05-16 23:21:00",
        "participant_info": {
            "participant_phone": "",
            "participant_alias": "demo_script.py",
            "participant_email": "",
            "participant_type_id": 6,
            "participant": "demo_script.py"
        },
        "suggestion": "lilei\n",
        "participant": "demo_script.py",
        "state_id": 9,
        "participant_type_id": 6,
        "transition": {
            "transition_name": "脚本执行完成",
            "transition_id": 10
        },
        "id": 50,
        "intervene_type_id": 0
    }, {
        "gmt_created": "2018-05-16 23:24:03",
        "participant_info": {
            "participant_phone": "",
            "participant_alias": "demo_script.py",
            "participant_email": "",
            "participant_type_id": 6,
            "participant": "demo_script.py"
```


(下页继续)

(续上页)


```
        },
        "suggestion": "lilei\n",
        "participant": "demo_script.py",
        "state_id": 9,
        "participant_type_id": 6,
        "transition": {
          "transition_name": "脚本执行完成",
          "transition_id": 10
        },
        "id": 52,
        "intervene_type_id": 0
      }, {
        "gmt_created": "2018-05-16 23:24:44",
        "participant_info": {
          "participant_phone": "",
          "participant_alias": "demo_script.py",
          "participant_email": "",
          "participant_type_id": 6,
          "participant": "demo_script.py"
        },
        "suggestion": "lilei\n",
        "participant": "demo_script.py",
        "state_id": 9,
        "participant_type_id": 6,
        "transition": {
          "transition_name": "脚本执行完成",
          "transition_id": 10
        },
        "id": 54,
        "intervene_type_id": 0
      }, {
        "gmt_created": "2018-05-16 23:33:26",
        "participant_info": {
          "participant_phone": "",
          "participant_alias": "demo_script.py",
          "participant_email": "",
          "participant_type_id": 6,
          "participant": "demo_script.py"
        },
        "suggestion": "lilei\n",
        "participant": "demo_script.py",
        "state_id": 9,
        "participant_type_id": 6,
```


(下页继续)

(续上页)


```
        "transition": {
          "transition_name": "脚本执行完成",
          "transition_id": 10
        },
        "id": 56,
        "intervene_type_id": 0
      }],
      "order_id": 4,
      "state_name": "授权脚本-自动执行中"
    }, {
      "state_id": 10,
      "state_flow_log_list": [{
        "gmt_created": "2018-05-17 06:45:58",
        "participant_info": {
          "participant_alias": "李磊",
          "participant_type_id": 1,
          "participant": "lilei",
          "participant_phone": "13888888888",
          "participant_email": "lilei@163.com"
        },
        "suggestion": "请处理",
        "participant": "lilei",
        "state_id": 10,
        "participant_type_id": 1,
        "transition": {
          "transition_name": "转交操作",
          "transition_id": 0
        },
        "id": 57,
        "intervene_type_id": 1
      }, {
        "gmt_created": "2018-05-17 06:47:46",
        "participant_info": {
          "participant_alias": "张三",
          "participant_type_id": 1,
          "participant": "zhangsan",
          "participant_phone": "13888888888",
          "participant_email": "zhangsan@163.com"
        },
        "suggestion": "请协助处理",
        "participant": "zhangsan",
        "state_id": 10,
        "participant_type_id": 1,
```


(下页继续)

(续上页)


```
        "transition": {
          "transition_name": "加签操作",
          "transition_id": 0
        },
        "id": 58,
        "intervene_type_id": 2
      }],
      "order_id": 6,
      "state_name": "发起人-确认中"
    }, {
      "state_id": 11,
      "state_flow_log_list": [],
      "order_id": 7,
      "state_name": "结束"
    }]
  },
  "msg": "",
  "code": 0
}
```


20.12 修改工单状态

- url

api/v1.0/tickets/{ticket_id}/state

- method

put

- 请求参数

参数名	类型	必填	说明
state_id	int	是	目标状态 id
suggestion	varchar	否	处理意见

- 使用场景

用于干预工单的当前状态, 可以直接将工单状态修改为指定状态，系统会根据 state_id 获取对应的处理人信息

- 返回格式


```
{
  "msg": "",
```


(下页继续)

(续上页)

```
  "data": {},
  "code": 0
}
```

20.13 批量获取工单状态

- url

api/v1.0/tickets/states

- method

get

- 请求参数

参数名	类型	必填	说明
ticket_ids	varchar	是	工单 ids, 逗号隔开的字符串

- 使用场景

调用方自己保存工单基础信息并根据 loonflow 中工单 id 关联，在显示工单列表时直接从自己后端获取工单列表。但是工单状态需要实时从 loonflow 中获取，那么可以通过此接口获取一页工单列表每个工单的状态

- 返回数据

```
{
  "code": 0,
  "data": {
    "1": {
        "state_id": 1,
        "state_name": "发起人-编辑中"
      },
    2: {
        "state_id": 2,
        "state_name": "新建中"
      }
  },
  "msg": ""
}
```

20.14 修改工单字段的值

- url

api/v1.0/tickets/{ticket_id}/fields

- method

patch

- 请求参数

参数名	类型	必填	说明
需要修改值的字段的 key1	varchar	是	如需要修改标题，则就是 title
需要修改值的字段的 key2	varchar	是	如需要修改标题，则就是 title
其他需要修改的字段的字段标识	varchar	是	如需要修改标题，则就是 title

- 返回数据

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

20.15 重试工单脚本/hook 任务

- url

api/v1.0/tickets/{ticket_id}/retry_script

- method

post

- 请求参数

无

- 使用场景

当工单的脚本 (或者 hook[v0.3.17 版本支持]) 执行失败后，工单详情接口中获取的数据中 script_run_last_result 为 false. 这时可以在工单详情界面 step 图中此状态下显示有个” 重试按钮 “，用户点击此按钮后，可以调用此接口重新执行或重新触发 hook

- 返回数据

```
{
  "msg": "Ticket script or hook retry start successful",
  "data": {},
  "code": 0
}
```

20.16 新增工单评论/注释

- url

api/v1.0/tickets/{ticket_id}/comments

- method

post

- 请求参数

参数名	类型	必填	说明
suggestion	varchar	是	处理意见（与处理工单类型，用户在处理工单的时候点击了按钮操作可以填写附加的一些意见如: 麻烦尽快处理）

- 返回数据

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

20.17 工单 hook 回调

- url

api/v1.0/tickets/{ticket_id}/hook_call_back

- method

post

- 请求参数

参数名	类型	必填	说明
result	boolean	是	hook 任务执行是否成功，false, true
msg	varchar	是	hook 执行输出信息, 可留空’’
field_value	dict object	否	需要修改值的字段. 这些字段需要在状态表单设置中为可选或者必填

- 使用场景

当工作流状态设置处理人类型为 hook，工单到达此状态时，会触发 hook 请求，被请求方可以执行一些操作，执行完成后回调用 loonflow, 告知 loonflow 任务执行结果，以触发 loonflow 中工单状态的流转 (当 hook 配置中 wait 为 false 时，无需回调，hook 发出后会立即触发流转，wait 为 true 会等待回调)。回调参数如果 result 为 false, 那么 loonflow 会标记该工单的 script_run_last_result 为 False(获取工单详情接口也会返回此标识，前端可以根据这个标识来显示一个重试的按钮，用户点击这个重试按钮后调用” 重试工单脚本/hook 任务” 接口)，同时也会将 msg(你可以传失败的原因) 中的内容记录到工单流转记录中。

- 返回数据

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

20.18 工单当前的参与人详情

- url

api/v1.0/tickets/{ticket_id}/participant_info

- method

get

- 使用场景

此接口将返回该工单当前的参与人详细信息，如果是部门或角色会返回对应部门角色下所有用户。调用方可基于此提供工单催办的功能。用户在前端点击催办按钮，前端弹窗要求用户选择通知的类型：短信、邮件、微信、钉钉等等以及需要的备注信息，然后调用方后端发送相应的通知消息给工单的当前处理人

- 返回数据

```
{
  "msg": "",
  "data": {
    "participant_info_list": [{
      "alias": "\u8d85\u7ea7\u7ba1\u7406\u5458",
```

(下页继续)

(续上页)


```
      "username": "admin",
      "phone": "13888888888",
      "email": "blackholll@163.com"
    }, {
      "alias": "\u8f68\u8ff9",
      "username": "guiji",
      "phone": "13888888888",
      "email": "guiji@163.com"
    }, {
      "alias": "\u674e\u78ca",
      "username": "lilei",
      "phone": "13888888888",
      "email": "lilei@163.com"
    }, {
      "alias": "\u5f20\u4e09",
      "username": "zhangsan",
      "phone": "13888888888",
      "email": "zhangsan@163.com"
    }, {
      "alias": "\u674e\u56db",
      "username": "lisi",
      "phone": "13888888888",
      "email": "lisi@163.com"
    }, {
      "alias": "\u738b\u4e94",
      "username": "wangwu",
      "phone": "13888888888",
      "email": "wangwu@163.com"
    }, {
      "alias": "\u6770\u514b",
      "username": "jack",
      "phone": "13888888888",
      "email": "jack@163.com"
    }],
    "participant_username_list": ["admin", "guiji", "lilei", "zhangsan", "lisi",
↪"wangwu", "jack"]
  },
  "code": 0
}
```

20.19 强制关闭工单

- url

api/v1.0/tickets/{ticket_id}/close

- method

post

- 请求参数

参数名	类型	必填	说明
suggestion	varchar	否	关闭原因

- 使用场景

超级管理员在查看工单详情时，可以在界面上显示一个强制关闭工单的按钮，点击后调用关闭工单按钮，实现强制关闭工单。另外工单创建人在工单处于初始状态下 (创建人撤回、退回到初始状态等情况工单状态会处于初始状态) 也可以强制关闭工单。

- 返回数据

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

20.20 撤回工单

- url

api/v1.0/tickets/{ticket_id}/retreat

- method

post

- 请求参数

参数名	类型	必填	说明
suggestion	varchar	否	撤回原因

- 使用场景

在配置工作流状态时，可以指定某些状态下允许创建人撤回工单，那么当工单处于这些状态时，创建人可以撤回该工单 (调用方前端在这个情况下显示一个撤回按钮)

- 返回数据

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

CHAPTER 21

相关常量

21.1 状态类型

1: 开始状态. 工单的最初始状态，如发起人新建中

2: 结束状态. 工单的最终状态，如完成、结束、关闭等等

21.2 分配方式

1: 主动接单。工单到达时如果当前处理人是多人，需要用户先接单再处理 (避免多人同时处理。场景: 开发人员提交了一个定制化的机器的申请，在运维人员处理中这个状态，此状态下配置的处理人是整个运维部门，那么所有运维都会看到这个工单，其中一个运维人员点击接单后代表其将为其服务。这时候其他人将在工单详情中看到处理人已经是这运维人员)

2: 直接处理。工单到达时如果当前处理人是多人，不需要先接单，谁都可以处理

3: 随机分配。工单到达时候，如果处理人为多人，那么系统将随机分配给某个人。如上面这个例子，系统将直接给工单的 ◆ 当前处理人设置为随机的一名运维人员

4: 全部处理。当设置成某个状态为全部处理时，工单在此状态下需要所有相关人员都处理完成后，才会进入到下个状态

21.3 处理人类型

1: 个人

2: 多人

3: 部门

4: 角色

5: 变量如工单创建人、工单创建人 leader

6: 脚本/机器人执行脚本的情况

7: 工单字段工单的某个字段 (需要是用户名或者是逗号隔开的用户名), 如工单的某个自定义字段是测试人员’devs’, 工单流转过程中其中一个状态是测试人员测试中，那么那个状态的处理人类型可以为 7, 处理人为’devs’)

8: 父工单字段父工单的某个字段 (需要是用户名或者是逗号隔开的用户名), 如上述项目和应用周期的工单，应用工单在某个状态下需要项目的负责人’po’审批，那么该状态的处理人类型可以为 8，处理人为’po’

9: 多人全部处理 (处理人为多个，且每个人都需要处理)，当状态处理人配置为全部处理，且处理人数大于 1 时，实际的处理人类型则为此

10. hook 方式，当工单状态叨叨处理人类型配置为 kook 的状态时，loonflow 将触发一个 hook 请求，被请求方可以执行有些自动化操作然后回调 loonflow

21.4 流转类型

1: 常规流转

2: 定时器流转

21.5 自定义字段类型

5: 字符串

10: 整形

15: 浮点型

20: 布尔类型

25: 日期类型

30: 日期时间类型

35: 单选框 radio

40: 多选框 checkbox

45: 下拉列表

50: 多选的下拉列表

55: 文本域

60: 用户名 (需要调用方系统自行处理用户列表，loonflow 只保存用户名)

70: 多选用户名 (需要调用方系统自行处理用户列表，loonflow 只保存用户名，多人的情况使用逗号隔开)

80: 附件，多个附件使用逗号隔开。调用方自己实现上传功能，loonflow 只保存文件路径

21.6 字段属性

1: 只读调用新建或处理工单的接口时如果传了设置为只读的字段的值，loonflow 将忽略，不会更新工单此字段的值

2: 必填调用新建或处理工单的接口时必须传递此字段的值，如果未提供则新建或处理工单接口将调用失败

3: 可选调用新建或处理工单的接口时可传可不传此字段的值，如果传了此类型的字段，则 loonflow 将更新工单此字段的值

21.7 工单权限类别

1: 用户当前拥有此工单的处理权限 (因为随着工单的状态变化，权限也会相应变化)

2: 用户当前拥有此工单的查看权限 (因为随着工单的状态变化，权限也会相应变化)

CHAPTER 22

常见问题

- 为什么不使用外键

关于是否使用数据库外键，网上各有说法，各有利弊。本人基于如下几个原因所以没使用外键:1. 外键属于业务逻辑，不应该吧这种逻辑放到数据库层而加重数据库的负担 2. 使用外键会有数据的强校验，导致日常维护时会有些麻烦 (后来发现其实可以使用 db_constraint=False, on_delete=False 参数来关闭强校验, 通知也保留外键本来的便利性, 1.0 版本也有开始少量应用) 3. 有些公司 db 规范里面就是不允许使用外键的

- 为什么没使用 django rest framework

drf 在未使用外键的情况下, 无法发挥其价值。无显式外键时，自定义序列话返回数据需要逐条记录查询关联信息，效率非常低

- 为什么使用 http api 方式提供服务而不是 python 三方包

1.loonflow 的理念是: 工单应该是嵌入到各个系统中 (如 oa,cmdb, 运维平台、客服系统等等), 这些系统通过后端 api 调用 loonflow。所以 loonflow 只有管理界面 (v0.1 版本直接使用 django admin, 后面会重写管理界面)。后续会提供几个调用方 demo 供大家参考。感谢 @youshutong 帮忙写的调用方 demo(vue+django): https://github.com/youshutong2080/shutongFlow 另外帮忙 jimmy201602 写的 demo(bootstrap+django): https://github.com/jimmy201602/workflowdemo 2.loonflow 引擎功能较复杂功能较复杂不适合使用包的方式

- 为何不建议调用方前端直接调用 loonflow

调用方和 loonflow 之前需要做权限验证，签名算法考虑到安全只能写在调用方后端; 作为引擎,loonflow 不提供用户登录验证功能, 只校验调用方的合法性，所以登录验证需要做在调用方自己的后端; 每个调用方除了纯粹的工单的功能，还会需要一些额外的功能，比如根据自定义字段筛选工单列表，loonflow 提供了工单列表的接，但是因为 loonflow 的自定义字段是纵表形式存储的，无法提供根据这些字段来筛选工单列表。如果需要自定义字段的筛选，需要调用方自己保存一份工单数据，用于筛选; 比如需要做一个项目全生命周期管理的

系统，需要用到工作流。但是还有比如发布, 获取人员信息、和其他系统交互、日志查看、项目数据统计等等功能。这些需要做在自己的后端

- 调用方是否需要保存工单的基础数据

根据情况而定，如果调用方在显示工单数据的时候需要显示更多相关信息，可以本地保存一份附属信息与 loonflow 中对应关系。针对本地保存的情况，如果涉及工单流转的字段 (如参与人等)，在本地修改时需要同时调用 loonflow 修改 loonflow 中保存的字段的值 (v0.2 版本会提供修改工单字段值的接口)

- 如何限制用户查看工单权限

默认会限制工单的查看权限 (通过 api 获取工单详情时, 只有 username 参数是工单相关人员时才能获取到数据)。如果需要放开限制，可以修改工作流配置中的“查看权限校验”为否。权限配置只针对工作流的，多个类型的工作流需要单独配置

- 为何需要同步用户及部门信息到 loonflow

因为工单流转涉及到较多的用户信息获取，所以需要将用户信息 (包括部门) 同步到 loonflow 的账户系统中。同步部门信息的时候，如果发现部门被删除，建议修改部门名字，如前面加个“已废弃:”, 否则如果该部门存在某个工单的当前处理人的时候会有问题。用户离职的情况设置 is_active=0. 另外用户密码请随便填写 (为了不允许普通用户登录)。管理员账户请通过 python manage.py creatsuperuser 来创建。只需要管理员实现一个同步脚本定时执行即可, 其他调用方不用考虑此问题

- 为什么调用方 demo 中要保存用户信息

调用方 demo 可以理解为你的 cmdb、客服系统、运维系统、oa 系统。这些系统你可以自己管理用户信息，也可以在涉及用户方面的地方都调用接口来获取。不同公司提供用户信息的方式不同，所以 demo 里面自己实现了个简单的用户管理供大家参考。

- 为什么调用方 demo 中的用户，也需要存在于 loonflow 的用户中

loonflow 的工单在流转过程中，如目标状态的处理人是创建人 TL，那么 loonflow 需要在自己的用户体系中找到这个用户所在部门，然后获取这个部门的审批人或者 tl 信息

- 如何支持根据工单的自定义字段查询

loonflow 只提供工单基础字段的查询，如果需要针对自定义字段的查询，请在自己系统中保存一份工单数据 (注意工单处理过程中，如果有字段修改，也需要更新自己系统中的数据)

- 工单列表支持排序

只支持根据创建时间排序。其他字段排序可以在调用方系统中保存一份数据来自己实现排序，然后只有在获取工单详情的时候调用 loonflow 接口

- 工单类型需要支持多级

比如需要支持“运维-权限申请-vpn 权限申请”。因为 loonflow 的工作流只有一级，如果需要支持多级类型, 需要在调用方保存一份工单类型与 loonflow 工作流关联的数据。表字段可以如下：type_id, type_name, up_type_id, loonflow_workflow_id

- 如何实现工单的评分和处理优先级功能

因为不同公司对于评分的需求不同，如评分有 1 星、2 星、3 星，满意、及格、不合格。如优先级有高、中、低，紧急、中等、不急等等。因此 loonflow 不提供通用功能。用户可以针对不同的工作流定义不同的自定义字段以表示评分或者优先级，自定义字段可以选择 checkbox 类型，也可以通过字段的标签灵活处理 (前端根据约定好的标签，特殊显示)。当然如果你这边的需求非常统一，你可以给 loonflow 的基础表中添加一个字段，以实现公用。不过修改此逻辑后，后续 loonflow 更新时需要特别注意下

- 为什么为提供拖拽生成工作流的功能

loonflow 目前非商业化系统，面向用户是有一定计算机基础的人。完全拖拽只能实现非常基础的工作流的配置，复杂点的还是需要填写一些个性化的如标签来实现定制功能。涉及定制功能的工作流配置使用当前方式效率不比拖拽低。因此拖拽生成工作流的功能优先级较低，会等其他功能都比较完善后再考虑支持

- 为什么不用 migration 维护表结构

django 提供的 migration 功能建议只用于自己的开发环境。生产环境直接操作执行风险较大，在实际生产环境中涉及数据库表结构的变更一般需要 dba 审核，这种审核是基于 DDL 语句来审核的（没理由要求 DBA 查看你的 django modal）

有问题请优先查文档及在 github 上提 issue，再考虑到群里 (qq 群:558788490) 咨询, 群内问题我会每天晚上 21:00-第二天 8 点之间答复 (可能会遗漏)。github 上 issue 不会遗漏。常见问题我会定期整理添加到此文档中

CHAPTER 23

欢迎捐助

您的支持是我最大的动力, 欢迎支付宝扫码捐助, 捐助满 300 元。即可享受 VIP 服务, 权益包括:

1. 加微信好友，微信回复更及时
2. 支持微信语音问题解答
3. 提出的合理通用新需求，优先支持
4. 后续会推出视频讲解课程，免费获取课程

CHAPTER 24

发布说明

24.1 r1.0.4

- 修复: 工单自定义字段的值不能被正常更新问题
- 修复: 处理人为多部门时, 处理人计算错误问题
- 修复: 撤回工单未更新工单状态问题

24.2 r1.0.3

- 修复: 强制修改工单状态后处理人异常问题
- 修复: 撤回工单条件判断逻辑错误问题
- 新增: 新增 docker compose 方式部署 loonflow_shutongflow(仅供演示用)

24.3 r1.0.2

- 修复: 获取工作流状态详情接口报错问题
- 修复: 还没有配置工作流时，工单管理界面报错问题
- 修复: 部门编辑时未选择部门审批人无法保存问题修复
- 修复: 编辑工作流时候标题模板，内容模板未成功保存问题

- 修复: 处理人类型为工单字段时，获取处理人信息错误问题
- 修复: 配置流转时候目标状态不选时，导致流转列表出不来问题
- 修复: 管理后台中强制修改工单状态导致工单无法被继续处理问题
- 修复: 状态强制修改为初始状态或者结束状态时，处理人错误问题
- 修复: 调用权限编辑后再新增记录时，表单中遗留了上次编辑的内容问题
- 修复: readthedoc 文档中允许启动命令中中两个-被转成了一个–问题说明
- 修复: 使用 uwsgi 部署后，日志文件没有内容问题 (临时改成打印日志到控制台，可取 uwsgi 日志中查看日志)
- 优化: 新增工作流后提示用户去添加调用权限
- 优化: 配置工作流选择通知的地方，加个提示如何新增通知

24.4 r1.0.1

- 修复: 生产环境依赖包 uwsgi 版本更新
- 修复: 工单列表查询条件创建起止时间处理逻辑错误
- 修复: 评论工单接口逻辑错误
- 修复: 强制关闭后工单的进行状态属性未更新问题
- 修复: 状态参与人类型是角色时导致处理人异常问题
- 修复: 部分情况下工单列表接口查询我的待办工单返回数据错误
- 新增功能: 工单列表支持我处理过的工单查询
- 新增功能: 工单列表查询 api 的状态属性条件支持“已关闭”查询
- 优化: 管理后台中工单管理异常情况提示信息优化及一些其他细节优化

24.5 r1.0.0

- 升级 python3.6
- 配置文件统一修改为 config.py
- 新增接口：撤回工单
- 工单详情接口新增返回当前状态的详细信息
- 允许工单创建人在工单的初始状态直接关闭工单
- 工单列表接口性能优化

- flowstep 接口中新增返回当前状态信息，并且记录按照 state 的顺序 id 排序
- 工单列表查询接口新增支持查询条件: 草稿中、进行中、被撤回、被退回、完成
- 自定义通知由脚本修改为 hook 方式
- 管理后台首页新增工单数量分类统计
- 管理后台显示当前详细版本号
- 管理后台支持用户、部门、角色编辑
- 管理后台配置状态时，初始及结束状态隐藏处理人输入框信息
- 管理后台支持对工单干预处理: 直接关闭、转交、修改工单状态、删除
- 状态参与人类型是部门时，支持设置多个部门
- 流转操作支持目标状态为初始状态：不再需要额外配置一个”发起人编辑中“这样的中间状态
- 工作流状态 hook，支持配置额外参数信息
- 管理后台权限控制细化：分为超级管理员和工作流管理员
- 使用 readthedoc 管理项目文档
- 静态文件由 cdn 移到本地, 避免内网部署无外网访问权限时无法正常使用
- 代码结构及内部逻辑优化 (去除冗余代码、单例模式减少内存占用、数据库操作语句优化、type hints、view 参数强校验等)

24.6 r0.x.x

见 github release https://github.com/blackholll/loonflow/releases