一．准备工作

activiti插件安装，改编码，画流程图，properties配置信息，启动类注解

1. 内部无法下载，就外部下载然后拷到bin下，重启idea
2. 解决乱码问题，bin下.vmoptions文件里加-Dfile.encoding=UTF-8保存
3. 简单开启审核网关判断结束根据业务
4. **nullCatalogMeansCurrent**=**true***# activiti default configuration***spring.activiti.database-schema-update** = **true  
   spring.activiti.check-process-definitions** = **true  
   spring.activiti.process-definition-location-prefix** = **classpath:/processes/***# process-definition-location-suffixes:  
   # - \*\*.bpmn  
   # - \*\*.bpmn20.xml***spring.activiti.history-level** = *full*
5. @SpringBootApplication(exclude=SecurityAutoConfiguration.**class**)启动成功会新增24张表act\_开头

二，流程图部分

1. 流程成图的id是启动id，CandidateUsers指定任务办理人（可多个）

${users}可以设变量接受也可以写死，formKey制定操作路径

1. 网关后的线可以判断根据传入条件进行判断 格式${isgo=='true'}

（待深入了解）

三．主要接口

1. @Autowired  
RuntimeService **runtimeService**; *//流程控制服务*@Autowired  
TaskService **taskService**; *//任务服务*@Autowired  
RepositoryService **repositoryService**; *//管理流程仓库，部署，删除，读取*@Autowired  
HistoryService **historyService**; *//历史服务*@Autowired  
IdentityService **identityservice**;*//用户服务*

*下面运用里会有，这里就不多说了*

四．启动一个流程

*1.id***identityservice**.setAuthenticatedUserId(userId);

记录办理人或启动人的id

ProcessInstance instance=**runtimeService**.startProcessInstanceByKey(**"流程图id"**, *BusinessKey*业务主键,variables流程变量);*//BusinessKey一般为实例的主键实现业务数据与流程的绑定*

//variables就是一个map集合，根据流程图中定义的变量主键作为主键

Map<String,Object> variables=**new** HashMap<String, Object>();  
variables.put(**"users"**,userId);*//指定处理该事务的人,多个人用,分隔，实现传入参数值*

*至此流程就启动了*

五，流程流转

*1.* Task task = **taskService**.createTaskQuery().taskId(workId).singleResul();ProcessInstance ins=**runtimeService**.createProcessInstanceQuery()*//根据ProcessInstanceId获取当前运行的流程实例* .processInstanceId(task.getProcessInstanceId())  
 .singleResult();

可以根据ins.getBusinessKey()获得业务主键对业务状态进行更改

然后再次运用流程变量variables实现流程流转**taskService**.complete(workId,variables);

这里处理之后可根据processInstanceId获取实例看看如为null则流程已完结可配合业务实现修改业务状态，因为本方法基本是多次使用的

（备注：这里可能需要route也可能需要选取下级审核人，建议分别封装三个方法，后续会有大量重复调用）

六，个人待办

1. 正常按个人用户id查询

List<Task> tasks = **taskService**.createTaskQuery().orderByTaskCreateTime().desc()  
 .taskCandidateUser(role)  
 .listPage((starPage-1)\*size,size);

提供的有类似分页功能这边建议自己封装个分页工具类配合使用

List<Task> taskss = **taskService**.createTaskQuery()  
 .taskCandidateUser(role)  
 .list();

至此就获得了该用户的所有待办，这里是按CandidateUser查询， 还可以按分组，角色，指定人Assignee等可以灵活运用。

2.按处理业务分类加用户id获取个人待办

**if** (ins.getProcessDefinitionName().equals(typeName)){  
 newTasks.add(t);  
}简单来说就是在上一步的基础上再加一个筛选

**后续操作基本靠业务的接口，一定要写一个按主键id获取业务的接口，高频率使用**

**这个按分类查会较多，建议根据使用情况进行封装方法，避免代码冗余**

七．个人已办

**1.** List<HistoricTaskInstance> historicTaskInstances = **historyService***//与历史数据（历史表）相关的Service* .createHistoricTaskInstanceQuery().orderByHistoricTaskInstanceEndTime().desc()*//创建历史任务实例查询* .taskCandidateUser(role)*//指定历史任务的办理人* .finished()*//已完成的任务* .listPage((starPage-1)\*size,size);

这是操作已办理的流程历史纪录的接口的使用，也提供的有类似分页的方法，建议配合封装的分页工具类进行操作

HistoricProcessInstance hins=**historyService** *// 历史任务Service，根据ProcessInstanceId获取历史流程实例* .createHistoricProcessInstanceQuery() *// 创建历史流程实例查询* .processInstanceId(h.getProcessInstanceId()) *// 指定流程实例ID* .singleResult();

这里会有更细节的数据，比如BusinessKey等数据

1. 按个人用户id和业务类型查询已办

**if** (ins.getProcessDefinitionName().equals(typeName)){  
 newTasks.add(t);  
}和待办类似同样都建议封装方法使用

八．获取流程的审批记录

List<HistoricTaskInstance> historicTaskInstances = **historyService***//与历史数据（历史表）相关的Service* .createHistoricTaskInstanceQuery()*//创建历史任务实例查* .processInstanceId(processInstanceId)  
 .list();

根据流程实例id processInstanceId进行获取的。

九．流程变量的运用

1. **存：taskService**.setVariable(workId,**"opinions"**,opinions);  
**taskService**.setVariable(workId,**"reqEndTime"**,reqEndTime);

取：(String)**taskService**.getVariable(t.getId(),**"opinions"**)

//对于为完结的流程

这里根据任务id做得一个暂时性的流程可携带数据的操作，很方便，例如一些流成处理的附属信息，批注，要求等，但是流程结束，就没有了，而且没有了任务Id也没有办法区分，（待深入了解）。

还可以使用流程变量实现预设值审批人功能

对于两者都适用来说需要操作Comment批注

存：**taskService**.addComment(workId,task.getProcessInstanceId(),opinions);

取：List<Comment> comments = **taskService**.getProcessInstanceComments(h.getProcessInstanceId());  
**if** (comments!=**null**){  
 **for** (Comment c:comments) {  
 **if** (c.getTaskId().equals(h.getId())){  
 endList.setMessage(c.getFullMessage());  
 }  
 }  
}  
List<HistoricVariableInstance> lists = **historyService**.createHistoricVariableInstanceQuery().processInstanceId(h.getProcessInstanceId()).list();  
**if** (lists!=**null**){  
 **for** (HistoricVariableInstance h1:lists) {  
 **if**(h1.getVariableName().equals(**"reqEndTime"**)){  
 **if** (h1.getValue()!=**null**) {  
 endList.setReqEndTime(h1.getValue().toString());  
 }  
 }  
 }  
}

十．工作组的初步使用

1.可独立动态实现管理每个流程的审批人员

2.获取待办时变化不大，处理任务时，需要每次传入处理人的id进行任务认领，最后进行处理这样是最后查询历史审批人时不会造成查不到的情况