**考勤系统设计文档**

****

**学 院 智能与计算学部**

**专 业 软件工程**

**班 级 2**

**学 号 3018216116**

**姓 名 尤嘉宁**

目录

[0. 设计思路 3](#_Toc38314427)

[1. RESTful API 4](#_Toc38314428)

[1.1 登录页面 4](#_Toc38314429)

[1.3 申请页面 5](#_Toc38314430)

[1.4 个人页面 6](#_Toc38314431)

[1.5 其他员工页面 7](#_Toc38314432)

[1.6 行政页面 8](#_Toc38314433)

[1.7 财务页面 10](#_Toc38314434)

[1.8 消息页面 11](#_Toc38314435)

[1.9 审批检查页面 12](#_Toc38314436)

[1.10 全体员工一览页面 13](#_Toc38314437)

[2.Controller VO设计 14](#_Toc38314438)

[2.Service BO设计 16](#_Toc38314439)

[3. Repository DO设计 18](#_Toc38314440)

[4.Controller-Service-Repository接口设计 20](#_Toc38314441)

[5.数据库设计(E/R图) 24](#_Toc38314442)

## 0. 设计思路

如图所示, 由于本系统内部使用TCP/IP进行通信, 那么系统的架构应与网站架构相似.

那么, 当系统外部人员(或者设备)发来请求时, 首先应该访问暴露给外部的RESTful api, 按照API规定的动词和字段构造请求. 当请求到达服务器后, Controller负责提取请求中的字段和动词, 将字段中的数据封装入BO(business object)对象, 并根据url中的路径和动词路由至Service层中的相关业务方法.

随后, Service中的具体业务方法应该按照预先编写的流程和BO中设置的相关方法, 对数据进行操作, 例如更新相关变量的值, 发送邮件通知和(如果有必要)调用Repository层中对数据库的操作. 如果请求的动词是GET的话, 还需要构造response并以VO(ViewObject)的形式返回给Controller. Controller将其转为Json的形式后返回给客户端.

而Repository则负责具体的对数据库增删查改. DO(data object)的成员变量和数据库中的数据表字段相对应.

## 1. RESTful API

### 1.1 登录页面

#### 1.1.1 登录

注: 如果只有一个json代码段, 说明此API不需要回复任何消息, 若为两个json代码段, 则第一个为请求, 第二个为回复. 下同.

# POST /api/user/

{

'id':'123456',

'passwd':'654321'

}

#### 1.2 打卡页面

1.2.1 打卡

# POST /api/record/

{

'id':'123456',

'datetime':'2020-4-11'

}

#### 1.2.2 加班登记

# POST /api/record/overtime

{

'id':'123456'

'datetime':'2020-4-11 22:00'

}

### 1.3 申请页面

#### 1.3.1 申请请假

# POST /api/application/leave

{

'id':'123456',

'startTime':'2020-4-11 22:00',

'endTime':'2020-4-12 22:00',

'type':'annualLeave',

'reason':'I want to play a game'

}

#### 1.3.2 外出申请

# POST /api/application/out

{

'id':'123456',

'startTime':'2020-4-11 22:00',

'endTime':'2020-4-12 22:00',

'reason':'I want to play a game'

}

#### 1.3.3 补签

# POST /api/application/remedy

{

'id':'123456',

'date':'2020-4-11',

'reason':'I wanted to play a game'

}

#### 1.3.4 修改请假申请

# PUT /api/application/mod/leave

{

'id':'123456',

'applicationId':'1',

'leaveId':'abc',

'startTime':'2020-4-11 21:00',

'endTime':'2020-4-13 21:00',

'type':'annualLeave',

'reason':'I still want to play a game'

}

#### 1.3.5 修改外出申请

# PUT /api/application/mod/out

{

'id':'123456',

'outId':'abc',

'startTime':'2020-4-11 21:00',

'endTime':'2020-4-13 21:00',

'reason':'I still want to play a game'

}

#### 1.3.6 修改补签申请

# PUT /api/application/mod/remedy

{

'id':'123456',

'remedyId':'abc',

'date':'2020-4-10',

'reason':'I still wanted to play a game'

}

### 1.4 个人页面

#### 1.4.1 查看考勤记录

# GET /api/record

{

'id':'123456'

}

{

'records':[

{

'date':'2020-4-09',

'status':'leave'

},

{

'date':'2020-4-10',

'status':'present'

},

{

'date':'2020-4-11',

'status':'non-present'

}

]

}

#### 1.4.2 查看当前状态

# GET /api/user/status

{

'id':'123456'

}

{

'status':'present'

}

#### 1.4.3 查看剩余假期额度

# GET /api/user/Holidaybalance

{

'id':'123456'

}

{

'annualLeave': 7

}

### 1.5 其他员工页面

#### 1.5.1 查看员工列表

# GET /api/user/employees/

{

'id':'123456'

}

{

'members':[

{

'id':'123456',

'name':'尤嘉宁',

'title':'总经理'

},

{

'id':'456789',

'name':'mv',

'title':'副总经理'

}

]

}

#### 1.5.2 查看其它员工当前状态

# GET /api/user/employees/654321

{

'id':'123456',

'otherUserId':'654321'

}

{

'id':'654321',

'name':'胡阿尤',

'status':'present'

}

### 1.6 行政页面

#### 1.6.1 获取全体员工的考勤数据

# GET /api/record/allData

{

'id':'741852',

'startTime':'2020-3-12',

'endTime':'2020-3-13'

}

{

'datas':[

{

'id':'123456',

'name':'尤嘉宁',

'records':[

{

'date':'2020-3-12',

'status':'present'

},

{

'date':'2020-3-13',

'status':'present'

}

],

'overtime':[

{

'startTime':'2020-3-12 20:00',

'endTime':'2020-3-12 22:00'

}

]

},

{

'id':'654321',

'name':'mv',

'records':[

{

'date':'2020-3-12',

'status':'leave'

},

{

'date':'2020-3-13',

'status':'out'

}

],

'overtime':[]

}

]

}

#### 1.6.2 上传行政部处理后的数据

# POST /api/administration-department/after-process/data/

{

'data':[

{

'id':'123456',

'present':[

'2020-3-12',

'2020-3-13'

],

'leave':[

'2020-3-14'

],

'out':[

'2020-3-15'

],

'non-present':[]

}

]

}

### 1.7 财务页面

#### 1.7.1 获取行政部处理后数据

# GET /api/administration-department/after-process/data/

{

'id':'963852'

}

{

'data':[

{

'id':'123456',

'present':[

'2020-3-12',

'2020-3-13'

],

'leave':[

'2020-3-14'

],

'out':[

'2020-3-15'

],

'non-present':[]

}

]

}

#### 1.7.2 上传财务部处理数据

# POST /api/finance-department/after-process/data/

{

'data':[

{

'id':'123456',

'salaryAdd':'300'

},

{

'id':'852741',

'salaryAdd':'-200'

}

]

}

### 1.8 消息页面

#### 1.8.1 获取消息列表(包括审批结果和申请信息)

# GET /api/user/messages/

{

'id':'123456'

}

{

'messages':[

'applicationResult':[

'applicationId':'1',

'result':'pass',

'LeadersOpinion':[

{

'title':'总经理',

'name':'尤嘉宁',

'opinion':'I agree with you!'

}

]

],

'applicationNeedToProcess':[

'applicationId':'2',

'startTime':'2020-4-11 22:00',

'endTime':'2020-4-12 22:00',

'type':'annualLeave',

'reason':'I want to play a game'

]

]

}

#### 1.8.2 处理申请

# POST /api/application/process-command/

{

'id':'123456',

'applicationId':'1'

'result':'pass',

'opinion':'I agree with you!'

}

### 1.9 审批检查页面

#### 1.9.1 检查其他管理者的审批

# GET /api/manager/approval-results/

{

'id':'123456'

}

{

'otherManagerApprovalResults':[

{

'id':'123456',

'startTime':'2020-4-11 22:00',

'endTime':'2020-4-12 22:00',

'type':'annualLeave',

'reason':'I want to play a game',

'managerId':'741852',

'managerName':'mv',

'result':'pass',

'opinion':'I agree with you!'

}

]

}

### 1.10 全体员工一览页面

#### 1.10.1 查看全体员工的请假情况

# GET /api/manager/all/leaving-members/

{

'id':'123456'

}

{

'leaveMembers':[

{

'userId':'741852',

'userName':'mv'

},

{

'userId':'987654',

'userName':'seeler'

}

]

}

#### 1.10.2 查看全体员工的外出情况

# GET /api/manager/all/outing-members/

{

'id':'123456'

}

{

'outMember':[

{

'userId':'741852',

'userName':'mv'

},

{

'userId':'987654',

'userName':'seeler'

}

]

}

#### 1.10.3 查看全体员工的加班情况

# GET /api/manager/all/overtime-members/

{

'id':'123456'

}

{

'overtimeMembers':[

{

'userId':'741852',

'userName':'mv'

},

{

'userId':'987654',

'userName':'seeler'

}

]

}

## 2.Controller VO设计

ViewObject对象在本项目中, 应在Controller层被转为json格式, 返回给api调用者.

1. ViewObject

作为其它ViewObject的基类. 带有toJson方法, 可以把该VO对象转为Json格式, 返回给调用者.

class ViewObject:

*'''VO基类'''*

    def toJson(self):

*pass*

2. PersonRecordViewObject

某用户的考勤记录的集合.

class PersonRecordViewObject(ViewObject):

*'''某个员工的考勤记录'''*

    records:List[RecordViewObject]

3. currentStatusViewObject

某用户的当前状态.

class CurrentStatusViewObject(ViewObject):

*'''某员工的当前考勤状态'''*

    status:str

4. HolidayBalanceVO

某员工的假期余额

class HolidayBalanceVO(VO):

*'''某员工的假期余额'''*

    annualLeave:int

5. UserListVO

全体员工的列表

class UserListVO(VO):

*'''全体员工的列表'''*

    members:List[MemberBO]

6. OtherMemberStatusVO

其他某员工的当前状态

class OtherMemberStatusVO(VO):

*'''其他某员工的当前状态'''*

    id:str

    name:str

    status:str

7. AllRecordDataVO

全体员工的考勤记录数据, 供行政部门使用

class AllRecordDataVO(VO):

*'''全体员工的考勤记录数据, 供行政部门使用'''*

    datas:List[MemberWithRecordDataBO]

8.AdMinistrationResultVO

行政部门处理后的数据集合

class AdMinistrationResultVO(VO):

*'''行政部门处理后的数据集合'''*

    data:List[MemberWithAdministrationResultBO]

9. MessagesVO

所有可查看的消息

class MessagesVO(VO):

*'''所有可查看的消息'''*

    messages:List[Message]

10. LeaveMembersVO

所有休假的员工

class LeaveMembersVO(VO):

*'''所有休假的员工'''*

    leave\_members:List[MemberBO]

11. OutMembersVO

class OutMembersVO(VO):

*'''所有外出工作的员工'''*

    out\_members:List[MemberBO]

12. OverTimeMembersVO

class OverTimeMembersVO(VO):

*'''所有加班的员工'''*

    over\_time\_members:List[MemberBO]

## 2.Service BO设计

0. VO

VO基类

class BO:

*'''BO基类'''*

*pass*

1. RecordBO

代表一条考勤记录

class RecordBO(BO):

*"""代表一条考勤记录"""*

    data:str

    status:str

2. MemberBO

代表某位员工

class MemberBO(BO):

*'''代表某位员工'''*

    id:str

    name:str

    title:str

3. MemberWithRecordDataBO

某位员工及其部分考勤记录, 考勤记录具体有哪些条, 应该由Service中具体业务规则决定.

class MemberWithRecordDataBO(BO):

*'''某员工及其部分考勤记录'''*

    id:str

    name:str

    records:List[RecordBO]

4. MemberWithAdministrationResultBO

某员工及其行政部门处理后的数据, 即对其考勤记录数据的加工.

class MemberWithAdministrationResultBO(BO):

*'''某员工及其行政部门处理后的记录'''*

    id:str

    present:List[str]

    leave:List[str]

    out:List[str]

    non\_present:List[str]

5. LeaderOpinionBO

代表某个领导意见

class LeaderOpinionBO(BO):

*'''代表某个领导意见'''*

    title:str

    leader\_name:str

    opinion:str

6. ApplicationBO

代表申请表的基类

class ApplicationBO(BO):

*'''代表申请表的基类'''*

member\_id:str

    applicaiton\_id:str

    startTime:str

    endTime:str

    reason:str

7. LeaveApplicationBO

代表一个休假申请

class LeaveApplicationBO(ApplicationBO):

*'''代表一个休假申请'''*

    \_\_type:str

8. OutApplicationBO

代表一个外出申请

class OutApplicationBO(ApplicationBO):

*'''代表一个外出申请'''*

Pass

9. MessageBO

可供查看的消息

class MessageBO(BO):

*"""可供查看的消息"""*

    application:List[ApplicationBO]

10. ApplicationResultMessageBO

可供查看的申请处理结果信息

class ApplicationResultMessageBO(Message):

*'''可供查看的申请处理结果信息'''*

    result:str

    leadersOpinion:List[LeaderOpinionBO]

11. ApplicationNeededProcessMessageBO

可供查看的需要员工处理的申请消息

class ApplicationAllInfoBO(BO):

*'''一个申请的全部消息, 包括请求信息和处理者,处理结果等'''*

    application:ApplicationBO

    result:str

    leadersOpinion:List[LeaderOpinionBO]

12. ApplicationAllInfoBO

一个申请的全部消息, 包括请求信息和处理者,处理结果等,处理意见等等

class ApplicationAllInfoBO(BO):

*'''一个申请的全部消息, 包括请求信息和处理者,处理结果等'''*

    application:ApplicationBO

    result:str

    leadersOpinion:List[LeaderOpinionBO]

## 3. Repository DO设计

1. DO

DO基类

class DO:

*'''DO基类'''*

    def insert(self):

*pass*

    def delete(self):

*pass*

    def select(self):

*pass*

    def update(self):

*pass*

2. MemberDO

对应操作关系member

class MemberDO(DO):

*'''对应操作关系member'''*

    name:str

    id:str

    title:str

3.RecordDO

操作关系record

class RecordDO(DO):

*'''操作关系record'''*

    id:str

    date:str

    \_\_type:str

4. OutApplicationDO

操作关系out\_application

class OutApplicationDO(DO):

*'''操作关系out\_application'''*

    application\_id:str

    startTime:str

    endTime:str

    user\_id:str

    reason:str

    result:str

    processor\_id: List[str]

    opinion: List[str]

5. OutApplicationDO

操作关系out\_application

class OutApplicationDO(DO):

*'''操作关系out\_application'''*

    application\_id:str

    startTime:str

    endTime:str

    user\_id:str

    reason:str

    result:str

    processor\_id: List[str]

    opinion: List[str]

    \_\_type:str

6. Message

对应操作关系message

class Message(DO):

*"""对应操作关系message"""*

    send\_to\_user\_id:str

    application\_id:str

7. HolidayBalanceDO

操作关系holiday\_balance

class HolidayBalanceDO(DO):

*'''操作关系holiday\_balance'''*

    id:str

    \_\_type:str

    balance:int

## 4.Controller-Service-Repository接口设计

class Service:

    controller:Controller

    def login(self)->str:

*pass*

    def sign\_in(self, request:str)->str:

*pass*

    def overtime\_record(self, request:str)->str:

*pass*

    def leave\_applicate(self, request:str)->str:

*pass*

    def out\_applicate(self, request:str)->str:

*pass*

    def remedy\_applicate(self, request:str)->str:

*pass*

    def update\_leave\_applicaiton(self, request:str)->str:

*pass*

    def update\_out\_applicate(self, request:str)->str:

*pass*

    def update\_remedy\_applicate(self, request:str)->str:

*pass*

    def get\_records(self, request:str)->str:

*pass*

    def get\_status(self, request:str)->str:

*pass*

    def get\_holiday\_balance(self, request:str)->str:

*pass*

    def get\_members\_list(self)->str:

*pass*

    def get\_member\_status(self, request:str)->str:

*pass*

    def get\_all\_members\_record(self, request:str)->str:

*pass*

    def commit\_result\_after\_administration(self, request:str)->str:

*pass*

    def get\_result\_after\_administration(self, request:str)->str:

*pass*

    def commit\_result\_after\_finance(self, request:str)->str:

*pass*

    def get\_messages(self, request:str)->str:

*pass*

    def process\_application(self, request:str)->str:

*pass*

    def get\_application\_results(self, request:str)->str:

*pass*

    def get\_leave\_members(self, request:str)->str:

*pass*

    def get\_out\_members(self, request:str)->str:

*pass*

    def get\_overtime\_members(self, request:str)->str:

*pass*

class Controller:

    repository:Repository

    def login(self)->VO:

*pass*

    def sign\_in(self, user\_id)->VO:

*pass*

    def overtime\_record(self, user\_id:str, date:str)->VO:

*pass*

    def leave\_applicate(self, leave\_application:LeaveApplicationBO)->VO:

*pass*

    def out\_applicate(self, out\_application:OutApplicationBO)->VO:

*pass*

    def remedy\_applicate(self, remedy\_applicate:RemedyApplicationBO)->VO:

*pass*

    def update\_leave\_applicaiton(self, leave\_application:LeaveApplicationBO)->VO:

*pass*

    def update\_out\_applicate(self, out\_application:OutApplicationBO)->VO:

*pass*

    def update\_remedy\_applicate(self, remedy\_applicate:RemedyApplicationBO)->VO:

*pass*

    def get\_records(self, id:str)->PersonRecordVO:

*pass*

    def get\_status(self, id:str)->CurrentStatusVO:

*pass*

    def get\_holiday\_balance(self, id:str)->HolidayBalanceVO:

*pass*

    def get\_members\_list(self)->UserListVO:

*pass*

    def get\_member\_status(self, id)->OtherMemberStatusVO:

*pass*

    def get\_all\_members\_record(self, id, startTime:str, endTime:str)->AllRecordDataVO:

*pass*

    def commit\_result\_after\_administration(self, id)->VO:

*pass*

    def get\_result\_after\_administration(self, id)->AdMinistrationResultVO:

*pass*

    def commit\_result\_after\_finance(self, id)->VO:

*pass*

    def get\_messages(self, id)->MessagesVO:

*pass*

    def process\_application(self, applicaiton\_id:str, result:str, opinion:str)->VO:

*pass*

    def get\_application\_results(self, id)->AdMinistrationResultVO:

*pass*

    def get\_leave\_members(self, id)->LeaveMembersVO:

*pass*

    def get\_out\_members(self, id)->OutMembersVO:

*pass*

    def get\_overtime\_members(self, id)->OverTimeMembersVO:

*pass*

class Repository:

    def qurey\_user\_password(self, id)->MemberDO:

*pass*

    def insert\_sign\_in\_record(self, record:RecordDO):

*pass*

    def insert\_overtime\_record(self, record:RecordDO):

*pass*

    def insert\_leave\_application(self, leave\_application:LeaveApplicationDO):

*pass*

    def insert\_out\_application(self, out\_application:OutApplicationDO):

*pass*

    def update\_leave\_application(self, leave\_application:LeaveApplicationDO):

*pass*

    def update\_out\_application(self, out\_application:OutApplicationDO):

*pass*

    def qurey\_person\_records(self, id)->List[RecordDO]:

*pass*

    def qurey\_holiday\_balance(self, id)->List[HolidayBalanceDO]:

*pass*

    def qurey\_members\_list(self)->List[MemberDO]:

*pass*

    def qurey\_all\_record(self)->List[RecordDO]:

*pass*

    def qurey\_person\_message(self)->List[MessageDO]:

*pass*

    def update\_out\_application\_result(self, result:str, opinion:LeaderOpinionBO):

*pass*

    def qurey\_all\_application(self)->List[ApplicationBO]:

*pass*

## 5.数据库设计(E/R图)

