## 需求list

将需求形成一条一条的记录

例如

想要在详情页上展示头像

想要授权才能加好友等

## 功能点分析

详情页信息展示功能,展示头像,发布者等

好友审批功能

## 接口设计

接口设计原则

1. 对于单个条记录的crud 使用主键直接传入参数
2. 对于存在 修改,删除,更新的接口,应该在后台使用身份认证机制,防止数据越权漏洞
3. 接口的梳理采用业务流的方式,即使用一个角色从头走到尾
4. 如果存在多个角色,那么分角色使用业务流的方式进行梳理
5. 接口字段的使用应该和数据库中的字段对应,这样在json,jersey的时候可以复用model对象
6. 在存在修改,删除,更新的表中,应该存在隐含的owner字段和create\_time字段,方便更新与维护
7. 接口设计时的字段原则,采用驼峰lowerCamel,字段的命名应该统一,比如userid ,不能一时userId 一时 userid
8. 接口的命名,应该采用实体作为父path,操作作为子path,比如对用户表的操作接口
   1. /user/save
   2. /user/delete
   3. /user/getDetail
   4. /user/updateIdentity

/{实体}/{动词}{名词}

1. 接口设计完成后,就禁止修改,可以增加

10.接口文档的作为前后台交接的工具,应该标注提交方式,提交的数据类型,返回的数据类型等,在设计接口的时候,应该使用接口设计工具如eolinker等,好处是可以在线测试接口可用性

11.在设计接口的时候,应该提供测试用例,应该提供两组,一组用来进行接口后台的验收,一组用来进行接口使用方的验收

12.明确字段含义如果返回值返回id 表明这个appid那么这个字段的命名就应该是appid,而不应该为了简洁而使用id

## 后台的接口实现

1. 复用model层,作为dto对象,在jersey中为了测试,如果引用类型没有被复制,那么在这个对象作为回传参数的时候,没有赋值的属性不会被返回
2. 使用枚举作为常量值的表示,创建的方式是与数据库对应
3. 在接口方法中应该在controller层,进行参数校验,参数错误直接返回
4. 注意nullpointexception的产生与解决,解决方案是在controller进行参数校验,往下传的参数一定不为空,只有引用类型才会报空指针,所以在find\*,get\*等方法的返回值是,应该要校验
5. 数据越权应该在controller层进行校验,只有正确的数据才回到service层进行业务处理
6. 日志功能应该使用工具类使用api的形式进行如 logutil.info(“收到”),这样做的好处是方面日志的迁移,可以统一管理某一个类型的日志,此外sf4j,和log4j2,以及logback相比 性能上logback>log4j2>sf4j
7. 字段命名采用驼峰lowerCamel,方法命名根据和接口命名规则一致
8. 在使用orm框架的时候,应该注意缓存和事务的控制,hibernate的缓存和session有关

## 数据库表设计原则

1. 字段名如果多个名词则采用下划线分割
2. 数据库字段名应该小写
3. 数据库的外键在代码上进行规范,不在数据库中进行限制
4. 数据库的表结构应该和接口的字段结构对应,这样可以跟容易理解接口的含义,以及代码上进行接口的参数控制
5. 数据库中的时间类型应该使用datetime而不应该使用timestamp
6. 数据库中存表情应该使用utf8mb4
7. 数据库中的表实体之间,如果多个实体相互管理,那么尽量使用一张中间表,维护多个实体的映射关系
8. 数据库中的表命名采用 实体命名,关联表的命名采用实体下划线分割

## 验收和测试

1. 采用单接口测试原则测试,接口的可用性
2. 采用数据流测试,测试模块的可用性
3. 采用整体测试,测试系统的可用性

### 上线和维护

1. 上线不需要显示sql语句,只要日志就够了
2. 采用自动发布的部署方式,可以是脚本也可以是jerkins的工具

