Alice-拯救你的孤独 （ 0180929 ）概要设计说明书

文件状态：正式发布
文件标识：NPC2018110110
当前版本：ver1.0

作者：

姓名：孙伊凡，李迪安...
学号：201501191,20150...
团队：NPC

完成日期：2018/11/1

版本更新信息：

2018-11-1 ver1.0 第一次编写完成

1 引言

1.1 使用人员：

该项目业务方负责人，项目经理，开发组长，产品经理

1.2 编写目的：

本文档对《Alice-拯救你的孤独》软件开发需求进行了详细说明。文档首先给出项目的整体结构和功能结构概貌，试图从总体架构上给出整个系统的轮廓。同时对功能需求、性能需求进行了详细的描述。 便于用户、开发人员进行理解和交流，反映出用户问题的结构，可以作为软件开发工作的基础和依据以及确认测试和验收的依据。
本文档面向多种读者对象：
(1)项目经理:项目经理可以根据该文档了解预期产品的功能，并据此进行系统设计、项目管理。
(2)设计员:对需求进行分析， 并设计出系统，包括数据库的设计。
(3)程序员:了解系统功能，编写《用户手册》。
(4)测试员:根据本文档编写测试用例，并对软件严品进行功能性测试和非功能性测试。
(5)用户:了解预期产品的功能和性能，并与分析人员一一起对整个需求进行讨论和协商。
在阅读本文档时，首先要了解产品的功能概貌，然后可以根据自身的需要对每一功能进行适当的了解。

1.3 背景：

待开发的软件系统的名称：Alice-拯救你的孤独
本项目的任务提出者：NPC用户组
开发者：NPC软件开发团队
用户：广大人民群众
实现该软件的计算中心或计算机网络：WLAN/4G移动网络

1.4 定义与缩写

术语：Nginx
解释：

*Nginx*是一个高性能的HTTP和反向代理服务，也是一个IMAP/POP3/SMTP服务。

术语：uWSGI
解释：

uWSGI是一个Web服务器，它实现了WSGI协议、uwsgi、http等协议。Nginx中HttpUwsgiModule的作用是与uWSGI服务器进行交换。WSGI是一种Web服务器网关接口。它是一个Web服务器（如nginx，uWSGI等服务器）与web应用（如用Flask框架写的程序）通信的一种规范。

术语：django
解释：
Django是一个开放源代码的Web应用框架，由Python写成。采用了MVC的框架模式，即模型M，视图V和控制器C。它最初是被开发来用于管理劳伦斯出版集团旗下的一些以新闻内容为主的网站的，即是CMS（内容管理系统）软件。

1.5 参考资料

- 《第一行代码——Android》郭霖著，人民邮电出版社，出版时间2014年8月
- activitypub
- activitystream

2 总体设计

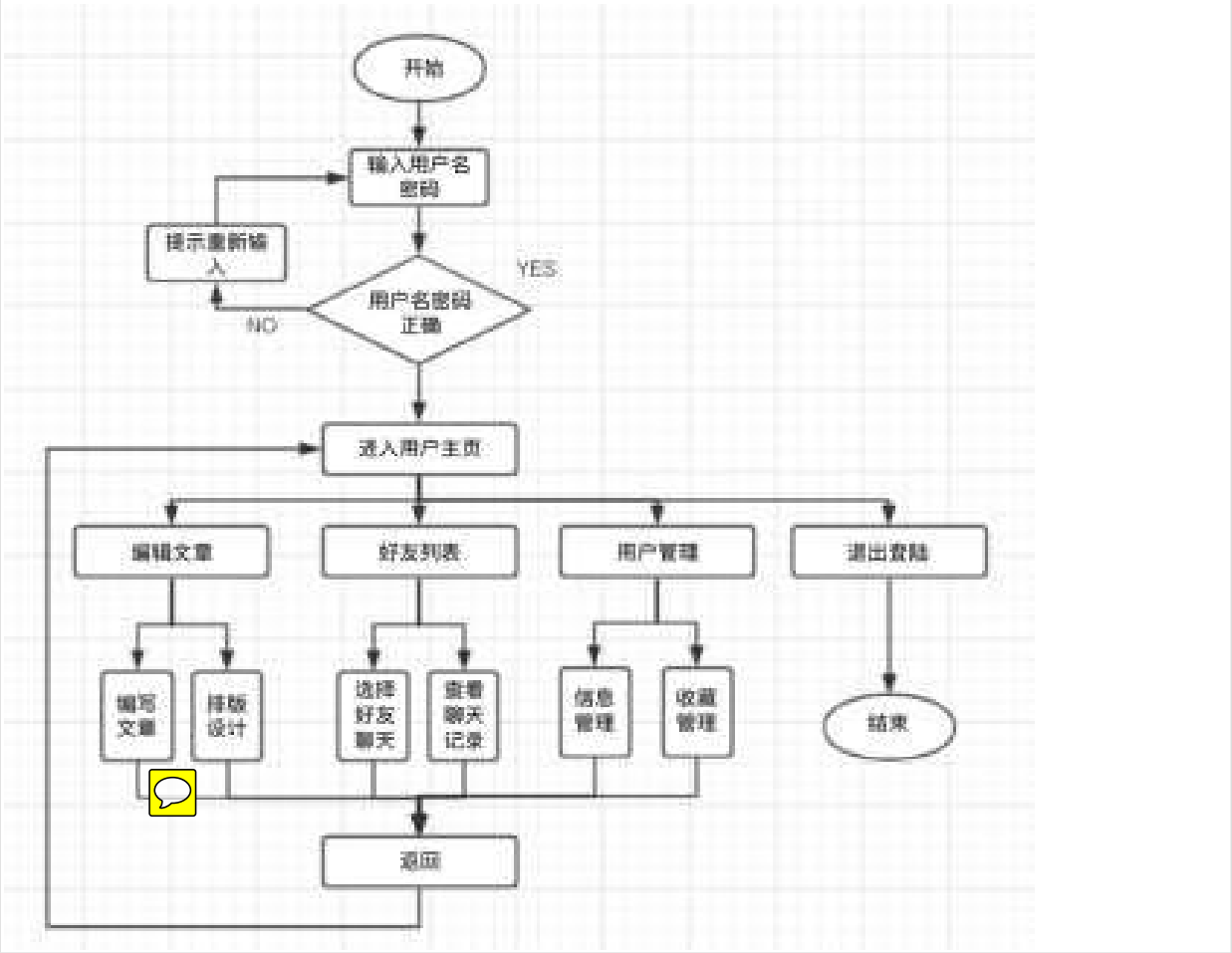
2.1 需求规定：

Alice软件的使用者为用户和管理员。用户通过它来查询核对自己的信息，与他人聊天、分享文章等，管理员则通过它来达到查询管理用户信息以及文章图片信息的目的。
A、主要输入：注册信息、好友信息、聊天内容信息都会输入到数据库中保存
B、主要输出：查询信息、用户信息、聊天记录信息、好友分享的文章信息
进入登录系统的界面，主要是对系统用户进行管理，登录退出。在根据需求设定该系统的时候规定了用户角色，包括用户和管理员，用户的管理权限是不断的包围的。
用户管理模块：查询已有的用户、删除注销的用户、修改用户权限、用户聊天记录的查看。
文章模块：用户可以对自己的文章进行编辑和排版发布。用户进行文章查询时，应将文章名称以列表的形式直观展现。
聊天模块：用户选择好友聊天。对于聊天信息查询，将数据直接输出到的查询界面上。

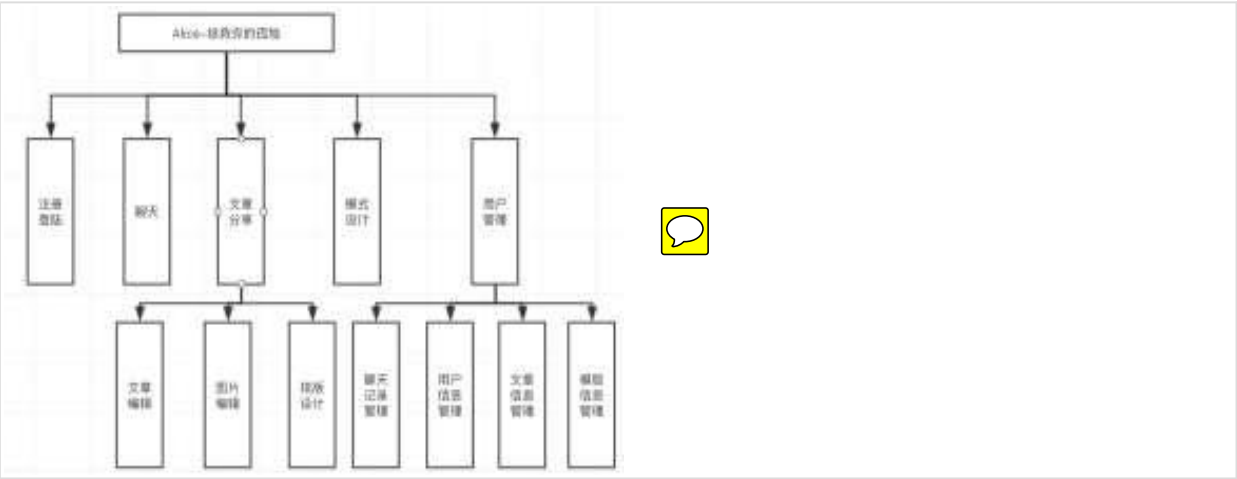
2.2 运行环境：

支持android 5.1以上的智能手机，处理器双核1GHz以上，内存1GB以上，可以用数据服务与服务器相连，源码为2进制，其中采用编码成汉字与数字，也支持字母符号。以三星、华为为代表的数据通信设备均可，支持一对多访问数据，支持设置快捷键。

2.3 基本设计概念和处理流程：



2.4 结构：



2.5 功能需求与个部分程序的关系：

序号	功能名称	功能需求标识	优先级	简要描述
1	用户登录子系统	用户登录	A	用户按等级进行登录
2	用户聊天子系统	聊天功能	A	一般用户之间进行聊天，查看聊天记录
3	用户文章子系统	文章功能	A	分享和查看文章
4	管理员子系统	管理员管理	A	管理和查看用户信息

2.6 人工处理过程：

软故障：
对本人的操作被他人冒用的情况，系统管理员可凭身份证等身份证明清除用户生的数据，以便由其本人进行测试。

硬故障：
网络不通，排除故障后需要重新进入系统，系统不保存在用户提交测评结果前的临时数据。
在统计过程中服务器当机，可在重启服务器后再统计一次即可。

2.7 尚未问决的问题：

消息及时性、丢包的问题没有解决

2.8 自由表述：

开发基于Android系统的图文社交软件，用户可以更加方便的分享自己内心的想法或生活现状，设计出更优美舒适的图文布局。

3 接口设计

3.1 用户接口：

注册（app）

请求

采用POST方法，地址为/masuser/createmasuser。

参数	类型	是否必须	描述
phoneNumber	String	是	用户手机号
password	String	是	password=md5(password.append(reverse(username)))

响应


```
{
    "msgCode": 666,
    "msg": {
        "masuser": {
            "uid": "9609352425",
            "nick_name": "",
            "slogan": "",
            "work_mes": "",
            "interest_mes": "",
            "travel_mes": "",
            "avatar": {
                "avatar_image": 0,
                "avatar_color": 0
            },
            "created_time": 1540089381.400334
        },
        "token": "ZXlKMGVYQWlPaUpLVjFBaUxDSmhiR2NpT2lKalpXWmhkV3gwSW4wOjFnRTNiSjoOTFIyekEyNzNFMO1KbWpJUFdadVFKYjlJVWs.ZXlKMWMyVnlibUZOWlNJNklqazJNRGt6TlRJME1qVWlMQOpwWVhRaU9qRTFOREF3TORrek9ERXVOREEyTWpRNWZROjFnRTNiSjpLQk1jMUNOZzZ6dmlsWXRzbFc1dWFGYmhWZ2s.6aeaffa75b6d9375f9ee77ccb16b2bea"
    }
}
```


登录

app 端请求

采用POST方法，地址为/masuser/login

参数	类型	是否必须	描述

phoneNumber	String	是	用户手机号	130
sign	String	是	sign= md5(password(加密规则与登录一样) + timestamp)	900

响应


```
{
    "msgCode": 666,
    "msg": {
        "masuser": {
            "uid": "7028492784",
            "nick_name": "",
            "slogan": "",
            "work_mes": "",
            "interest_mes": "",
            "travel_mes": "",
            "avatar": {
                "avatar_image": 0,
                "avatar_color": 0
            },
            "created_time": 1540091094.890288
        },
        "token": "ZXlKMGVYQWlPaUpLVjFBaUxDSmhiR2NpT2lKalpXWmhkV3gwSW4wOjFnRTQOVjpRcFh1czRFeS
10UXRvU2g0bGc4SndzROdwN28.ZXlKMWMyVnlibUZOWlNJNklqY3dNamcwT1RJMO9EUWlMQOpwWVhRaU9qRTFOREF3T1
RFeE9URXVNVE01TkRFMO1uMDoxZOUONFY6cG9TbjBGVHotSHpzZDY5NUdNRWp4cjNucWFB.ec3a897d667539303f970
d3c0fe5a626"
    }
}
```


修改用户信息

请求

采用POST方法，地址为/masuser/updateUser

参数	类型	是否必须	描述	
slogan	String	否（不修改请给空）	用户签名，最长50个字符	"男/
work_mes	String	否（不修改请给空）	工作信息，最长20个字符	"北京
interest_mes	String	否（不修改请给空）	兴趣爱好，最长20个字符	"打球
travel_mes	String	否（不修改请给空）	曾经去过的旅行，最长20个字符	"新疆
nick_name	String	否（不修改请给空）	用户昵称	PJHub

响应


```
{
    "msgCode": 666,
    "msg": {
        "masuser": {
            "uid": "7028492784",
            "nick_name": "pjhubs",
            "slogan": "2333",
            "work_mes": "2333",
            "interest_mes": "2333",
            "travel_mes": "2333",
            "avatar": {
                "avatar_image": 1,
                "avatar_color": 1
            },
            "created_time": 1540091094.890288
        }
    }
}
```

获取某个用户详细信息

请求

采用GET方法，地址为/masuser/getUserDetails

参数	类型	是否必须	
无	无	无	无

响应


```
{
    "msgCode": 666,
    "msg": {
        "masuser": {
            "uid": "7028492784",
            "nick_name": "",
            "slogan": "",
            "work_mes": "",
            "interest_mes": "",
            "travel_mes": "",
            "avatar": {
                "avatar_image": 1,
                "avatar_color": 1
            },
            "created_time": 1540091094.890288
        }
    }
}
```


发布新文章

请求

采用POST方法，地址为/blog/createBlog

参数	类型	是否必须	描述
content	String	是	文章内容实体，不限长度，但前端最好做内容限制

响应


```
{
    "msgCode": 666,
    "msg": "发布成功"
}
```


获取所有文章列表

请求

采用GET方法，地址为/blog

参数	类型	是否必须
page	String	否，默认为第一页，一页返回10条文章

响应


```
{
    "msgCode": 666,
    "msg": {
```

```
        "blogs": [
            {
                "id": 2,
                "content": "新人再报道！",
                "masuser_id": "7028492784",
                "created_time": "2018-10-21T09:00:35.153Z",
                "last_updated_time": "2018-10-21T09:00:35.153Z",
                "is_deleted": 0,
                "read_num": 0
            },
            {
                "id": 1,
                "content": "新人报道！",
                "masuser_id": "7028492784",
                "created_time": "2018-10-21T08:51:23.223Z",
                "last_updated_time": "2018-10-21T08:51:23.223Z",
                "is_deleted": 0,
                "read_num": 0
            }
        ]
    }
}
```


删除文章

请求

采用POST方法，地址为/blog/deleteBlog

参数	类型	是否必须	
blog_id	String	是	文

响应


```
{
    "msgCode": 666,
    "msg": "删除成功",
}
```


获取某个用户发布的所有文章

采用GET方法，地址为/blog/userBlogs

参数	类型	是否必须
page	String	否，默认为第一页，一页返回10条文章

响应


```
{
    "msgCode": "666",
    "msg": {
        "blogs": [
            {
                "id": 7,
                "content": "不想去上班。_(:з」∠)_。啊哈哈哈哈哈",
                "created_time": "2018-08-02T12:40:16.087Z"
            },
            {
                "id": 4,
                "content": "重构一版，感觉还行吧~",
                "created_time": "2018-08-02T12:38:26.401Z"
            },
            {
                "id": 3,
                "content": "不太对劲，感觉哪里怪怪的~",
```

```
                "created_time": "2018-08-02T12:37:07.340Z"
            }
        ]
    }
}
```


获取某篇文章

请求

采用GET方法，地址为/blog/blogDetails

参数	类型	是否必须	
blog_id	String	是	文

响应


```
{
    "msgCode": "666",
    "msg": {
        "blog": {
            "read_num": 9,
            "comment_num": 1,
            "like_num": 1,
            "blog_content": "不太对劲，感觉哪里怪怪的~",
            "blog_created_time": 1533213427.34092
        },
        "masuser": {
"masuser_id": 2,
"nick_name": "pjhubs",
"slogan": "哎呀，我的妈呀！",
"work_mes": "北京信息科技大学网络实践创新联盟",
"interest_mes": "游泳/跑步/撸码",
"travel_mes": "新建、青海、西安、重庆",
"avatar": "/media/avatar/pjhubs.jpg",
"created_time": 1533305916,
}
    }
}
```


3.2 外部接口

硬件接口:

server 集群采用微服务架构，集群间通信采用基于 HTTP restful API 方式进行数据传输。

移动设备通信采用基于运营商的蜂窝网络进行。

软件接口:

server 与 client 的通信全部同样基于 HTTP restful API 进行，client 中还会与 server 保持长连接通信，保证消息通知 push 的及时可达性。

在聊天模块中 client 将会与 server 进行 socket 连接通信。

3.3 内部接口

硬件接口:

server 集群统一采用基于 Ubuntu 16.04 版本，内存 1G，带宽 1M，系统盘 50 GB 资源进行配置。

client 基于安卓 8.0 平台进行开发，使用 java 8 为主要实现语言，Kotlin 为辅助语言进行 app 开发。

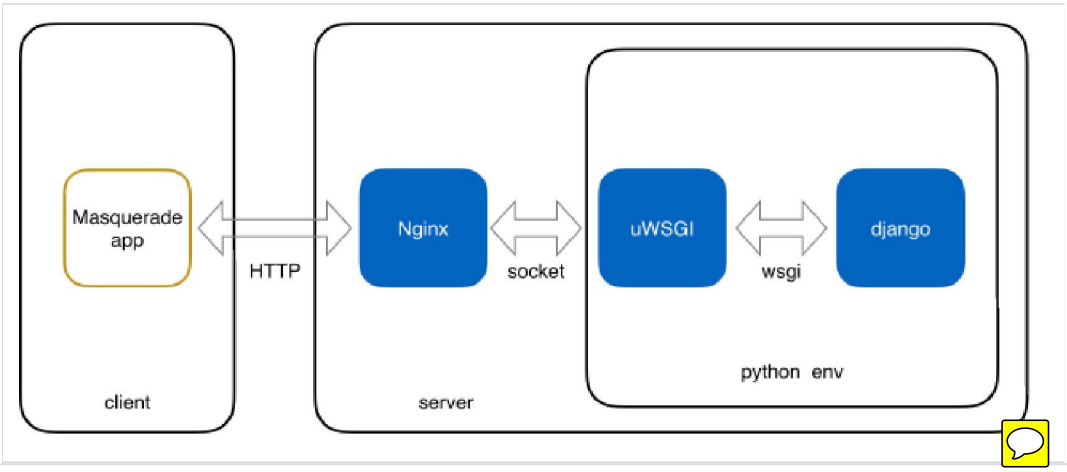
软件接口:

server 采用 django 框架且基于 MVC 模式进行搭建 API 服务，client 采用 MPV 模式进行开发。

4 运行设计

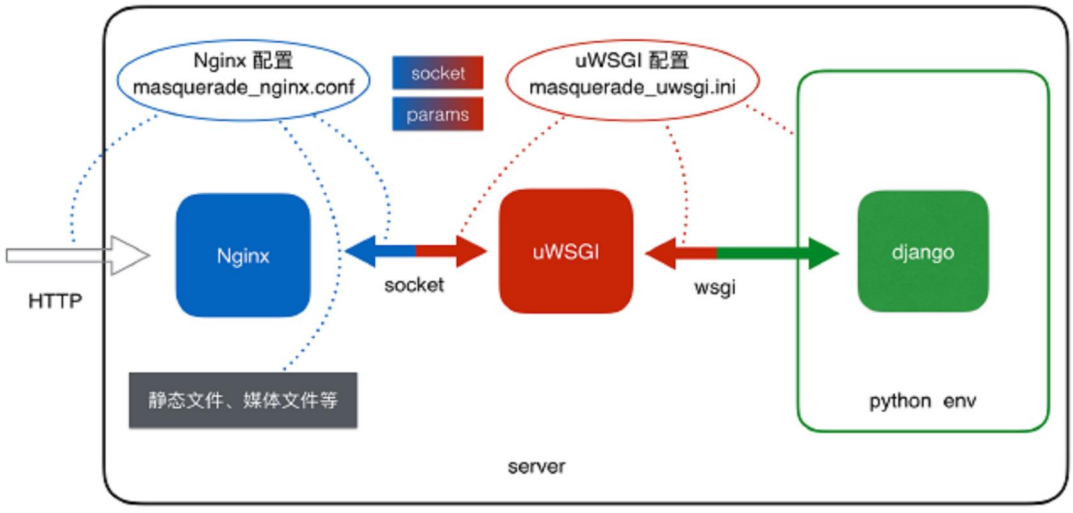
4.1 运行模块组合:

当客户端上的Masquerade运行行行获取数据时,实际上是客户端和服务端的web服务产生生了了一一个HTTP 协议的通信,客户端就会看到返回的数据.django是运行行行在python环境中的，而而且我们把它封装在了了一一个python的虚拟环境中,为达到和其它模块不不产生生冲突。uWSGI模块使用用wsgi与django进行行行连接通信，使用用socket套接字和Nginx进行行行通信。



4.2 运行控制:

当HTTP协议来了,先到达Nginx,Nginx通过它的配置使用用socket连接到uWSGI,uWSGI通过配置连接到了了python的运行行行环境，uWSGI要和Nginx进行行行通信需要给一一个套接字位置和参数表。当然，严格来说应该是python的运行行行环境和真实环境都需要安装uWSGI,在此省略略。



4.3 运行时间:

系统的运行时间基本可以达到用户所提出的要求

4.4 自由表述:

注:要求先部署完测试环境,且工工程文文件夹位于当前用用户目目录下。

增加软件:Nginx服务器器,uWSGI应用用协议服务模块。
nginx能够每次重启服务器器时都自自动重启，但是uWSGI不不行行行，没找到一一个合适的方方法，只能先把启动uWSGI服务的命令放到服务器器启动文文件中。
Masquerade API并没有去除HTTP状态码，并且也没有对HTTP状态码有任何的二二义处理理，客户端需要自自行行行处理理特殊报错，如500、404 等。
为了了防止止API被hack，在客户端上推荐所有请求都加上一一个随机hash，可以是根据当前时间戳加其它特殊字符，该字段服务端接口口不不做处理理。

5 系统数据结构设计

5.1 逻辑结构设计要点:

文章点赞

参数	类型	是否必须	描述	示例
content_type	String	是	blog	"blog"
object_id	String	是	文章id	"2"
is_like	String	是	是否点赞	"true"

获取某个用户点赞过的文章

请求

采用 `POST` 方法，地址为 `/like/getLikeBlogs`

参数	类型	是否必须	描述	示例
page	String	否，默认为第一页，一页返回10条文章	文章分页	"2"

删除文章

请求

采用 `POST` 方法，地址为 `/blog/deleteBlog`

参数	类型	是否必须	描述	示例
blog_id	String	是	文章id	"3"

创建评论

请求

采用 `POST` 方法，地址为 `/comment/createComment`

参数	类型	是否必须	描述	示例
content_type	String	是	给文章评论为blog；给评论回复为comment	"blog"/"comment"
content_id	String	是	内容 id	"2"
content	String	是	评论内容实体，不限长度，但前端最好做内容限制	"好文章！+1"
parent_id	String	否	如果需要对文章进行评论，不需要该字段；如果是对评论进行回复，该字段为被回复评论的 id	"3"

5.2 物理结构设计要点：

用户表结构：

Field	Type	Null	Key	Default	Extra
uid	varchar(10)	NO	PRI	NULL	
phoneNumber	varchar(11)	NO		NULL	
password	varchar(32)	NO		NULL	
openid	varchar(50)	NO		NULL	
nick_name	varchar(15)	NO		NULL	
slogan	varchar(50)	NO		NULL	
work_mes	varchar(20)	NO		NULL	
interest_mes	varchar(20)	NO		NULL	
travel_mes	varchar(20)	NO		NULL	
created_time	datetime(6)	NO		NULL	
last_updated_time	datetime(6)	NO		NULL	

文章表设计：

Field	Type	Null	Key	Default	Extra
id	int(11)	NO	PRI	NULL	auto_increment
content	longtext	NO		NULL	
created_time	datetime(6)	NO		NULL	
last_updated_time	datetime(6)	NO		NULL	
is_deleted	int(11)	NO		NULL	
masuser_id	varchar(10)	NO	MUL	NULL	

评论表设计：

Field	Type	Null	Key	Default	Extra
id	int(11)	NO	PRI	NULL	auto_increment
object_id	int(10) unsigned	NO		NULL	
content	longtext	NO		NULL	
comment_time	datetime(6)	NO		NULL	
is_deleted	int(11)	NO		NULL	
content_type_id	int(11)	NO	MUL	NULL	
masuser_id	varchar(10)	NO	MUL	NULL	
parent_id	int(11)	YES	MUL	NULL	
root_id	int(11)	YES	MUL	NULL	

```
+-----------------+-----------------+------+-----+---------+----------------+
```

5.3 数据结构与程序的关系：

用户；
在用户相关接口中，我们对用户模型进行了实例化，并对每一个用户通过 UUID 生成了唯一 hash 值，替代原先设计采用手机号为 uid 的弊端。

文章：
在文章相关接口，我们对筛选出符合条件的文章进行分页，通过客户端自定义 pageSize 进行文章数据自动化的、可伸缩的控制。

评论：
在获取文章对应评论接口中，我们参考了市场现有产品的评论设计，遵循了“盖楼”的常用设计，通过对评论添加冗余字段 parent_id 和 type 来达到索引父文章/评论和是对文章进行评论还是对评论进行回复。

5.4 自由表述：

用户这块最开始是直接继承了 django 的用户模块，这个模块有个利弊分明的地方，如果我们是做 web 开发，用 django 提供的用户模块是最方便不过了，但如果只是用 django 做 API 服务，太重，冗余字段和比较“僵硬”的约束，比如没法得知 django 的密码加密方式，前后端没法一致，客户端是没有所谓的cookie 和 session 管理（web壳子除外），如果非得要硬上也不是不行，但是这要付出很多额外的时间，没法做到 django 天然支持的 web 开发之便利,还有其它一些觉得用起来比较棘手的地方。当然，这些约束仁者见仁智者见智。最终的做法是自建了一个 MasUser 模型，需要提前考虑后续开发要接入的其它用户权限认证方式，比如QQ、微信、微博等的第三方登录权限 token。
在用户的权限认证这一块，因为没有采用 django 自带的用户模型，也就导致了没法用其提供的默认权限认证，遂直接全部抛弃，直接用了最粗暴的方法做了权限认证。自定义了一套 token 生成规则，当然，从本质上说还是基于 django-cache 的二次封装，我所做的只是重新制定了一套 token 生成的规则，其余的一概未动。token 存储是基于 mysql 做的数据库缓存，我知道数据库缓存实在是不好，在性能问题上得不到很好的解决，但这就算是一个验证吧，看看数据库缓存能有多大的问题。如果真的压不住了或者其它情况，推荐直接上 Redis。

6 系统出错处理设计

6.1 错误处理

出错信息：

程序在运行时主要会出现两种错误：
1、由于输入信息，或无法满足要求时产生的错误，称为软错误。
2、由于其他问题，如网络传输超时等，产生的问题，称为硬错误。
对于软错误，须在分享动态及其评论操作成功判断及输入数据验证模块由数据进行数据分析，判断错误类型，再生成相应的错误提示语句，送到输出模块中。
对于硬错误，可在出错的相应模块中输出简单的出错语句，并将程序重置。返回输入阶段。

补救措施：

1.启用备用数据库：当主数据库发生故障或发生其他数据丢失的情况，启动备用数据库；
2.硬件方面，使用稳定的设备；

6.2 系统维护设计：

维护方面主要为对服务器上的数据库数据进行维护。可使用 SQL SERVER 的数据库维护功能机制。例如，定期为数据库进行Backup，维护管理数据库死锁问题和维护数据库内数据的一致性等。