# 几点说明

1. 基本上 每个post的json中都有token ,encrypttime两个键，它的值是这样

encrypttime 是token生成的时间，也就是发送请求的时间，精确到分，采用标准格式

不够位数就补0，如2014-11-24 18:55 2014-01-01 01:01‘

后台会先获取当前时间（精确到分）与encrypttime 比对，~~若不等，认为超时，返回的msg置为3，若没超时，则将“songzimingdb$”和token相连接后进行SHA1 加密，然后与token比对，若相等则证明为合法请求，否则为非法请求，返回msg为1或其他非0数~~

若msg 为0 ，则其余字段有意义，若msg 为非0，证明有错误出现，此时一定会有一个info键值对，用于打印错误信息，其余字段不保证有意义。

2. 对于每一个回传的json 一般都有一个msg键，它的值至少有2个，0表示成功，1表示失败，只要不是0，都是非正常返回，具体错误码如果需要的话自行定义，但是不能占用0和1. 前台处理的时候先看msg，这个键是一定有的，只要不是0，后面的键值对就不看了，直接对话框报错处理。

3. 对于创建医院、添加科室、添加医生 和配置医院、配置科室、配置医生的区别在于：前者是Insert 后者是Update，Insert 过程中允许一些传过去的键值对的值是空。，所以后台插入的时候注意判断一下，if(val==null || val=’’) 怎么插，是插空还是插default值具体分析。

4. User 表增加一个字段 Amount real default 0 存储余额

User 表增加 Sex int(1) not null 0 为女 1 为男

User 表和Admin 表均应该存在一个FailTime字段和LastLogInTime字段（datetime类型）

Admin 增加Mail字段

5. 应该存在一个脚本每天定时执行，检测Reservation 表中的Reservation\_Time 和当前时间对比，若已经过期，从Reservation 表删除，并插入到History\_Reservation表

6. 请严格参照接口信息，因为可能有的参数是大写，有的参数是小写，有的英文拼写不对，但是请不要擅自更改，严格按照最新的借口信息规范文档

7. 关于 Province 和Area 表应该内置，因为量太大，咱们这样 ，Province 就把34个省插进去，Area 只把北京的这些地区（昌平区、海淀区。Etc 10多个）插进去。

8. 关于查医院、查科室、查管理员、查医生，post 都多加了一个start 和size ，要求后台把查到的符合条件的内容，limit start，size 打包后发给前台，同时return中增加了一个total

即，后台要把符合条件的总数饭给前台，于是，对于这几个接口，请求发送后，其实后执行两个sql查询，

select count(1) from… where condition,获取total ，

select \* from … where condition limit start,size,获取内容

9.

Doctor\_Level

Hospital\_Level

均为int(2)，所以

Doctor\_Level

1 一级医师

2 二级医师

3 三级医师

Hospital\_Level

31 三级甲等

32 三级乙等

33 三级丙等

21 二级甲等

22 二级乙等

23 二级丙等

11 一级甲等

12 一级乙等

13 一级丙等

后台只管取和存数字，前台负责转换

10，增加表

CREATE TABLE `Doctor\_Time` (

`Doctor\_ID` INT NOT NULL ,  
`Duty\_Time` INT( 2 ) NOT NULL

);

Duty\_Time 的值为以下固定的几个值

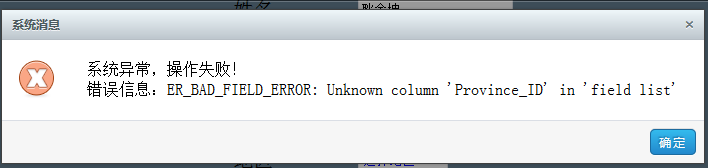
11,12,21,22,31,32,41,42,51,52,61,62,71,72

分别代表周一到周日的上下午，因为医生坐班的最小单位是半天

相应的配置医生信息、查看医生详情、添加医生需要变动，同时添加接口

Find\_Doctor\_By\_Condition

# 注册



post destination： host/Register

{

token

encrypttime

Birthday

UserName

Identity\_ID

Location

Sex

Password (已经做了加密处理，只负责插入）

Phone

Mail

}

Return(json)

{

msg:

Info: "0" or "1" 0 is ok 1 is fail

~~User\_ID:~~

}

User 表插入一条记录，

isChecked 初始0

Appointment\_Limit 初始为3

Credit\_Rank 初始为3

# 查询信息

post destination： host/Find\_Hospital

{

token

encrypttime

Hospital\_Level 医院等级

Area\_ID 和地区

start 起始 ！！！Attention

size 个数 ！！！Attention

}

return

{

msg: 0 or 1

content:

[

{

Hospital\_ID

Hospital\_introduction

Hospital\_Location

},

...

]

}

post destination： host/Find\_Doctor

{

token encrypttime

Depart\_ID

start

size

}

return

{

msg: 0 or 1

content:

[

{

Doctor\_ID

Doctor\_Name

Doctor\_Fee

Doctor\_Limit

Doctor\_Major

}

...

]

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

# 重置密码

User填写Identity\_ID,Admin填写Admin\_Name,然后前台通过Find\_User\_By\_Identity\_ID和

Find\_Admin\_By\_Admin\_Name,两个附加接口查到相应信息（接口见最后）

~~post destination： host/Check\_Match\_Name\_Mail 检查填写的邮箱和用户名是否匹配~~

~~{~~

~~token~~

~~encrypttime~~

~~type 0 is Admin 1 is user~~

~~UserName~~

~~Mail~~

~~}~~

~~return~~

~~{~~

~~msg: 0 or 1 0 is ok(send reset link to the user's mail ) 1 is not match (Mail and Username)~~

~~}~~

重置链接

和更改密码合并

~~域地址 + resetpwd.php？userid= &\*\*\*\*\*\*（100位 随机）~~

~~resetpwd.php 即为重置密码的页面，用户在上填写重置密码~~

~~Admin 和User 采用不同的url~~

~~post destination： Reset\_Pwd\_User.js~~

~~{~~

~~token encrypttime~~

~~Identity\_ID~~

~~Password~~

~~}~~

~~return~~

~~{~~

~~msg:~~

~~Info: 0 is ok~~

~~}~~

~~post destination： Reset\_Pwd\_Admin.js~~

~~{~~

~~token encrypttime~~

~~Identity\_ID~~

~~Password~~

~~}~~

~~return~~

~~{~~

~~msg:~~

~~Info: 0 is ok~~

~~}~~

# 登录系统

User 和Admin 采用不同url

post destination： host/LogIn\_User

{

token

encrypttime

Identity\_ID

Password

}

return

{

msg: 0 is ok, 1 is not match, 2 is locked (连续失败登录三次）

UserID

UserName

Identity\_ID

Credit\_Rank

Appointment\_Limit

Sex

Birthday

Location

Phone

Mail

Amount

Province\_ID

Province\_Name

Area\_ID

Area\_Name

}

注意： 如果登录成功 ，再回传json 之前要将User表或Admin表中对应的FailTime 置为0

post destination： host/LogIn\_Admin

{

token

encrypttime

Admin\_Name

Password

}

注意： 如果登录成功 ，再回传json 之前要将User表或Admin表中对应的FailTime 置为0

Return

{

Admin\_ID

isSuper

}

{

msg: 0 is ok, 1 is not match, 2 is locked (连续失败登录三次）

Admin\_ID

isSuper

}

# 更改密码

post destination： host/UpdatePwd\_Admin

{

token

encrypttime

Admin\_ID

Password

}

return

{

msg: 0 is ok

}

post destination： host/UpdatePwd\_User

{

token

encrypttime

User\_ID

Password

}

return

{

msg: 0 is ok

}

# ~~更改个人信息（只有用户可以更改一部分个人信息，管理员不可以）~~

~~post destination： host/Update\_Individual\_Info~~

~~{~~

~~token~~

~~encrypttime~~

~~User\_ID~~

~~Birthday~~

~~Location~~

~~Phone~~

~~Mail~~

~~Province\_Name~~

~~Area\_Name~~

~~}~~

~~return~~

~~{~~

~~msg: 0 is ok, 1 is fail~~

~~}~~

# 查看预约

post destination： host/Check\_Reservation\_Simple

{

token

encrypttime

User\_ID

}

return

{

msg:

content:

{

{

Reservation\_ID

Reservation\_Time

Operation\_Time

Doctor\_Name

Reservation\_Payed

}

...

}

}

# 查看单条详情：

post destination： host/Check\_Reservation\_Detail

{

token

encrypttime

Reservation\_ID

}

return

{

msg:

Reservation\_Time

Reseration\_Symptom

Reservation\_Payed

Operation\_Time

Hospital\_Name

Hospital\_Location

Depart\_Name

Doctor\_Name

Doctor\_Level

Doctor\_Fee

}

# 查看历史

post destination： host/Check\_Reservation\_History\_Simple

{

token

encrypttime

startDate

endDate

start

size

}

return

{

msg:

content:

{

{

History\_Reservation\_ID

History\_Reservation\_Time

History\_Operation\_Time

Doctor\_Name

}

...

}

}

# 查看单条历史详情

post destination： host/Check\_Reservation\_History\_Detail

{

token

encrypttime

History\_Reservation\_ID

}

return

{

msg:

History\_Reservation\_Time

History\_Reservation\_Symptom

History\_Reservation\_Payed

History\_Operation\_Time

Hospital\_Name

Hospital\_Location

Depart\_Name

Doctor\_Level

Doctor\_Fee

}

# 预约医生(修改)

post destination： host/Reservation

{

token

encrypttime

Doctor\_ID

User\_ID

Reservation\_Time

Reseration\_Symptom

}

return

{

msg:

0 is ok 1 is fail

}

预约表插入一条记录，

Doctor 已经预约人数字段加1

# 查看是否支付

post destination host/Check\_PayState

{

token

encrypttime

Reservation\_ID

}

return

{

msg:0

Info: 0 is paied

}

# 充值

分成两个接口

查看余额

post destination host/ Check\_Cash

{

token

encrypttime

User\_ID

}

充值

post destination host/ In\_Cash

{

token

encrypttime

User\_ID

Amount //这个是充值金额而不是总金额

}

return

{

msg:0 or 1

}

# 支付挂号费

post destination host/ Pay\_Reservation

{

token

encrypttime

Reservation\_ID

Reservation\_PayTime

User\_ID

}

return

{

msg:0 or 1

Info：0 is paied, 1 is lack for money.

}

\*\*\*\*\*\*\*\*\*\*\*\*

打印预约单

打印挂号单

注销系统

我来处理

\*\*\*\*\*\*\*\*\*\*\*\*\*

# 审核注册

post destination host/Check\_Register

{

token

encrypttime

UserID

type=1 or 0

1 通过注册

}

type=1

return

{

msg:0 or 1

0 is okay,1 is fail

Mail

Phone

}

type=0

return

{

msg:0 or 1

0 is okay,1 is fail

}

# 管理预约信息

post destination host/Get\_Reservation\_Info

{

token

encrypttime

Hospital\_ID //!

start

size

}

return

{

msg:

content:

[

{

UserName

Doctor\_Name

Reservation\_ID

Reservation\_Time

Operation\_Time

Doctor\_Name

}

...

]

}

# 查看某个用户的当前预约记录

post destination host/Search\_By\_Identity

{

token

encrypttime:

Identity\_ID:

}

return

{

msg:

content:

{

{

UserName

Doctor\_Name

Reservation\_ID

Reservation\_Time

Operation\_Time

Doctor\_Name

}

...

}

}

# 取消预约

post destination host/Cancel\_Reservation

{

token

encrypttime:

Reservation\_ID

}

return

{

msg :0 or 1

0 is ok 1 is fail

}

别忘了检查支付状态，若已经支付，要退钱

# ~~调整用户信用等级~~

~~post destination host/Set\_CreditRank\_user\_ID~~

~~{~~

~~token~~

~~encrypttime:~~

~~User\_ID~~

~~Credit\_Rank~~

~~}~~

~~return~~

~~{~~

~~msg :0 or 1~~

~~}~~

# 创建医院

post destination host/Create\_Hospital

{

token

encrypttime:

Hospital\_Name

Hospital\_Level

Hospital\_Introduction

Hospital\_Location

Reservation\_Start\_Time

Reservation\_End\_Time

Province\_ID

Area\_ID

Hospital\_Picture\_url

}

以上字段可以有空的，后台只管插入

return

{

msg: 0 or 1

}

# 获取医院信息(获取某个普通管理员管理权限范围内的医院条目）

post destination host/Get\_HospitalInfo\_simple

{

token

encrypttime

Admin\_ID

}

return

{

msg:

content:

[

{

Hospital\_ID

Hospital\_Name

}

...

]

}

# 获取单个医院详细信息

post destination host/Get\_HospitalInfo\_detail

{

token

encrypttime

Hospital\_ID

}

return

{

msg:0 or 1

0 is okay 1 is fail

Hospital\_Level

Hospital\_Introduction

Hospital\_Name

Hospital\_Location

Reservation\_Start\_Time

Reservation\_End\_Time

Hospital\_Picture\_url

}

# 配置医院信息

post destination host/Set\_HospitalInfo

{

token

encrypttime

Hospital\_ID

Hospital\_Level

Hospital\_Introduction

Hospital\_Name

Hospital\_Location

Reservation\_Start\_Time

Reservation\_End\_Time

Province\_ID

Area\_ID

Hospital\_Picture\_url

}

return

{

msg: 0 or 1

}

# 添加科室

post destination host/Add\_Depart

{

token

encrypttime

Hospital\_ID

Depart\_Name

}

return

{

msg:

}

# 获取科室信息

post destination host/Get\_DepartInfo

{

token

encrypttime

Hospital\_ID

//以下两项可以为空，代表没有limit start，size限制，查出结果全部返回

start

size

}

return

{

msg:

total

content:[

{

Depart\_ID

Depart\_Name

},

...

]

}

# 获取医生信息

post destination host/Get\_DoctorInfo

{

token

encrypttime

Depart\_ID

start

size

}

return

{

msg:

total

content:

[

{

Doctor\_ID

Doctor\_Name

},

...

]

}

# 查看医生详情

post destination host/Get\_DoctorInfo\_detail

{

token

encrypttime

Doctor\_ID

}

return

{

msg: 0 or 1 0 is okay 1 is fail

Doctor\_Name

Doctor\_Level

Doctor\_Fee

Doctor\_Limit

Doctor\_Major

Duty\_Time://不止一个键值对，以数组存储

[

]

Doctor\_Picture\_url

}

# 添加医生

post destination host/Add\_Doctor

{

token

encrypttime:

Depart\_ID

Doctor\_Name

//以下各项可能为空

Doctor\_Level

Doctor\_Fee

Doctor\_Limit//在一个的单位工作时间内的挂号上限（即半天内接受的挂号数，因为我们分上午和下午）

Doctor\_Major

Duty\_Time://不止一个键值对，以数组存储，但也可能为空

[

]

Doctor\_Picture\_url

}

return

{

msg:0 or 1 0 is okay 1 is fail

}

# 配置医生信息

post destination host/Set\_DoctorInfo

{

token

encrypttime

Doctor\_ID

Depart\_ID

Doctor\_Name

Doctor\_Level

Doctor\_Fee

Doctor\_Limit

Doctor\_Major

Duty\_Time://不止一个键值对，以数组存储，但也可能为空

[

]

Doctor\_Picture\_url

}

return

{

msg:0 or 1 0 is okay 1 is fail

}

# 创建普通管理员账号

post destination host/Add\_Admin

{

token

encrypttime:

Admin\_Name

Password

}

return

{

msg:0 or 1

}

# 获取管理员信息

post destination host/Get\_AdminInfo

{

token

encrypttime

start

size

}

return

{

msg:

total

content:

{

{

Admin\_ID

Admin\_Name

},

...

}

}

# 查看管理员权限

post destination host/Get\_Privilege

{

token

encrypttime

Admin\_ID

}

return

{

msg:

content:

{

{

Hospital\_ID

Hospital\_Name

},

...

}

}

# 赋予权限

post destination host/Give\_Privilege

{

token

encrypttime

Admin\_ID

Hospital\_ID :[,,,] 不止一个

}

return

{

msg: 0 or 1

0 is ok 1 is fail

}

# 解除权限

post destination host/Del\_Privilege

{

token

encrypttime

Hospital\_ID [,,,]不止一个

Admin\_ID

}

return

{

msg: 0 or 1

0 is ok 1 is fail

}

# 删除账号

post destination host/del\_Admin

{

token

encrypttime

Admin\_ID: [,,,] 不止一个

}

Admin\_ID !

return

{

msg :0 or 1

}

# 增加的接口

## post destination host/ Find\_User\_By\_Identity\_ID

{

token

encrypttime

Identity\_ID

}

return

{

msg :0 or 1

User\_ID

UserName

Identity\_ID

Credit\_Rank

Appointment\_Limit

Sex

Birthday

Location

Phone

Mail

Amount

Province\_ID

Province\_Name

Area\_ID

Area\_Name

}

## post destination host/ Find\_Admin\_By\_Admin\_Name

{

token

encrypttime

Admin\_Name

}

Return

{

msg:

Admin\_ID,

Mail

isSuper

}

Get\_Province\_Info

## post destination host/ Get\_Province\_Info

{

token

encrypttime

}

Return

{

msg:

content

[

{

Province\_ID

Province\_Name

}

……

]

}

## post destination Get\_Area\_Info\_By\_Province\_ID

{

token

encrypttime

Province\_ID

}

Return

{

msg

content:

[

{

Area\_ID

Area\_Name

}

…

]

}

为了简化起见，查询医院合成一个，注意

Province\_ID

Area\_ID

Hospital\_Level

可以为空，后台需要检验，比如

If(Area\_ID==’’)

Select … from Hospital where Province\_ID=(有限定) and Area\_ID=‘\*’（没有限定）

And Hospital\_Level=（有限定）

懂我的意思吧

## post destination host/ Find\_Hospital\_By\_Condition

{

token

encrypttime

//以下三项可以为空，空代表无限制 即\*

Province\_ID

Area\_ID

Hospital\_Level

、、、、、、、、、、、、、、、、、

start

size

}

Return

{

msg:

total (满足条件的医院的总数也要，select count(1) where condition)

content:

[

{

Hospital\_ID

Hospital\_Name

Hospital\_Picture\_url

}

….

]

}

## Post destination host/ Get\_History\_Reservation\_For\_Flexigrid

因为采用了Flexigrid，它的post 和接受的格式是写死的，稍微改动就不显示，只能按照他来

Post{

|  |  |
| --- | --- |
| Page 这个 |  |
| qtype |  |
| query |  |
| rp |  |
| sortname |  |
| sortorder | } |

Post

{

page 这个就相等于 start

qtype 不用管

query 我把startDate 和endDate 放在这里用！分隔，后台自己解析，没办法，只能这么传，其他键值对过不去

rp 这就相当于size

sortname 按照某个字段排序

sortorder 正序还是逆序 asc 还是desc

}

例子

|  |  |
| --- | --- |
| page | 1 |
| qtype | Reservation\_ID |
| query | 2014-12-01 00:00:00!2014-12-01 23:59:59 |
| rp | 10 |
| sortname | Reservation\_ID |
| sortorder | asc |

返回

{

Total

from 把我传过去的start填过来

to start+rp-1

rows:[

{

id: 就是Reservation\_ID,

cell:

[

Reservation\_ID,

Reservation\_Time,

Operation\_Time,

Doctor\_Name,

State

//state 为1，表示成功成功预约，state为0 表示爽约

这个必须由后台解析后直接把汉字打包成json

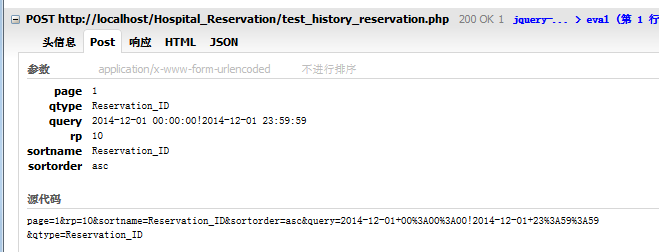
]

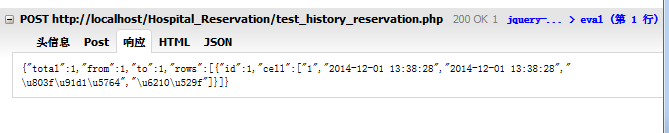
}这是一条记录

]

}







## Post destination host/ Get\_Hospital\_Number\_By\_Condition

{

token

encrypttime

//以下三项可能为空，空代表\*，全部

Province\_ID,

Area\_ID,

Hospital\_Level,

}

Return

{

msg

num

}

# 新增接口

## Post destination host/ Find\_Doctor\_By\_Condition

{

token

encrypttime

//以下四项可能为空，空代表\*，全部

Hospital\_ID,

Depart\_ID,

Doctor\_Level,

Duty\_Time

}

Return

{

msg

content:[

{

Doctor\_ID,

Doctor\_Name

}

]

}

## Post destination host/ Check\_Admin\_Repeat

{

token

encrypttime

Admin\_Name

}

Return

{

msg

isRepeat 0 is not repeat

1 is repeat

}

## ~~Post destination host/ Find\_Doctor\_State~~

~~{~~

~~token~~

~~encrypttime~~

~~Doctor\_ID~~

~~Duty\_Time~~

~~}~~

~~Return~~

~~{~~

~~msg~~

~~isfree 0 is free and can reserve~~

~~1 is~~

~~}~~

## Post destination host/ Find\_Doctor\_By\_Condition\_Free /注意这个接口和Find\_Doctor\_By\_Condition的区别，返回的Doctor应该是符合条件而且在目前是Duty\_Time（坐诊时间）还没有预约满的医生的集合，注意，我们的时间单位是半天，半天作为一个工作单位，Doctor的limit也是在一个工作单位内的预约数量，比如limit为3，而这个医生在周一上下午全天坐诊，那么就是他上午挂3个号，下午挂3个号

比如Duty\_Time 传过去的是41（周四下午），我们还会传一个用户传输的Reservation\_Time.那就意味着我想查看周四下午符合我的条件且未挂满的医生的集合

那么就是

Select Doctor\_ID,Doctor\_Name from Doctor

where (condition )

and Doctor\_ID in

(select Doctor\_ID from Reservation

where Reservation\_Time>=2014.12.11:12:01 and Reservation\_Time<=2014.12.11:23:59:59)

group by Doctor\_ID

having Count(Reservation\_ID)>Doctor.Limit)

这里的2014.12.11是通过用户传过去的Reservation\_Time 提取出年月日，再根据Duty\_Time是上午还是下午来判断是00:00:01~12：00:00 还是 12:00:01~23:59:59

目前看来只能这么搞

{

token

encrypttime

Hospital\_ID,

Depart\_ID,

Duty\_Time

Reservation\_Time

(前台会让用户选择一个想预约的时间点，同时根据这个时间计算出

Duty\_Time，比如用户选择是2014 12 11 16:55：00 ，前台会计算出

Duty\_Time是42）

}

Return

{

msg

content:[

{

Doctor\_ID,

Doctor\_Name

}

]

}

## Post destination host/Config\_User//根据User\_ID更改用户信息

{

token

encrypttime

User\_ID

UserName

Identity\_ID

Credit\_Rank

Appointment\_Limit

Sex

Birthday

Location

Phone

Mail

Province\_ID

Area\_ID

}

Return

{

msg

}

## Post destination host/Del\_Doctor

注意：只删除Doctor表、Doctor\_Time 表相关内容，不考虑Reservation 表的内容

{

token

encrypttime

Doctor\_ID

}

Return

{

msg

}

在更改密码之前应该先检查输入的旧密码是否正确，所以应该先能获取到旧密码

## Post destination host/Get\_Old\_Pwd\_User

{

token

encrypttime

User\_ID

}

Return {

msg

Password

}

## Post destination host/Get\_Old\_Pwd\_Admin

{

token

encrypttime

Admin\_ID

}

Return {

msg

Password

}

## Post destination host/Find\_User\_By\_Condition

{

token

encrypttime

//以下四项可以为空，空代表无限制

Area\_ID

isChecked

start

size

}

Return {

msg

content:[

{

User\_ID

User\_Name

Identity\_ID

}

]

}

# 宋子明部分

1.浏览器访问一个域名，设为stat\_host,抓取到该浏览器所处客户端的IP，回传json

{

Province\_ID,

Province\_Name,

redirect\_url

}

## 2.

Post destination start\_host/redirect

{

Province\_ID

}

重定向的url本来就应该完全暴露给用户，不存在窃取信息的概念，不考虑encrypttime加密校验

Return

{

redirect\_url

}

## 3

定时脚本：将Reservation表中的Reservation\_time小于现在时间（也就是已经过去了，不管成功还是失败，爽约还是不爽）的记录从该表删除，同时插入History\_Reservation,注意字段的对应关系，当然ID不用对应了，都是自增的。

模仿火车订票系统，每晚23点执行一次该脚本。

## 图片上传 Upload\_Picture\_Hospital

{

filetype jpg/png/gif/jpeg

oldurl：若为空，可以不管，否则，证明该医生/医院的照片已经存在，本次操作是覆盖而不是创建，你可以选择使用oldurl，或者删除该照片和链接，新建一个url出来，反正最后回传给我一个url

PicturetoUpload：以二进制形式存储的一段字节流

}

Return

{

msg:

Hospital\_Picture\_url

}

## Upload\_Picture\_Doctor

{

filetype jpg/png/gif/jpeg

oldurl：若为空，可以不管，否则，证明该医生/医院的照片已经存在，本次操作是覆盖而不是创建，你可以选择使用oldurl，或者删除该照片和链接，新建一个url出来，反正最后回传给我一个url

PicturetoUpload：以二进制形式存储的一段字节流

}

Return

{

msg:

Doctor\_Picture\_url

}