v0.1

1、将审核成功率改为成功数（分工3）

2、删除了新增存证、新增交易中请求参数的时间戳（分工1）

3、删除了新增存证中的filepath、evidenceId（分工1）

4、增加了预约公证（分工1）

5、公证申请记录查询，增加了organizationId（分工2）

v0.2

1、在证据表的查询中增加了按文件大小查询（分工2）

2、统计生成的三张表不输入时间戳，排名生成先不做，只有排名查询 ，再加一个公证机构排名查询（分工3）

3、排名查询时可选择按公证数量或公证成功数排序（分工3）

3、acceptFlag 1通过，0不通过；decryptFlag 1解密，0不解密

4、接口参数由accept\_flag、。。。改为acceptFlag，decryptFlag和timeFlag

5、新增公证员统计时间查询、公证机构统计时间查询（分工2）

6、新增了公证员可预约公证集合，然后把3可预约、待处理、已处理合并为了一个接口（分工2）

7、所有用户交易查询新增对用户的模糊匹配，新增对交易对象的精确查询(分工2)

8、transactionStatus从交易表移到证据表了，大家对应的接口记得要变一下(分工1，分工2)

v0.3

1、修改了公证机构查询的接口（分工2）

2、现在查询可以不传所有参数了

3、公证员排名查询、公证员统计查询、公证机构统计查询请求部分加入参数decryptFlag，调整返回字段（分工3）

# 主要格式

请求格式：param1=xxx，param2=xxx

返回格式：

* status：true\false
* message：一个字符串
* data：一个json，下面的接口都是这个json对象

对于所有参数，如果某些框用户没有选，就一律设成-1或“none”

**标\*的是需要与区块链交互获取区块链ID的，该ID不会作为前端参数传入**

# 用户端接口

* 用户注册

**请求：**

地址：/user/regist

参数：

* username
* password
* phoneNumber
* idCard
* email
* sex

**返回：**{ publicKey }

* 用户信息完善

**请求：**

地址：/user/regist

参数：

* userId
* newPassword
* phoneNumber
* idCard
* email
* sex

**返回：**无

* 用户登录

**请求：**

地址：/user/login

参数：

* username
* password

**返回：**userId

* \*新增存证：用户上传证据

**请求：**

地址：/user/addEvidence

参数：

* userId
* evidenceType
* evidenceName
* file

**返回：**区块链ID

* 在线查看证据文件

**请求：**

地址：/user/watchFile

参数：

* evidenceId

**返回：file**

* 取证服务：返回对应证据的文件（不要）

**请求：**

地址：/user/getEvidenceFile

参数：

* evidenceId

**返回：file**

* \*公证申请：为某个存证申请公证

**请求：**

地址： /user/ notarReq

参数：

* userId
* evidenceId
* organizationId
* notarizationType

**返回：**{ notarizationType:xxx, notarizationMoney: xxx}

（申请完公证需要缴费询问）



后台为用户生成公私钥对



* \*公证缴费：为某次公证申请缴费

**请求：**

地址： /user/ notarPay

参数：

* userId
* evidenceId
* transactionPeople （organizationId）
* notarizationMoney

**返回：** transaction的一行（transactionId，transactionMoney，transactionPeople，transactionType，transactionStatus，transactionTime，userRemains）

* \*充值：积分商店充值（交易对象为空/预定义）

**请求：**

地址： /user/charge

参数：

* userId
* transactionMoney

**返回：**transaction的一行（transactionId，transactionMoney，transactionPeople，transactionType，transactionStatus，transactionTime，userRemains）

* \*转赠：积分商店转赠

**请求：**

地址： /user/give

参数：

* userId
* transactionPeople
* transactionMoney

**返回：**transaction的一行（transactionId，transactionMoney，transactionPeople，transactionType，transactionStatus，transactionTime，userRemains）

* \*提现：积分商店提现（交易对象为空/预定义）

**请求：**

地址： /user/withdraw

参数：

* userId
* transactionMoney

**返回：**transaction的一行（transactionId，transactionMoney，transactionPeople，transactionType，transactionStatus，transactionTime，userRemains）

* \*购买存储空间：积分商店购买存储空间（交易对象为空/预定义）

**请求：**

地址： /user/memPay

参数：

* userId
* transactionMoney
* storageSize

**返回：**transaction的一行（transactionId，transactionMoney，transactionPeople，transactionType，transactionStatus，transactionTime， storageSize，userRemains）

* 公证申请记录查询：查询某个用户的公证记录

**请求：**

地址： /user/ notarRecord

参数：

* evidenceId
* userId **必填**
* usernameWildcard **必为”none”**
* evidenceName
* notarizationStatus **必为(“1”,”2”,”3”,”4”,”none”)之一**
* notarizationType
* paymentStatus
* evidenceType
* decryptFlag
* notarizationMoneyUpper
* notarizationMoneyFloor
* notarizationStartTimeStart
* notarizationStartTimeEnd
* notarizationEndTimeStart
* notarizationEndTimeEnd
* notaryId
* notaryNameWildcard
* organizationId
* organizationNameWildcard

**返回：**{list：一个list对象}，list中每个元素包含evidence中的一行

* 交易记录查询：查询用户的交易记录

**请求：**

地址： /user/ transQuery

参数：

* transactionId
* transactionBlockchainId
* transactionType
* transactionTimeStart
* transactionTimeEnd
* blockchainTimeStart
* blockchainTimeEnd
* transactionMoneyFloor
* transactionMoneyUpper
* decryptFlag
* userId **必填**
* usernameWildcard **必为“none”**
* transactionPeople
* transactionPeopleCipher **必为“none”**

**返回：**{list：一个list对象}，list中每个元素包含evidence中的一行（transactionId、transactionType、transactionTime、transactionMoney、userRemains、transactionStatus）

* 存证列表查询：返回证据列表

**请求：**

地址：/user/evidenceQuery

参数：

* evidenceId
* userId **必填**
* usernameWildcard **必为空**
* fileSizeFloor
* fileSizeUpper
* evidenceName
* evidenceType
* evidenceTimeStart
* evidenceTimeEnd
* blockchainTimeStart
* blockchainTimeEnd
* notarizationStatus
* evidenceBlockchainId
* decryptFlag

**返回：**

# 公证员接口

* 公证员注册

**请求：**

地址：/notar /regist

参数：

* notaryName
* password
* phoneNumber
* idCard
* email
* sex organizationId
* notarizationType

**返回：**{ publicKey }

* 公证员信息完善

**请求：**

地址：/notar /regist

参数：

* notaryId
* password
* phoneNumber
* idCard
* email
* sex organizationId
* notarizationType

**返回：**无

* 公证员登录

**请求：**

地址：/notar /login

参数：

* username
* password

**返回：**userId

* 公证记录查询：查询公证员可预约/可处理/已处理的公证记录

**请求：**

地址： /notar/ notarRecord

参数：

* decryptFlag
* notaryId
* dealType 0:可预约/1:待处理/2:已处理

**返回：**{list：一个list对象}，list中每个元素包含evidence中的一行

* 预约公证：审核公证申请

**请求：**

地址： /notar / appoint

参数：

* evidence\_id
* notary\_id

**返回：**无data

把evidence对应的公证状态由“待公证”改为“审核中”，然后填上公证员ID

* 审核通过/不通过：审核公证申请

**请求：**

地址： /notar / audit

参数：

* evidence\_id
* notary\_id
* accept\_flag 1通过，0不通过

**返回：**无data

# 公证管理员接口

* 公证申请记录查询：查询所有用户的公证记录

**请求：**

地址： /aut/ notarRecord

参数：

* evidenceId
* userId
* usernameWildcard
* evidenceNameWildcard
* notarizationStatus **必为(“1”,”2”,”3”,”4”,”none”)之一**
* notarizationType
* paymentStatus
* evidenceType
* decryptFlag
* notarizationMoneyUpper
* notarizationMoneyFloor
* notarizationStartTimeStart
* notarizationStartTimeEnd
* notarizationEndTimeStart
* notarizationEndTimeEnd
* notaryId
* notaryNameWildcard
* organizationId
* organizationNameWildcard

**返回：**{list：一个list对象}，list中每个元素包含evidence中的一行

* 交易记录查询：查询所有用户的交易记录

**请求：**

地址： /aut / transQuery

参数：

* transactionId
* transactionBlockchainId
* transactionType
* transactionTimeStart
* transactionTimeEnd
* blockchainTimeStart
* blockchainTimeEnd
* transactionMoneyFloor
* transactionMoneyUpper
* decryptFlag
* userId
* usernameWildcard
* transactionPeople （可以在明文也可在密文下查询，详情咨询文溢）
* transactionPeopleCipher

**返回：**{list：一个list对象}，list中每个元素包含evidence中的一行（transactionId、transactionType、transactionTime、transactionMoney、userRemains、transactionStatus）

* 存证列表查询：返回所有证据列表

**请求：**

地址：/aut /evidenceQuery

参数：

* evidenceId
* userId
* usernameWildcard
* fileSizeFloor
* fileSizeUpper
* evidenceNameWildcard
* evidenceType
* evidenceTimeStart
* evidenceTimeEnd
* blockchainTimeStart
* blockchainTimeEnd
* notarizationStatus
* evidenceBlockchainId
* decryptFlag

**返回：**{evidenceList：{evidenceId：xxx，evidenceName：xxx，blockchainTime：xxx，evidenceType：xxx，fileHash：xxx，evidenceBlockchainId ：xxx，organizationName：xxx，notarizationStartTime：xxx，notarizationStatus ：xxx，notarizationMoney ：xxx}}

* 用户查询

**请求：**

地址： /aut / userQuery

参数：

* userId
* usernameWildcard
* phoneNumberWildcard
* idCard
* emailWildcard
* remainsFloor
* remainsUpper
* storageSpaceFloor
* storageSpaceUpper
* hasUsedStorageFloor
* hasUsedStorageUpper
* sex
* decryptFlag

**返回：**{list：一个list对象}，list中每个元素包含notary一行的内容

* 公证员查询

**请求：**

地址： /aut / notaQuery

参数：

* notaryId
* notaryNameWildcard
* phoneNumberWildcard
* jobNumberWildcard
* idCard
* emailWildcard
* sex
* organizationId
* notarizationType
* decryptFlag

**返回：**{list：一个list对象}，list中每个元素包含notary一行的内容

* 公证机构管理员查询

**请求：**

地址： /aut / autmanQuery

参数：

* autManId
* autNameWildcard
* phoneNumberWildcard
* jobNumberWildcard
* idCard
* emailWildcard
* sex
* organizationId
* decryptFlag

**返回：**{list：一个list对象}，list中每个元素包含notary一行的内容

* 公证机构查询

**请求：**

地址： /aut /orgaQuery

参数：

* organizationId
* organizationIdNameWildcard
* addressWildcard
* phoneNumberWildcard
* legalPeopleWildcard
* emailWildcard

**返回：**{list：一个list对象}，list中每个元素包含notary一行的内容

* 公证员统计生成

**请求：**

地址： /aut / notaStasGen

参数：无

返回：无data

* 公证员统计查询

**请求：**

地址： /aut / notaStasQue

参数：

* timeFlag
* decryptFlag

**返回：**{list：一个list对象}，list中每个元素包含notary\_statistics一行的内容

* 公证机构统计生成

**请求：**

地址： /aut / orgStasGen

* 参数：无

**返回：**无data

* 公证机构统计查询

**请求：**

地址： /aut / orgStasQue

参数：

* timeFlag
* decryptFlag

**返回：**{list：一个list对象}，list中每个元素包含organization\_statistics一行的内容

* 公证员排名查询

**请求：**

地址： /aut / rankStasQue

参数：

* timeFlag
* decryptFlag
* sort 0:按公证数量排序/1:按公证成功数排序

**返回：**{list：一个list对象}，list中每个元素包含rank一行的内容

# 其他接口

* 公证机构查询：查询所有公证机构

**请求：**

地址： /aut /orgaQuery

参数：下列参数全设为”none”

* organizationId
* organizationIdNameWildcard
* addressWildcard
* phoneNumberWildcard
* legalPeopleWildcard
* emailWildcard

**返回：**{orglist：一个list对象}，list中每个元素是{ organization\_id:xxx, organization\_name: xxx}

* 公证类型查询：查询所有公证类型

**请求：**

地址： / noTypeQuery

参数：无

**返回：**{orglist：一个list对象}

* 交易类型查询：查询所有交易类型

**请求：**

地址： / tranTypeQuery

参数：无

**返回：**{orglist：一个list对象}

* 存证类型查询：查询所有证据类型

**请求：**

地址： / eviTypeQuery

参数：无

**返回：**{orglist：一个list对象}

* 公证费用查询：查询所有类型公证所需费用

**请求：**

地址： / notPayQuery

参数：无

**返回：**{ notarizationType:xxx, notarizationMoney: xxx}

* 公证员统计时间查询：查询什么时候对公证员进行了统计

**请求：**

地址： / notStaTimeQuery

参数：无

**返回：**{ 一个时间标记的列表}

* 公证机构统计时间查询：查询什么时候对公证机构进行了统计

**请求：**

地址： / orgStaTimeQuery

参数：无

**返回：**{ 一个时间标记的列表}

# 同态加密部分

我的建议是这样的：我们只考虑对 用户表、交易表、存证公证表 中的某些字段进行加密。

加密数据分为数值类型与字符类型

对数值类型的查询只有一种，qw1<x<qw2

对字符类型的查询也只有一种，输入一个字符串，最多包含2个通配符。eg. A\*BC\*D

* 数值查询

**请求：**

地址： / numSearch

参数：

* field\_name
* table\_name
* qw1
* qw2
* decrypt\_flag

**返回：**{list：一个list对象}，list中每个元素根据table\_name分别为user\transaction\evidence

* 字符查询

**请求：**

地址： / wildcardSearch

参数：

* field\_name
* table\_name
* wildcard
* decrypt\_flag

**返回：**{list：一个list对象}，list中每个元素根据table\_name分别为user\transaction\evidence