供应链接口规范

修改记录

|  |  |  |  |
| --- | --- | --- | --- |
| 日期 | 作者/修改人 | 描述 | 审核人 |
| 2021-08-21 | 杨德榜 |  |  |
| 2021-08-23 | 李幸 | 新增业务交互时序流程 |  |

目录

[一、 接口清单 3](#_Toc12059)

[1.1 时序图 3](#_Toc21023)

[二、 使用授信额度申请 4](#_Toc30339)

[2.1接口规则 4](#_Toc23161)

[2.2接口请求参数 4](#_Toc3811)

[2.3签名生成规则 5](#_Toc5553)

[2.4接口返回内容 6](#_Toc1623)

[2.5返回格式 7](#_Toc4515)

[2.6状态码定义 7](#_Toc21172)

[2.7数据加密规则 7](#_Toc13974)

[3.1.1 时间格式文本 8](#_Toc26077)

[三 授信额度查询 8](#_Toc5810)

[3.1接口说明 8](#_Toc6631)

[3.2请求参数 9](#_Toc6678)

[3.3接口响应 9](#_Toc23376)

[3.4返回示例 9](#_Toc29889)

# 业务说明

本文档为第三方平台接入产融平台的说明文档，描述具体的产融平台产品流程，接口清单、接入步骤、技术接入细节等方面。

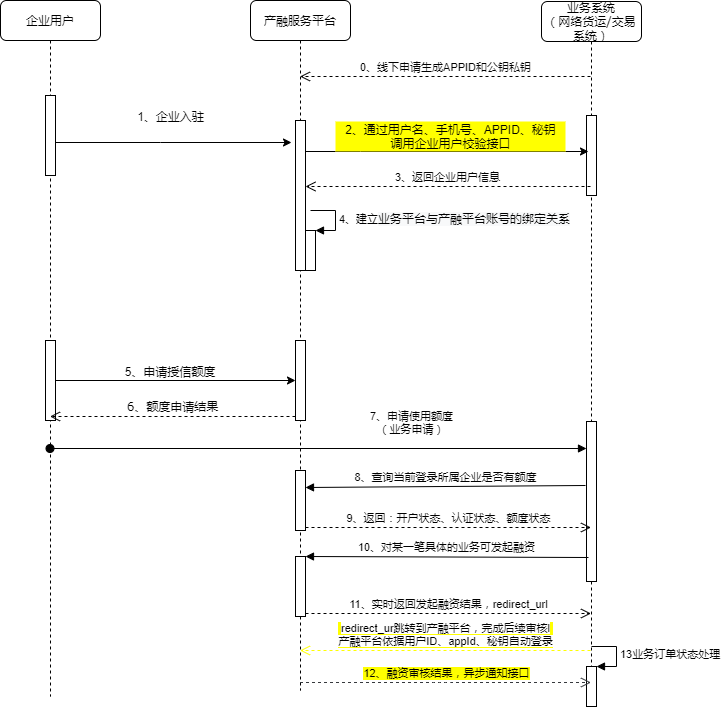
# 产品流程

1. 企业用户注册登录账号
2. 企业用户入驻申请
3. 企业用户绑定业务平台
4. 企业用户申请授信额度
5. 企业用户在业务平台上发起融资申请

# 接入流程

1. 线下邮件或其他有效的方式发起申请接入产融平台申请
2. 产融平台后台生成APPID+公钥私钥
3. 第三方平台依据获得到的APPID以及本接口文档相关要求开发实现系统对接
4. 双方测试环境对接联调

## 系统交互流程时序图



# 通讯规范与安全

## 协议规范

接口调用请求均采用HTTP POST方式，支持http和https协议，请求和响应报文均为JSON格式，UTF-8编码，请求URL全部为小写英文字符。

## 通用请求参数

POST请求参数分为header和body两部分，header为公共参数，每个接口的header部分参数格式均相同，body为接口参数，根据接口的不同，参数可能不一样。

header参数说明：

|  |  |
| --- | --- |
| 字段名 | 说明 |
| reqTime | 请求时间戳，毫秒数 |
| sign | 请求签名串，详见：[4.3 签名生成规则](#_签名生成规则) |
| version | 版本号，1.0 |

body参数说明（以工程基础信息查询接口为例）：

|  |  |
| --- | --- |
| 字段名 | 说明 |
| projectId | 工程ID，数组结构 |
| size | 每页记录数 |
| page | 页码 |

请求路径示例：

[https://xxxxx/aip/project/baseinfo](http://const.caihcloud.com/aip/project/baseinfo)

请求参数示例：

{

“header”:

{

“appId”:”12588”,

“reqTime”:”515132998”,

“version”:”1.0”,

“sign”:”6E5001070F264EC2B4127DC51CE198D4”

},

“body”:

{

“projectId”:[“100”,”101”,”102”],

“size”:”10”,

“page”:”1”,

}

}

## 接口完整性校验

产融平台通过验证签名来保证请求的真实性和数据的完整性

### 签名生成规则

1. 把header中的参数值按参数名（除去sign字段）升序排序后拼接；
2. 把body中的参数值按参数名升序排序后拼接；
3. 最后生成的源串 = head参数拼接串 + body参数拼接串 + secret（大小写敏感），以1.2的请求参数为例，最后生成的源串为”12588515132998100101102XXXXXX”，”XXXXXX”为sercret；
4. 对生成的源串进行md5加密，以1.2的请求参数为例，最后生成的加密串为”28b2914b85e5cdaa79e107ee36b94fb8”；
5. 服务端验签过程直接用提交参数中的sign字段跟服务端生成的签名进行字符串比较（大小写不敏感）；
6. 字符串中包含空格的处理方式：若空格出现在字符串前后，则将前后空格字符进行裁减，若空格出现在字符串中间，则不处理。处理规则 类似String.trim()；
7. 空值处理规则：如果某个待拼接的值为null对象，请使用空串””进行拼接，不要拼接”null”字符串；
8. sign签名生成工具，请参阅以下md5方法（以Java为例）；
9. 示例代码（以Java为例）：

// header

String appId = “12588”;

String reqTime = “515132998”;

String version = “1.0”;

// body

String projectId = “100101102”; // 如果是数组或者List，按元素的字符串默认排序规则拼成一整个字符串，此示例由100、101、102这3个元素拼成

String page = “1”;

String size = “10”;

// sercret

String sercret = “XXXXXX”;

String sign = signByKey(sb.toString(),"UTF-8",RSAUtils.getPrivateKey(privateKey));

String sign = SignUtils.md5(appId + reqTime + version + page + projectId + size + sercret);//注意拼接顺序

### 接口签名java demo

|  |
| --- |
| static char hexDigits[] = {'0', '1', '2', '3', '4', '5', '6', '7', '8', '9', 'A', 'B', 'C', 'D', 'E', 'F'};  static String MD5 = "MD5";//加签方式：MD5  /\*  \* @Author boy  \* @Description 数据签名  \* @Date 2019/8/31 1:57 PM  \* @Param [data, key]  \* @return java.lang.String  \*/  public static String sign(String data, String key) throws Exception {  //得到明文的字节数组  byte[] btInput = (data + key).getBytes();  // 创建一个提供信息摘要算法的对象(MD5摘要算法)  MessageDigest messageDigest = MessageDigest.getInstance(MD5);  // 使用指定的字节更新摘要  messageDigest.update(btInput);  // 得到二进制的密文  byte[] encryptData = messageDigest.digest();  // 把密文转换成十六进制的字符串形式  String encryptDataStr = bytesToHex(encryptData);  return encryptDataStr;  }  /\*  \* @Author boy  \* @Description 验签  \* @Date 2019/8/31 1:57 PM  \* @Param [data, key, sign][明文数据,签名key,接收到的签名]  \* @return boolean  \*/  public static boolean verifySign(String data, String key, String sign) throws Exception {  //调用加签方法，看加签后的签名是否和接收到的一致  String encryptData = sign(data, key);  if (encryptData.equals(sign)) {  return true;  } else {  return false;  }  }  /\*  \* @Author boy  \* @Description 将byte数组转化为16进制字符串  \* @Date 2019/8/31 1:58 PM  \* @Param [bytes]  \* @return java.lang.String  \*/  public static String bytesToHex(byte[] bytes) {  int k = 0;  char[] hexChars = new char[bytes.length \* 2];  for (int i = 0; i < bytes.length; i++) {  byte byte0 = bytes[i];  hexChars[k++] = hexDigits[byte0 >>> 4 & 0xf];  hexChars[k++] = hexDigits[byte0 & 0xf];  }  return new String(hexChars);  }  //示例  public static void main(String[] args) throws Exception {  Map<String, String> hashMap = new HashMap<>();  String data = "你好！MD5!";  String key = "1234567890abcdef";  String dataSign = MD5Utils.sign(data, key);  hashMap.put("data", data);  hashMap.put("dataSign", dataSign);  System.out.println("明文:" + hashMap.get("data"));  System.out.println("签名：" + hashMap.get("dataSign"));  System.out.println("验签结果：" + MD5Utils.verifySign(data, key, dataSign));  } |

## 敏感数据安全

技术开发人员在调用安全级别较高的接口（如：用信）时，会使用到公私进行数据的加密传输，以保证数据安全和防止敏感信息的泄露。

### 公私、私钥的生成规则：

1. 申请方式

线下向产融平台的产品负责人申请，生成公私钥对，

生成后通过邮件的方式把公钥发给第三方。

1. 使用流程

第三方拿到公钥后，对包含敏感数据接口，在进行数据请求和传输之前，先进行加密，使用公钥作为key生成验签，请求到本系统，本系统根据私钥，对请求的数据进行解密校验。

### 数据加密规则



|  |
| --- |
| /\*\*  \* RSA公钥加密  \*  \* @param data 加密字符串  \* @param publicKey 公钥  \* @return 密文  \* @throws Exception 加密过程中的异常信息  \*/  public static String encrypt(String data, String publicKey) throws Exception {  // base64 编码的公钥  byte[] decoded = Base64.getDecoder().decode(publicKey);  RSAPublicKey pubKey = (RSAPublicKey) KeyFactory.getInstance(ALGO).generatePublic(new X509EncodedKeySpec(decoded));  // RSA加密  Cipher cipher = Cipher.getInstance(ALGO);  // 公钥加密  cipher.init(Cipher.ENCRYPT\_MODE, pubKey);  byte[] b=data.getBytes(StandardCharsets.UTF\_8);  byte[] b1 = null;/\*\*执行加密操作\*/  for (int i = 0; i < b.length; i += 101){  byte[] doFinal = cipher.doFinal(ArrayUtils.subarray(b,i,i +101));  b1 = ArrayUtils.addAll(b1,doFinal);  }  BASE64Encoder encoder=new BASE64Encoder();  return encoder.encode(b1);  } |

## 接口返回内容

1. 返回的报文格式统一为JSON；
2. 返回的数据会根据接入账号数据权限的不同，进行不同级别的数据脱敏处理；
3. 返回参数中值为空的字段不输出。

### 返回格式

|  |  |
| --- | --- |
| 字段名 | 说明 |
| code | 状态码 |
| message | 请求结果描述 |
| data | 数据实体，JSON格式，有记录返回时输出 |

### 状态码定义

（仅供参考，以实际接口返回为准）

|  |  |
| --- | --- |
| 状态码范围 | 说明 |
| 0000 | 成功 //其他均为失败 |
| 0001 | 参数错误 |
| 0010 | 加签失败 |
| 0011 | 验签失败 |
| 9998 | 业务错误 |
| 9999 | 系统错误 |

返回报文示例：

{

"code":"00000",

"message":"成功",

"data":

[{

"name":"张三",

"age":"20"

},

{

"name":"李四",

"age":"16"

}]

}

## 其它说明

### 时间格式文本

时间格式文本统一采用“yyyy-MM-dd HH:mm:ss”格式，小时采用24小时制。

举例：2002-07-01 14:30:00

# 接口清单

## 测试环境接口地址

**[http://10.17.116.5:18897/](http://10.17.116.5:8002/)**

下表是所有的接口清单：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 序号 | 接口名 | 服务方 | 调用方 | 接口描述 |
| 1 | 查询业务系统企业用户信息 | 业务系统 | 供应链平台 | 调用第三方业务系统接口获取到第三方系统企业用户信息 |
| 2 | 授信额度查询 | 供应链平台 | 第三方 | 第三方调用接口，业务平台返回信息 |
| 3 | 申请使用融资额度 | 供应链平台 | 第三方 | 第三方调用接口，业务平台返回信息 |
| 4 | 融资申请结果异步通知 | 业务系统 | 供应链平台 | 调用第三方业务系统接口把融资申请的最终结果通知业务系统做后续业务流程处理 |

## 查询业务系统企业用户

### 接口说明

接口 URL：[https://ip:port/enterprise/erification](https://ip:port/pay/batch/commit)

描述：查询业务系统企业用户

请求方式：Post

### 请求参数

|  |  |  |  |
| --- | --- | --- | --- |
| **字段** | **含义** | **类型** | **备注** |
| unifiedCode | 统一社会信用代码 | String | 统一社会信用代码 |
| phoneNumber | 手机号 | String | 手机号 |
| Account | 用户账号 | String | 用户账号 |
| Sign | 签名 | String | 请求签名串，详见：[签名生成规则](#_sign签名生成规则) |

### 接口响应

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 类型 | 是否为空 | 字段说明 | 备注 |
| code | string | N | 返回码 | 0000为成功，其他为失败 |
| message | string | N | 返回说明 |  |

### 返回示例

JSON示例

{

”code”：”0000”,

”message”：“查询成功”

”result”：”账号存在 或账号不存在“

}

## 授信额度查询

### 接口说明

接口 URL：[https://ip:port/dockingOther](https://ip:port/pay/batch/commit)/querySupply

描述：开放对外接口，授信额度查询

请求方式：Post

### 请求参数

|  |  |  |  |
| --- | --- | --- | --- |
| **字段** | **含义** | **类型** | **备注** |
| unifiedCode | 统一社会信用代码 | String | 统一社会信用代码 |
| platformCode | 平台唯一code | String | 平台唯一code |
| Account | 用户账号 | String | 用户账号 |
| Sign | 签名 | String | 请求签名串，详见：[签名生成规则](#_sign签名生成规则) |

### 接口响应

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 类型 | 是否为空 | 字段说明 | 备注 |
| code | string | N | 返回码 | 0000为成功，其他为失败 |
| message | string | N | 返回说明 |  |
| authenticationAccountStatus | string | N | 账号认证状态 |  |
| existence | string | N | 是否存在账号 |  |
| lineCredit | string | N | 授信额度 |  |

### 返回示例

JSON示例

{

”code”：”0000”,

”message”：“查询成功”

”data”:{

"authenticationAccountStatus":"已认证",

"existence":"该账户已存在",

"lineCredit":"0"

}

}

## 申请使用授信额度

1. 流程

线下生成公私钥对，公钥给到第三方，第三方拿到公钥后，对数据进行加密，使用公钥作为key生成验签，请求到本系统，本系统根据私钥，对请求的数据进行解密校验。

### 接口说明

接口 URL：[https://ip:port/dockingOther](https://ip:port/pay/batch/commit)/receivingService

描述：开放对外接口，使用授信额度申请

请求方式：Post

加密方式：RSA算法

### 请求参数

POST请求参数为接口参数，根据接口的不同，参数可能不一样。

参数说明（以接收业务数据接口为例）：

|  |  |  |
| --- | --- | --- |
| 字段名 | 说明 | 类型 |
| platformCode | 平台唯一code | String |
| jsonData | 平台业务数据,公钥加密后的json字符串 | String //**json字符串** |
| timeRequest | 请求时间 | Date |
| userId | 用户ID | String |
| orderMoney | 订单金额 | BigDecimal |
| platformId | 平台ID | String |
| companyName | 企业名 | String |
| contractNumber | 接入平台的账号 | String |
| sign | 签名 | String |

请求路径示例：

[https://xxxxx/dockingOther/receivingService](http://const.caihcloud.com/aip/project/baseinfo)

请求参数示例：

{ “**platformCode**”:”1”,

“**jsonData**”:"**dataList**":[{"number":5914111507281122,"contractType":"签订合同","transportType":"陆运","unitWeight":5,"transportPrice":100.00,"transportTotal":100,"contractPrice":12000.00,"shipmentDate":"2020.11.13","latestDate":"2020.12.1","loadPlace":"某某地点","consignee":"张三","receiptPlace":"某某地点","carrier":"某某公司","shopCompany":"某某公司","transporter":"某某公司","information":"信息信息"}]}

“**timeRequest**”:”1”,

“**userId**”:”1”,

“**orderMoney**”:”1”,

“**platformId**”:”1”,

“**companyName**”:”1”,

“**contractNumber**”:”1”,

“**sign**”:”1”

}

**jsonData.dataList 参数**

**陆运运输**：运输总量、运输单价、合同金额、承运日期、卸货日期、装货地点、收货地点、运输物品、发票信息、签收单

|  |  |
| --- | --- |
| 字段名 | 说明 |
| transportType | 运输方式 |
| transportPrice | 运输单价 |
| transportTotal | 运输总量 |
| contractPrice | 合同金额 |
| shipmentDate | 承运日期 |
| latestDate | 卸货日期 |
| loadPlace | 装货地点 |
| receiptPlace | 收货地点 |
| carrier | 承运方 |
| shopCompany | 船公司（如有） |
| transporter | 运输物品 |
| information | 发票信息 |
| receipt | 签收单 |

**多式联运运输**：运输总量、运输单价、合同金额、承运日期、完成日期、装货地点、收货地点、运输物品、发票信息、水运单、签收单（如有）

|  |  |
| --- | --- |
| 字段名 | 说明 |
| transportType | 运输方式 |
| transportPrice | 运输单价 |
| transportTotal | 运输总量 |
| contractPrice | 合同金额 |
| shipmentDate | 承运日期 |
| latestDate | 完成日期 |
| loadPlace | 装货地点 |
| receiptPlace | 收货地点 |
| waterWaybill | 水运单 |
| transporter | 运输物品 |
| information | 发票信息 |
| receipt | 签收单 |

**现货交易平台：**买方单位、成交时间、交易模式、合同状态、结算方式、合同总量、预计合同金额、预付款金额、预付款时间、实际合同金额、手续费、交收状态、品名、实际交收价格、

提货明细：货品、仓库、库存、申请重量、品牌、等级、榨季、仓单状态、

合同交收流程：操作时间、交收结算信息

**jsonData.dataList 参数**

|  |  |
| --- | --- |
| 字段名 | 说明 |
| unit | 买方单位 |
| transactionTime | 成交时间 |
| transactionMode | 交易模式 |
| contractStatus | 合同状态 |
| settlementMethod | 结算方式 |
| totalAmount | 合同总量 |
| estimatedAmount | 预计合同金额 |
| advanceAmount | 预付款金额 |
| advanceTime | 预付款时间 |
| actualAmount | 实际合同金额 |
| handlingFee | 手续费 |
| deliveryStatus | 交收状态 |
| productName | 品名 |
| actualPrice | 实际交收价格 |
| operatingTime | 操作时间 |
| billingInfo | 交收结算信息 |
| dataList | 明细 |
| goodsType | 货品 |
| warehouse | 仓库 |
| stock | 库存 |
| applyWeight | 申请重量 |
| brand | 品牌 |
| grade | 等级 |
| season | 榨季 |
| warehouseStatus | 仓单状态 |

### 接口响应

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 类型 | 是否为空 | 字段说明 | 备注 |
| code | string | N | 返回码 | 0000为成功，其他为失败 |
| message | string | N | 返回说明 |  |
| orderId | string | N | 订单id | 操作成功生成的唯一订单ID |

### 返回示例

JSON示例

{

”code”：”0000”,

”message”：“操作成功”

”orderId”：“20210901555555”

}

## 融资申请结果异步通知

### 接口说明

接口 URL：[https://ip:port/financing](https://ip:port/pay/batch/commit)/results

描述：开放对外接口，授信额度查询

请求方式：Post

### 请求参数

|  |  |  |
| --- | --- | --- |
| **字段** | **类型** | **备注** |
| orderId | String | 订单号 |
| orderMoney | String | 订单金额 |
| loanAmount | String | 机构放款金额 |
| enterpriseName | String | 企业名称 |
| userId | String | 用户ID |
| unifiedCode | String | 统一社会信用代码 |
| Sign | String | 请求签名串，详见：[签名生成规则](#_sign签名生成规则) |

### 接口响应

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 类型 | 是否为空 | 字段说明 | 备注 |
| code | string | N | 返回码 | 0000为成功，其他为失败 |
| message | string | N | 返回说明 |  |
| data | Object | N | 返回数据 |  |

### 返回示例

JSON示例

{

”code”：”0000”,

”message”：“查询成功”

”data”:{

“financingResult”:”Success”|”FAILD”;

“financingMoney”:”123”;

“financingRate”: ”2.30%”

}

}