支付宝手机网站支付快速接入

第一步：创建应用并获取 APPID

先去蚂蚁金服开放平台（open.alipay.com），在开发者中心中创建登记您的应用，并提交审核，审核通过后会为您生成应用唯一标识（APPID），并且可以申请开通开放产品使用权限，通过 APPID 您的应用才能调用开放产品的接口能力。
在研发阶段通常采用沙箱环境，但仍需要实名认证才可以使用开放平台服务。步骤如下：

按提示操作即可。默认沙箱环境下会生成一个沙箱应用，其中 APPID、支付网关无法修改，直接使用，而应用公钥需要我们手动设置。如下图所示。

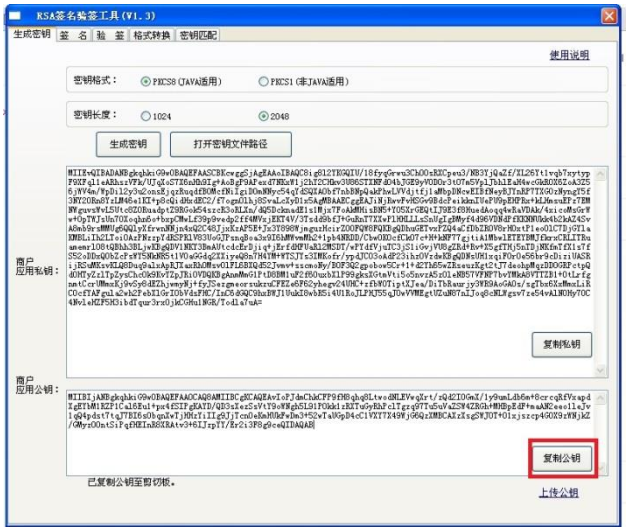
第二步：配置密钥

开发者调用接口前需要先生成 RSA 密钥，RSA 密钥包含应用私钥(APP_PRIVATE_KEY)、应用公钥(APP_PUBLIC_KEY）。生成密钥后在开放平台开发者中心进行密钥配置，配置完成后可以获取支付宝公钥(ALIPAY_PUBLIC_KEY)。

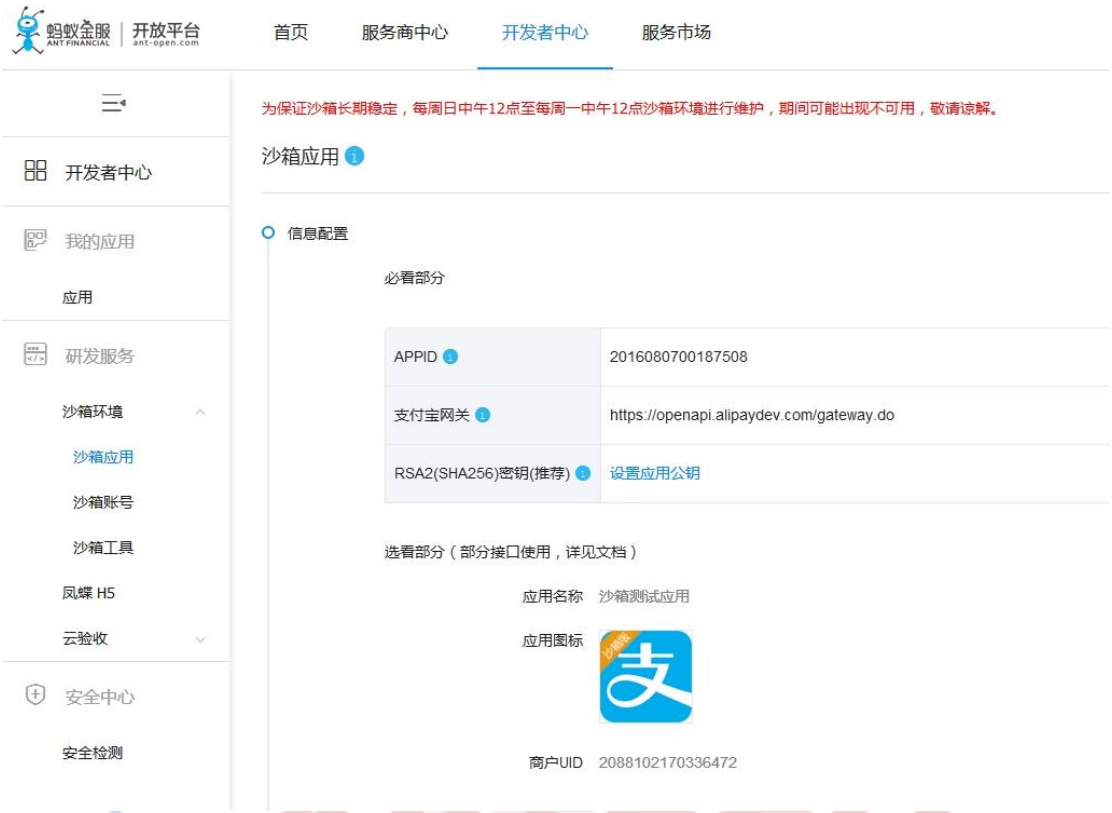
RSA 密钥生成工具下载地址：

http://p.tb.cn/rmsportal_6680_secret_key_tools_RSA_win.zip?spm=a219a.7629140.0.0.TUzjw4&file=rmsportal_6680_secret_key_tools_RSA_win.zip

运行效果如下图：



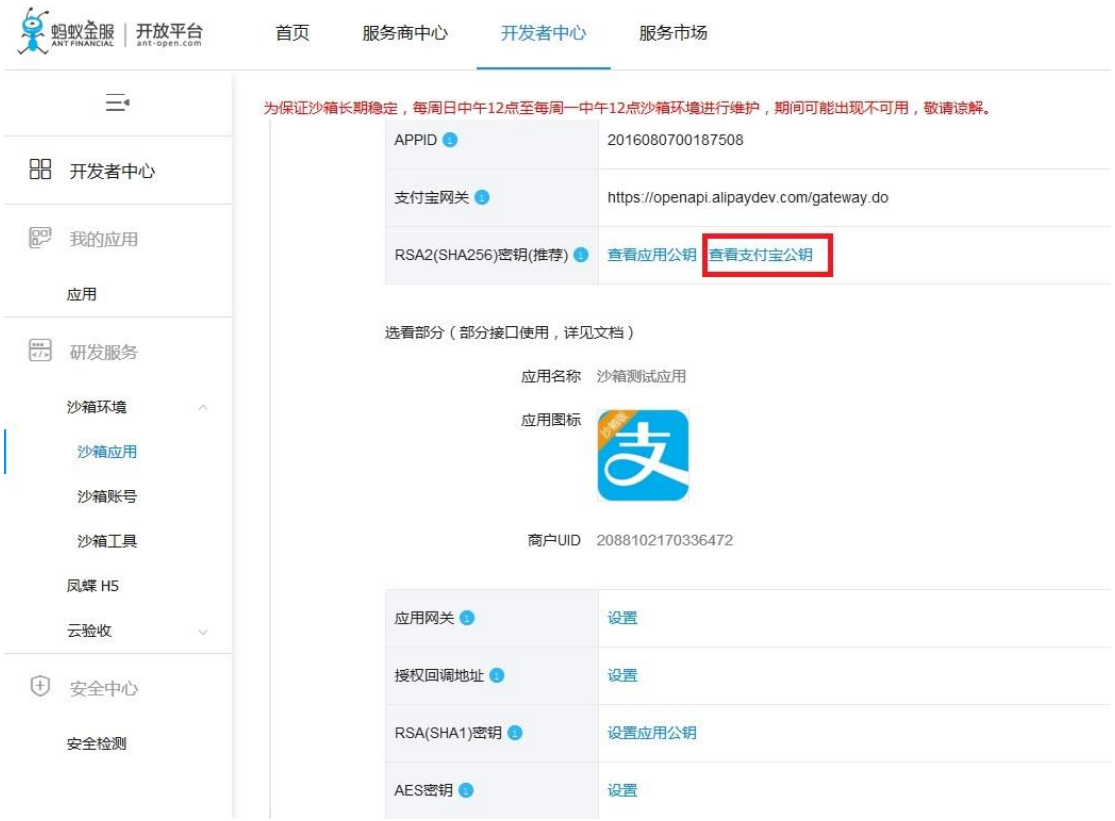
选择密钥长度 2048，点击生成密钥然后点击“复制公钥”，将公钥复制到沙箱应用中。商户应用公钥必须上传至沙箱应用配置中的应用公钥部分，私钥配置在 itriptrade 应用程序中。



点击“设置应用公钥”

点击“设置应用公钥”

拷贝刚刚复制的应用公钥，然后保存。



平台将自动生成支付宝公钥。此处应复制支付宝公钥以备用。如下图所示。

第三步：搭建和配置开发环境

1. 下载服务端 SDK（Java 版本）

https://doc.open.alipay.com/docs/doc.htm?spm=a219a.7629140.0.0.x8xDll&treeId=193&articleId=105910&docType=1

Java 版本 SDK：

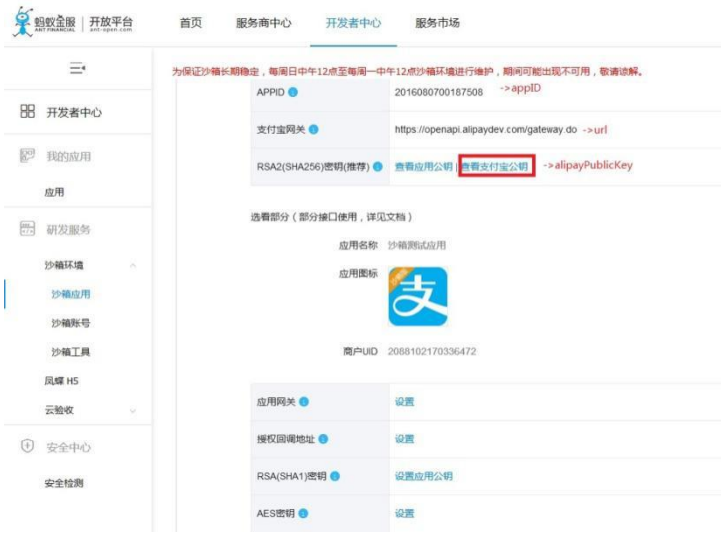
http://p.tb.cn/rmsportal_6680_alipay.trade.wap.pay-java-utf-8.zip?spm=a219a.7629140.0.0.mUA9CR&file=rmsportal_6680_alipay.trade.wap.pay-java-utf-8.zip

SDK 封装了签名&验签、HTTP 接口请求等基础功能。请先下载对应语言版本的 SDK 并引入您的开发工程。

2. 接口调用配置

```
<!-- 支付宝手机网站支付 -->
<bean class="cn.itrip.trade.config.AlipayConfig">
    <property name="appID" value="2016080600177878"/>
    <property name="rsaPrivateKey" value="MIIEvgIBADANBgkqhkiG9w0BAQEFAASCBKgwggSkAgEAAoIBAQ
    <property name="notifyUrl" value="http://101.200.141.234:8084/itriptrade/api/notify"/>
    <property name="returnUrl" value="http://101.200.141.234:8084/itriptrade/api/return"/>
    <property name="url" value="https://openapi.alipaydev.com/gateway.do"/>
    <property name="charset" value="UTF-8"/>
    <property name="format" value="json"/>
    <property name="alipayPublicKey" value="MIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEAxSl1
    <property name="logPath" value="/logs"/>
    <property name="signType" value="RSA2"/>
    <property name="paymentSuccessUrl" value="/itriptrade/success.jsp"/>
    <property name="paymentFailureUrl" value="/itriptrade/failure.jsp"/>
</bean>
```

其中 appID、url、alipayPublicKey 分别对应沙箱应用如下信息。



rsaPrivateKey 为密钥生成工具生成的商户应用私钥，应复制至此。

至此，配置沙箱环境已完成。

第四步：SDK 的使用

SDK 包说明

alipay-sdk-java*.jar————————————支付宝 SDK 编译文件 jar
alipay-sdk-java*-source.jar————————支付宝 SDK 源码文件 jar
commons-logging-1.1.1.jar————————SDK 依赖的日志 jar
commons-logging-1.1.1-sources.jar————SDK 依赖的日志源码 jar

注意

集成支付宝接口需要引入的文件是：
alipay-sdk-java*.jar
commons-logging-1.1.1.jar
若进一步了解代码实现请引入文件：
alipay-sdk-java*-source.jar
commons-logging-1.1.1-sources.jar
在 itrip-trade 应用中对支付功的实现详见：
cn.itrip.trade.controller.PaymentController.java

提交支付请求调用示例

```
// 超时时间 可空
String timeout_express = "2m";
// 销售产品码 必填
String product_code = "QUICK_WAP_PAY";
/**********************/
// SDK 公共请求类，包含公共请求参数，以及封装了签名与验签，开发者无需关注签名与验签
// 调用RSA签名方式
AlipayClient client = new DefaultAlipayClient(alipayConfig.getUrl(),
        alipayConfig.getAppID(), alipayConfig.getRsaPrivateKey(),
        alipayConfig.getFormat(), alipayConfig.getCharset(),
        alipayConfig.getAlipayPublicKey(), alipayConfig.getSignType());
AlipayTradeWapPayRequest alipay_request = new AlipayTradeWapPayRequest();

// 封装请求支付信息
AlipayTradeWapPayModel model = new AlipayTradeWapPayModel();
model.setOutTradeNo(WIDout_trade_no);
model.setSubject(WIDsubject);
model.setTotalAmount(WIDtotal_amount);
model.setTimeoutExpress(timeout_express);
model.setProductCode(product_code);
alipay_request.setBizModel(model);
// 设置异步通知地址
alipay_request.setNotifyUrl(alipayConfig.getNotifyUrl());
// 设置同步地址
alipay_request.setReturnUrl(alipayConfig.getReturnUrl());
// form表单生产
String form = "";
try {
    // 调用SDK生成表单
    form = client.pageExecute(alipay_request).getBody();
    System.out.println(form);
    response.setContentType("text/html;charset="
            + alipayConfig.getCharset());
    response.getWriter().write(form);// 直接将完整的表单html输出到页面
    response.getWriter().flush();
    response.getWriter().close();
} catch (AlipayApiException e) {
    // TODO Auto-generated catch block
    e.printStackTrace();
} catch (IOException e) {
    // TODO Auto-generated catch block
    e.printStackTrace();
}
```

导步通知调用示例

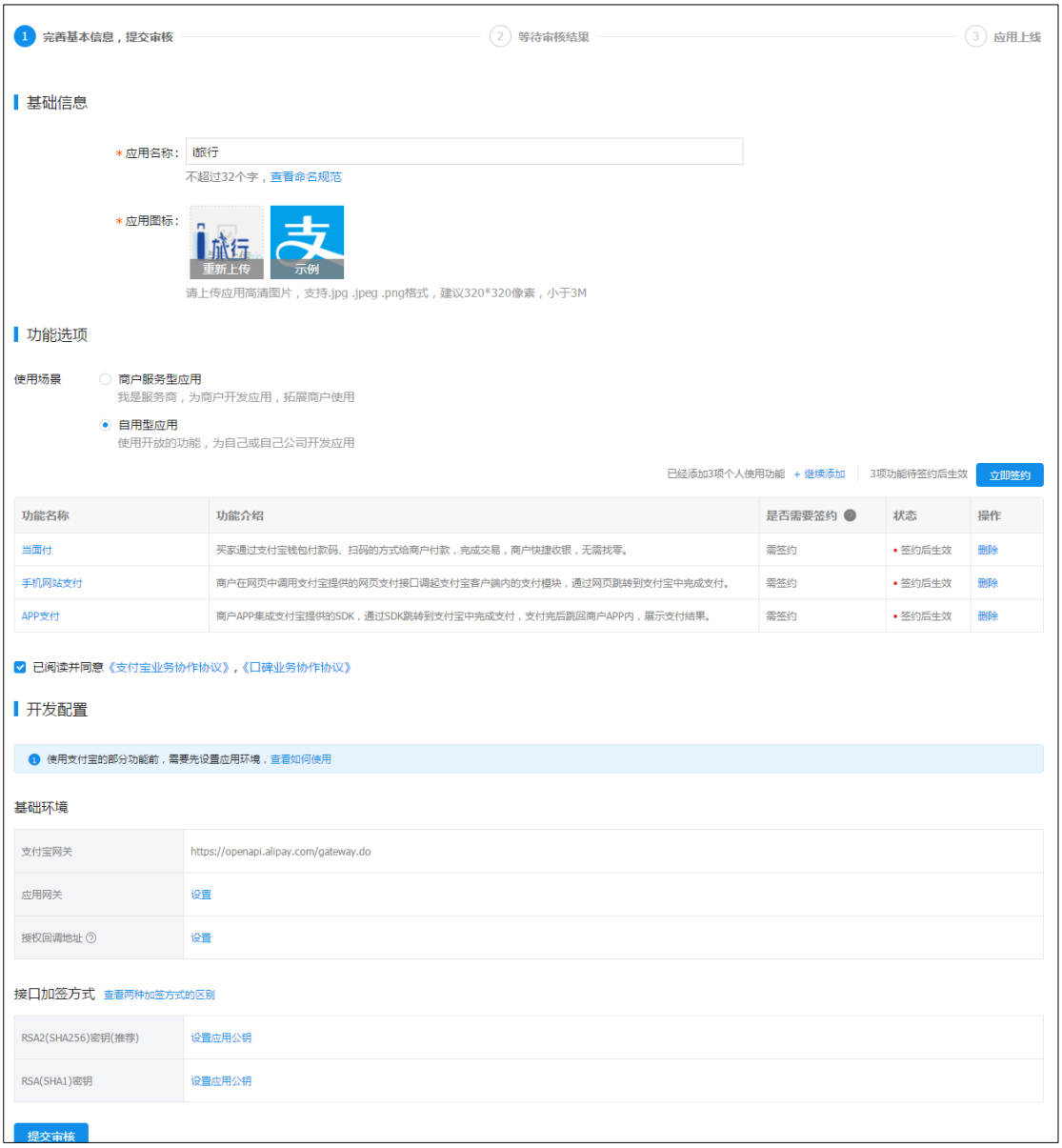
```
// 获取支付宝POST过来反馈信息
Map<String, String> params = new HashMap<String, String>();
Map requestParams = request.getParameterMap();
for (Iterator iter = requestParams.keySet().iterator(); iter.hasNext();) {
    String name = (String) iter.next();
    String[] values = (String[]) requestParams.get(name);
    String valueStr = "";
    for (int i = 0; i < values.length; i++) {
        valueStr = (i == values.length - 1) ? valueStr + values[i]
                : valueStr + values[i] + ",";
    }
    // 乱码解决，这段代码在出现乱码时使用。如果mysign和sign不相等也可以使用这段代码转化
    // valueStr = new String(valueStr.getBytes("ISO-8859-1"), "gbk");
    params.put(name, valueStr);
}
// 获取支付宝的通知返回参数，可参考技术文档中页面跳转同步通知参数列表(以下仅供参考)//
// 商户订单号
String out_trade_no = new String(request.getParameter("out_trade_no")
        .getBytes("ISO-8859-1"), "UTF-8");
// 支付宝交易号
String trade_no = new String(request.getParameter("trade_no").getBytes(
        "ISO-8859-1"), "UTF-8");
// 交易状态
String trade_status = new String(request.getParameter("trade_status")
        .getBytes("ISO-8859-1"), "UTF-8");

// 获取支付宝的通知返回参数，可参考技术文档中页面跳转同步通知参数列表(以上仅供参考)//
// 计算得出通知验证结果
// boolean AlipaySignature.rsaCheckV1(Map<String, String> params, String
// publicKey, String charset, String sign_type)
boolean verify_result = AlipaySignature.rsaCheckV1(params,
        alipayConfig.getAlipayPublicKey(), alipayConfig.getCharset(), "RSA2");

if (verify_result) {// 验证成功
    // ////////////////////////////////////////////////////////////////////////////////////////
    // 请在这里加上商户的业务逻辑程序代码

    //即时到账普通版，那么这时的交易状态值为： TRADE_FINISHED；如果是即时到账高级版，此时的交易状态值就为：TRADE_SUCCESS
    //收到TRADE_FINISHED请求后，这笔订单就结束了，支付宝不会再主动请求商户网站了；收到TRADE_SUCCESS请求后，后续一定还有
    if (trade_status.equals("TRADE_FINISHED")) {
        // 判断该笔订单是否在商户网站中已经做过处理
        // 如果没有做过处理，根据订单号（out_trade_no）在商户网站的订单系统中查到该笔订单的详细，并执行商户的业务程序
        // 请务必判断请求时的total_fee、seller_id与通知时获取的total_fee、seller_id为一致的
        // 如果有做过处理，不执行商户的业务程序
        if(!orderService.processed(out_trade_no))
        {
            orderService.paySuccess(out_trade_no, 2,trade_no);
        }
        logger.info("订单: "+out_trade_no+" 交易完成");
        // 注意:
        // 如果签约的是可退款协议，退款日期超过可退款期限后（如三个月可退款），支付宝系统发送该交易状态通知
        // 如果没有签约可退款协议，那么付款完成后，支付宝系统发送该交易状态通知。
    } else if (trade_status.equals("TRADE_SUCCESS")) {
        // 判断该笔订单是否在商户网站中已经做过处理
        // 如果没有做过处理，根据订单号（out_trade_no）在商户网站的订单系统中查到该笔订单的详细，并执行商户的业务程序
        // 请务必判断请求时的total_fee、seller_id与通知时获取的total_fee、seller_id为一致的
        // 如果有做过处理，不执行商户的业务程序
        if(!orderService.processed(out_trade_no))
        {
            orderService.paySuccess(out_trade_no, 2,trade_no);
        }
        logger.info("订单: "+out_trade_no+" 交易成功");

        // 注意:
        // 如果签约的是可退款协议，那么付款完成后，支付宝系统发送该交易状态通知。
    }

    response.getWriter().println("success"); // 请不要修改或删除

    // ////////////////////////////////////////////////////////////////////////////////////////
} else {// 验证失败
    orderService.payFailed(out_trade_no, 1,trade_no);
    response.getWriter().println("fail");
}
```

支付成功跳转调用示例

```
//获取支付宝GET过来反馈信息
Map<String,String> params = new HashMap<String,String>();
Map requestParams = request.getParameterMap();
for (Iterator iter = requestParams.keySet().iterator(); iter.hasNext();) {
    String name = (String) iter.next();
    String[] values = (String[]) requestParams.get(name);
    String valueStr = "";
    for (int i = 0; i < values.length; i++) {
        valueStr = (i == values.length - 1) ? valueStr + values[i]
                : valueStr + values[i] + ",";
    }
    //乱码解决，这段代码在出现乱码时使用。如果mysign和sign不相等也可以使用这段代码转化
    valueStr = new String(valueStr.getBytes("ISO-8859-1"), "utf-8");
    params.put(name, valueStr);
}

//获取支付宝的通知返回参数，可参考技术文档中页面跳转同步通知参数列表(以下仅供参考)//
//商户订单号

String out_trade_no = new String(request.getParameter("out_trade_no").getBytes("ISO-8859-1"),"UTF-8");

//支付宝交易号

String trade_no = new String(request.getParameter("trade_no").getBytes("ISO-8859-1"),"UTF-8");

//获取支付宝的通知返回参数，可参考技术文档中页面跳转同步通知参数列表(以上仅供参考)//
//计算得出通知验证结果
//boolean AlipaySignature.rsaCheckV1(Map<String, String> params, String publicKey, String charset, Strir
boolean verify_result = AlipaySignature.rsaCheckV1(params, alipayConfig.getAlipayPublicKey(), alipayConf

if(verify_result){//验证成功
    //提示支付成功
    response.sendRedirect(alipayConfig.getPaymentSuccessUrl());
}else{
    //提示支付失败
    response.sendRedirect(alipayConfig.getPaymentFailureUrl());
}
```


第五步：线上验收

在沙箱环境完成功能调试后，必须将支付宝网关、appid、应用私钥、支付宝公钥修改成正式环境的配置，并在蚂蚁正式环境进行完整的功能验收测试。
在开发者中心完善基本信息、提交审核即可。包括应用名称、图标、签约支付产品、开发配置。如下图：



其中签约时需要填写企业信息、经营信息、银行账户等信息，上传相关证件，并且需要单独审核验证，签约流程如下图：

提供资料如下图：

经营者信息

经营者姓名：[redacted]

身份证类型：二代身份证 临时身份证

身份证号：[redacted]

证件有效期：YYYY-MM-DD 长期

上传经营者身份证：个人信息页 国徽页

个体工商户信息 按照证书上的内容逐字填写

名称：输入名称 查找生僻字

注册号：输入注册号

单位所在地：请选择地区

如果找不到所在城市，可选择所在地区或上级城市

住所：输入住所

经营范围：输入经营范围

营业期限：YYYY-MM-DD 长期

上传营业执照：仅支持中国大陆地区的工商营业执照，且必须在有效期内。

格式要求：原件照、原件扫描件或复印件加盖公章的扫描件，支持jpg，jpeg，png，bmp格式文件，单个文件不超过 5 MB。

上传授权函：如果经营主体与申请主体不一致，请上传授权函。下载授权函模板

下一步