北京市医保电子凭证

医院端 app 对接规范 V1.0

版本号	修订人	修订时间	修订说明
V1.0	刘振华	2020-08-06	新建

北京市医保电子凭证 - 医院端 app 对接规范

1 概述

本规范用于说明《北京市医保电子凭证》-医院端 app 对接规范，用于支持业务系统研发设计人员进行对接研发和联调。

2 术语

术语/缩略语	说明
北京市医保电子凭证	指平台服务,负责北京通端、微信端、医院 app 端的出码
医院端 app	指各个医院自建的 app，本次在医院端接入电子凭证服务的载体
实名验证平台	北京市统一身份认证平台提供的校验用户实名身份的能力

3 流程介绍

本章节用于说医院端 app 与医保电子凭证服务及其各相关系统间的关系。

3.1 医院端 app 出码流程

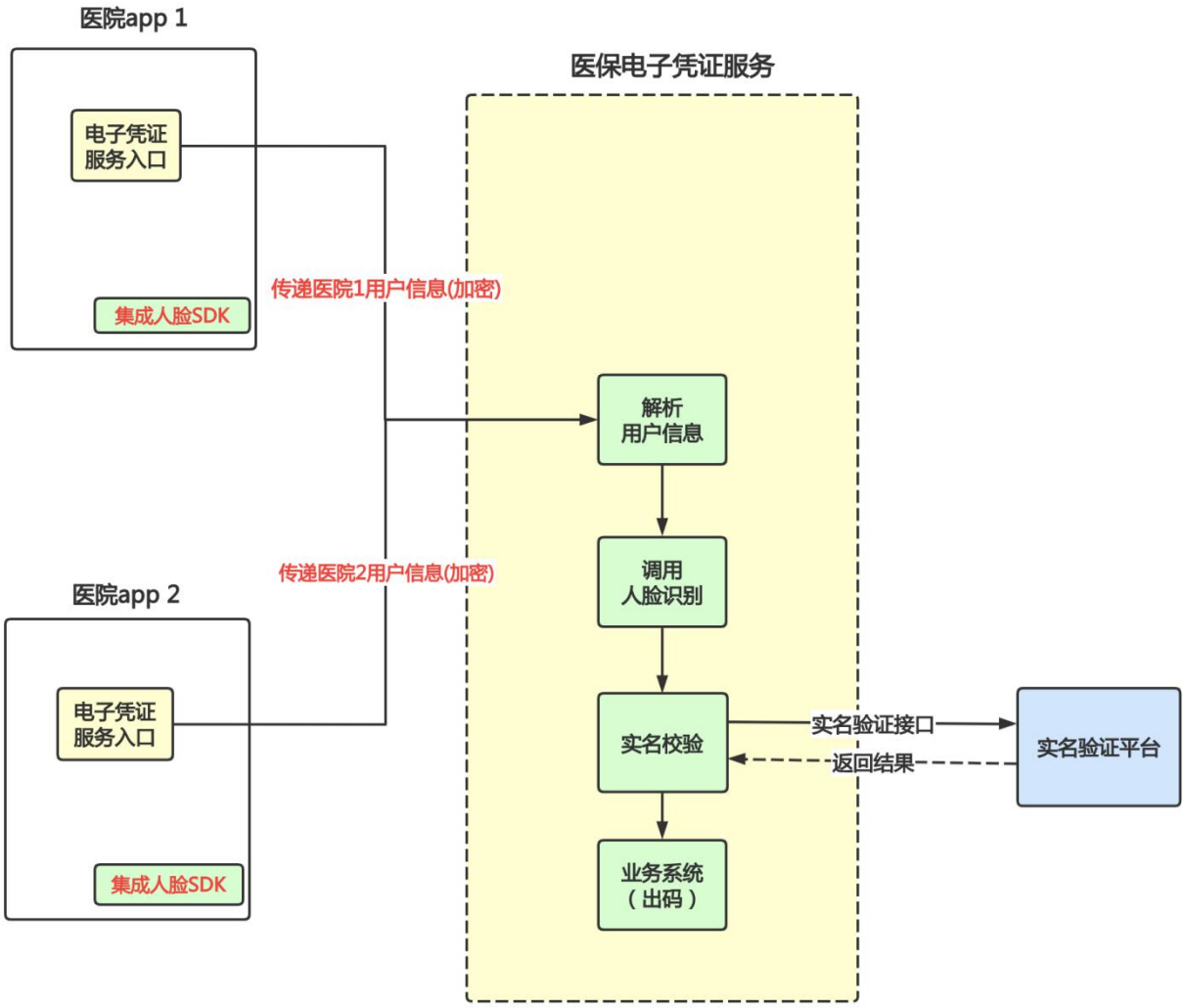

图 1 医院端 app 出码流程

4 对接方案

4.1 用户对接

4.1.1 流程说明

- 在医院端 app 上配置医保电子凭证服务的入口地址；
- 用户点击医保电子凭证服务入口；
- 医院端 app 在入口地址后面追加以下参数并跳转到医保电子凭证服务

例如：

原始地址：

XXX/homepage/hospital

追加参数后地址：

XXX/homepage/hospital?hospitalCode=1001&userName=XXX&cardType=01&cardNo=XXX&timestamp=1597028576255&sign=XXX

4.1.2 字段定义

参数	类型	是否必填	描述	是否需要加密
hospitalCode	String	是	医院编码	
userName	String	是	用户姓名	是
cardType	String	是	证件类型（01：身份证）	
cardNo	String	是	证件编码	是
timestamp	Long	是	时间戳	
sign	String	是	签名值	

4.1.3 密钥

线下为每个医院 app 提供医院编码和密钥（区分测试环境和生产环境）

4.1.4 鉴权方案

4.1.4.1 签名

所有非空字段，key 以升序排序，key 以=连接。Value 之间使用&连接。生成的待签名字符串使用 sha256 算法，以 appSecret 加盐生成签名。

示例：

请求参数：

```
{
    "hospitalCode": "1001",
    "userName": "XXX",
    "cardType": "01",
    "cardNo": "XXX",
    "timestamp": 1597028576255
}
```

待签名字符串：

hospitalCode=1001&userName=XXX&cardType=01&cardNo=XXX&timestamp=1597028576255

排序：

cardNo=XXX&cardType=01&hospitalCode=1001&timestamp=1597028576255&userName=XXX

追加密钥：

cardNo=XXX&cardType=01&hospitalCode=1001&timestamp=1597028576255&userName=XXX&appSecret=XXX

生成签名值：

```
Digester sha = new Digester(DigestAlgorithm.SHA256);
String sign = sha.digestHex(content);
```

签名结果添加到入参：

```
{
    "hospitalCode": "1001",
    "userName": "XXX",
    "cardType": "01",
    "cardNo": "XXX",
    "timestamp": 1597028576255,
    "sign": "XXX"
}
```

签名方法：

```
<!-- 引入依赖: -->
<dependency>
    <groupId>cn.hutool</groupId>
    <artifactId>hutool-all</artifactId>
    <version>5.2.0</version>
</dependency>

String content = "test";
Digester sha = new Digester(DigestAlgorithm.SHA256);
String sign = sha.digestHex(content);
System.out.println("签名结果-->" + sign);
```

4.1.4.2 加密

用户相关字段需要加密传输，加密字段为 userName 和 cardNo。加密使用 aes 算法。

将加密后的值更新到最终参数：

```
{
    "hospitalCode"： "1001",
    "userName"： "用户姓名 密文",
    "cardType"： "01",
    "cardNo"： "证件号码 密文",
    "timestamp"： 1597028576255,
    "sign"： "XXX"
}
```

加密方法：

```
<!-- 引入依赖： -->
<dependency>
    <groupId>cn.hutool</groupId>
    <artifactId>hutool-all</artifactId>
    <version>5.2.0</version>
</dependency>

String userName = "userName 值";
String cardNo = "cardNo 值";
String key = "78a0e4f3980117f135217cf3b3f0970d";

AES aes = SecureUtil.aes(key.getBytes());

String encryptUserName = aes.encryptHex(userName);
System.out.println("加密 userName 结果-->" + encryptUserName);

String encryptCardNo = aes.encryptHex(cardNo);
System.out.println("加密 cardNo 结果-->" + encryptCardNo);
```

4.2 活体检测 SDK 集成

4.2.1 SDK 集成

4.2.1.1 Android

gradle

```
implementation 'com.github.systoon:TNLiveDetect-Android:0.1.0'
```

maven

```
<dependency>
  <groupId>com.github.systoon</groupId>
  <artifactId>TNLiveDetect-Android</artifactId>
  <version>0.1.0</version>
  <type>pom</type>
</dependency>
```

4.2.1.2 iOS

CocoaPods

在工程的 `Podfile` 里面添加以下代码：

```
pod 'TNLiveDetect-iOS'
```

保存并执行 `pod install`,然后用后缀为`.xcworkspace` 的文件打开工程。

4.2.2 SDK 初始化流程

SDK 启动时进行初始化

4.2.2.1 流程说明

流程如下：

（一） 用户项实名认证平台申请一个账号（appId），每个账号有对应的密码（appSecret）用于参数签名验证。

（二） 初始化时传入 appId 和 appSecret。

（三） 调用 sdk 的检测接口 faceLiveDetect 开始检测并在回调方法中处理检测结果。

4.2.2.2 接口定义

4.2.2.2.1 活体检测接口

说明：

方法名	说明
faceLiveDetect	活体检测接口，返回检测结果

请求参数

参数	类型	是否必填	描述
无			

返回值

参数	类型	描述
code	Integer	响应码
message	String	响应信息
data	Object	图像数据

4.2.2.2.2 身份验证接口

说明：机具后端、公交出行业务系统后端分别提供给虚拟卡交易平台扣款处理完后回调

方法名	说明
checkFace	用于人脸识别,返回识别结果

请求参数

参数	类型	是否必填	描述
appId	String	是	应用 ID
appSecret	String	是	秘钥
name	String	是	姓名
certNo	String	是	身份证号码

响应参数

参数	类型	描述
code	Integer	响应码

message	String	响应信息
data	String	返回相似度，大于 0.7 认为信息匹配