**INSTASCAN**

Instascan is a low-cost, touch-less, mobile-basedbio-metric fingerprint identification solution that requires no additional hardware sensor.

**Mobile SDK:**

# Integration

* Add the following lines to the gradle (app) on root level
* The ARTIFACTORY\_ID and ARTIFACTORY\_PASS will be provided by Paysys Labs

allprojects { repositories {

jcenter() maven {

url ['http://artifactory.paysyslabs.com/instascan'](http://artifactory.paysyslabs.com/instascan%27) credentials {

username = "ARTIFACTORY\_ID" password = "ARTIFACTORY\_PASS"

}

}

}

}

* Add the following lines to the gradle (app) in dependencies

compile 'com.paysyslabs:instascan:1.0.0'

# Usage:

Following callbacks are provided by the SDK if activity inherits NadraBaseActivity:

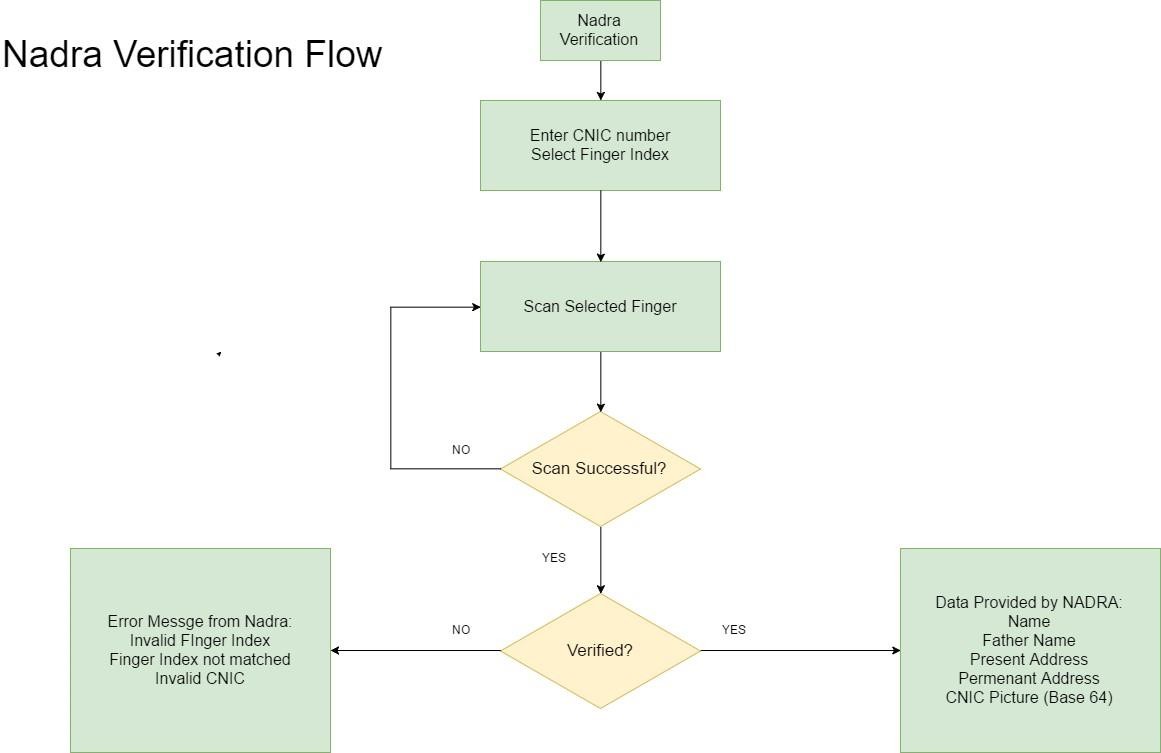
|  |  |  |
| --- | --- | --- |
| **Callback** | **Return Type** | **Parameters** |
| getLicenseKey | String |  |
| getScanFragmentContainer int | | |
| onNadraBaseSuccessfulScan | void | Map<String, String> tags |
| onNadraBaseError | void | String code, String message |
| onNadraBaseInvalidFingerIndex | void | String code, String message, String[] validFingers |

**OnNadraBaseSuccessfulScan** method will return a map of all fields. To parse the **PersonData** from the tags map, use the following snippet:

PersonData personData = new PersonData(); personData.setName(tags.get("name")); personData.setFatherHusbandName(tags.get("fatherHusbandName")); personData.setMotherName(tags.get("motherName")); personData.setGender(tags.get("gender")); personData.setPresentAddress(tags.get("presentAddress")); personData.setPermanentAddress(tags.get("permanentAddress")); personData.setDateOfBirth(tags.get("dateOfBirth")); personData.setBirthPlace(tags.get("birthPlace")); personData.setPhotograph(tags.get("photograph")); personData.setExpiryDate(tags.get("expiryDate")); personData.setCardExpired(tags.get("cardExpired"));

The SDK requires a license key which can be acquired from Paysys Labs.

# Demo Application Flow



**Custom Proxy Support**

For the concern of not keeping the Instascan license key on application, a custom proxy flow is integrated to the SDK which will post request to a custom URL with some authorization data, that custom URL will in turn post request to Instascan API after successful authorization.

### Client (Android):

1. Integrate the new SDK (with support of custom proxy) to the application.
2. In Nadra Rest Client, use the following callbacks:

@Override

public String getCustomProxyURL() { return "http://{YOUR-ENDPOINT}/ ";

}

@Override

public String getCustomCookie() { return "JSESSIONID=somesessionid";

}

@Override

public boolean useCustomProxy() { return true;

}

@Override

public Map<String, String> getCustomAuthenticationData() { HashMap<String, String> authorizationData = new HashMap<>();

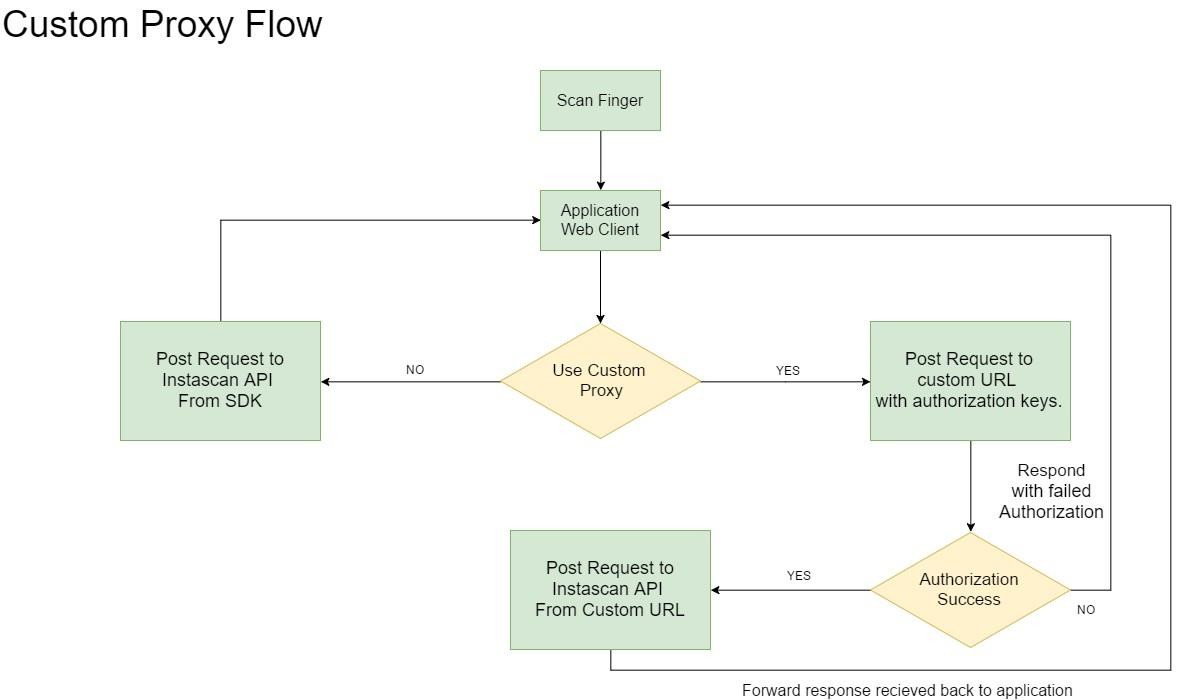
authorizationData.put("CustomKey", "Value"); return authorizationData;

}

1. If useCustomProxy returns false, application will have the previous flow. i.e the sdk will itself post requests to the Instascan API (pl.quickpay.pk/instascan). Otherwise, the SDK will call post the request to URL returned by getCustomProxyURL.
2. getCustomAuthenticationData returns a map which will contain authorizations keys.
3. If custom proxy is used, License key from the application will not be used. Rather it will be sent via server end. Thus license key can be removed from application.

### Server:

1. Request Body from client will have the following tags:
   * operation
   * data
   * forward
2. operation will be appended to ‘pl.quickpay.pk/instascan’
3. data will contain the map sent from application in getCustomAuthenticationData.
4. In case of successful authorization, forward will be posted to ‘pl.quickpay.pk/instascan/{operation} with the license key added to headers ‘Ocp-Apim-Subscription-key’
5. Response received will be directly sent back to application.



1. **Bio Metric Verifications:**

* 1. **Finger Print Verification**

**Request:**

{

"data" : {

"session" : "232322312124",

"username" : "inov8",

"password" : "inov81238"

},

"forward" : {

"finger" : "RIGHT\_RING",

"session" : null,

"identifier" : "6464948494949",

"institution" : "NADRA",

"wsq" : "finger template",

"sdk" : {

"application" : "com.inov8.jsblmfs",

"device" : "android",

"version" : "1.1.5-DG-0.2",

"applicationVersion" : "2.2",

"manufacturer" : "Xiaomi",

"model" : "Redmi Note 4"

}

},

"operation" : "verify"

}

**Response**

Successful

In case of successful verification of a customer from nadra, Mobile SDK will receive Tracking Id that will be used for account opening, Mobile app can show this tracking id on screen before sending Account Opening Request

**Note:** Customer data received from nadra will be verified against provided tracking id on Account Opening.

{

"status" : {

"code" : "100",

"message" : "successful"

},

"data" : {

"session" : "acdaebbf-4c2e-40e2-b74b-c820362a6c34",

"identifier" : "6464948494949",

"tags" : {

"cardExpired" : null,

"gender" : null,

"birthPlace" : null,

"photograph" : null,

"presentAddress" : null,

"cardType" : null,

"name" : null,

"fatherHusbandName" : null,

"motherName" : null,

"dateOfBirth" : null,

"permanentAddress" : null,

"fingerIndexes" : null,

"trackingId" : "125478965478"

}

}

}

### Response - 110

citizen number is not verified

{

"status": {

"code": "110",

"message": "citizen number is not verified"

}

}

### Response - 111

fingerprints does not exist in citizen database

{

"status": {

"code": "111",

"message": "fingerprints does not exist in citizen database"

},

"data": {

"session": "8bb9fe71-191a-402a-bcda-bed7d80c6fd1",

"identifier": "CNIC\_NUMBER"

}

}

### Response - 112

error generating session id

{

"status": {

"code": "112",

"message": "error generating session id"

}

}

### Response - 114

invalid verification reference number

{

"status": {

"code": "114",

"message": "invalid verification reference number"

}

}

### Response - 115

Invalid service provide transaction id

{

"status": {

"code": "115",

"message": "Invalid service provide transaction id"

}

}

### Response - 118

finger verfication has been exhausted for current finger.

{

"status": {

"code": "118",

"message": "finger verfication has been exhausted for current finger."

},

"data": {

"session": "1a01c4c0-4374-46ac-98bf-73799c2fc21f",

"identifier": "CNIC\_NUMBER"

}

}

### Response - 119

verification limit for current citizen number has been exhausted

{

"status": {

"code": "119",

"message": "verification limit for current citizen number has been exhausted"

}

}

### Response - 120

invalid input finger template

{

"status": {

"code": "120",

"message": "invalid input finger template"

}

}

### Response - 121

invalid finger index

{

"status": {

"code": "121",

"message": "invalid finger index"

},

"data": {

"session": "4772708a-4bab-4b69-a3ae-ec4919e53100",

"identifier": "CNIC\_NUMBER",

"tags": {

"fingerIndexes": "1,6,8,7"

}

}

}

### Response - 122

fingerprints does not matched

{

"status": {

"code": "122",

"message": "fingerprints does not matched"

},

"data": {

"session": "4772708a-4bab-4b69-a3ae-ec4919e53100",

"identifier": "CNIC\_NUMBER",

"tags": {

"fingerIndexes": "1,6,8,7"

}

}

}

### Response - 123

invalid finger template type

{

"status": {

"code": "123",

"message": "invalid finger template type"

}

}

### Response - 124

this operation will only be enabled if biometric verification of all available fingers is failed

{

"status": {

"code": "124",

"message": "this operation will only be enabled if biometric verification of all available fingers is failed"

}

}

### Response - 125

contact number is not valid

{

"status": {

"code": "125",

"message": "contact number is not valid"

}

}

### Response - 175

transaction id already exist

{

"status": {

"code": "175",

"message": "transaction id already exist"

}

}

### Response - 185

Invalid area name

{

"status": {

"code": "185",

"message": "Invalid area name"

}

}