**USE CASES And API SPECIFICATION DOCUMENT**

VERSION 1.2

**Merchant – Bill desk®**

**Preface**

This document is a brief description of the initial discussions had between the sales and pre-sales/technical team of Infrasoft Technologies ® with the corresponding teams of Central Bank of india and Bill desk ® about technical specifications and the architecture design of the application. Further changes or modifications to the design/architecture will be documented in the higher versions of same document with version number clearly specified in the title page and in the **Version Details** table of this document. (See **Version Details** table below).

**©Copyright Disclaimer**

No part of this document be reproduced, distributed, or transmitted in any form or by any means, including photocopying, recording or other electronic or mechanical methods, without prior written permission of Infrasoft Technologies® and is understood that the document is under company’s NDA restrictions. This document is intended for the purpose of evaluating our offer against the requirements of our clients and for assessing our competence in the IT services.

**Version Details**

|  |  |  |  |
| --- | --- | --- | --- |
| **Version No.** | **Modifications** | **Date** | **Author** |
| 1.1 | Initial discussions and understanding – Phase 1 | 05/06/2017 | Infrasoft Technologies ® |
| 1.2 | Added Message Encryption algorithm | 12/07/2017 | Infrasoft Technologies ® |
|  |  |  |  |
|  |  |  |  |

**Table of Contents**

1. [Requirement Description 3](#__RefHeading__2732_460001542)
2. [API Listing for Transaction Processing 4](#__RefHeading__2742_460001542)
3. [QueryTransaction Request 11](#__RefHeading__6770_1952340693)
4. [Sample Request and Response 15](#__RefHeading__2752_460001542)
5. [Initiate Collect 15](#__RefHeading__2754_460001542)
6. [QueryTransaction 15](#__RefHeading__2756_460001542)
7. Encryption Algorithm …………………………………………………………………………………………………….17

# Requirement Description

* In this case the client initiates the collect for payment request on the merchant app and website through the following steps:
* Customer visits any merchant portal and selects option as pay through UPI on Billdesk page.
* Client provides mandatory details such his VPA, mobile number, amount & transaction reference number.
* As the client choose to pay after he/she enters the required information, the information entered is sent to Central bank of India (CBI) through an API call and in turn CBI raises a payment request on client’s UPI app on his mobile.
* Client chooses ACCEPT or REJECT payment request on his mobile application within 5 minutes. If client does not do anything for 90 seconds the transaction stands CANCEL.
* If client accepts the payment request, the CBI api response back to Billdesk transaction status “successful”. If the client rejects the payment request, CBI api response back to Billdesk transaction status “failed”.

**Sequence of APIs for above use case:**

1. InitiateCollect (A1)
2. ~~UPIRequestStatusUpdation (B1) Or QueryTransaction (A2)~~

Transaction APIs are the set of APIs which would get called from Handheld Devices (Android or Windows) at the time of Payment of Delivery. Following APIs will be called during the transaction processing

## API Listing for Transaction Processing

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **API NAME** | **Request Parameter** | **Response** | **Description** |
| A1 | InitiateCollect | Payer Mobile Number, Payer Name, Payer Virtual Payment Address, merchant Code, Merchant VPA(Payee), Recipient Mobile Number, Transaction Reference Number, Amount, Transaction ID (optional), Signed Token,  currency code, Remark,  Ref Url, EntityId, AppID  Validity In Minutes | Ack Receipt with Transaction Number,  Status  Response code  Response Message,  order Number,  Transaction ID,  Amount debited,  Date time | This will generate a single collect request to the payer from beneficiary |
| ~~A2~~ | ~~QueryTransaction~~ | ~~Transaction Reference or Order Number, Token Number~~ | ~~Ack Response with Transaction processed information, Order Number, Transaction Reference Number, Status Message, Resp Code and Response Message~~ | ~~This would give retrieve the transaction details of any particular transaction~~ |

## Message Specifications

### Initiate Collect Request

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Sl.No** | **Request Input Fields** | **Datatype** | **Mandatory** | **Length** | **Description** |
| 1 | PayerMobileNumber | String | O | 12 | Buyer mobile number (10 digit) |
| 2 | PayerName | String | M | Min: 1 Max: 99 | Buyer Name |
| 3 | PayerVPA | String | M | Min: 1 Max: 99 | Buyer VPA |
| 4 | MerchantCode | String | M | Min: 1 Max: 6 | Merchant code at the time of registration |
| 5 | MerchantVPA(Payee) | String | M | Min: 1 Max: 99 | Merchant VPA |
| 6 | PayeeMobileNumber | String | M | 12 | Merchant Mobile number |
| 7 | TransactionRefNumber | String | M | 35 | Transaction reference number |
| 8 | Amount | Float | M | fractionDigits: 5 minInclusive: 0 totalDigits: 18 | Amount |
| 9 | TRN(optional) | String | O | Min: 1 Max: 35 | Can be left blank |
| 10 | AuthToken | String | M | 12 | Token received at the time of authentication. |
| 11 | Currency code | String | O | Min: 1 Max: 18 | Currency of the transaction defaulted to INR |
| 12 | Remark | String | M | Min: 1 Max: 50 | Remark |
| 13 | RefUrl | String | O | Min: 1 Max: 35 | Reference URL of the transaction <http://www.billdeskcom> |
| 14 | Validity | Number | M | It can be 1 minutes  to max 64,800  minutes | Request validity in minutes e.g. 1.5 minute |
| 15 | AppID | String | M | 7 | It should be “centralbank” |
| 16 | EntityID | String | M | 7 | It should be “centralbank” for UAT and production |

**Response**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Sl.No** | **Response** | **Datatype** | **Mandatory** | **Length** | **Description** |
| 1 | AckStatus | String | NA |  | Acknowledgement of the request |
| 2 | Status | String | NA | Success or fail | Status of collect request. |
| 3 | ErrorCode | String | NA | 2 | code 00 - successful and <> non zero failure |
| 4 | ResponseMsg | String | NA |  | Response/error Message in case of failure |
| 5 | TransactionID | String | NA | 35 | Transaction ID for initiated collect request |
| 6 | Amount | String | NA | fractionDigits: 5 minInclusive: 0 totalDigits: 18 | Collect amount to be debited. |
| 7 | TrnDateTime | Date | NA | ISODateTime Format  Max Length: 25 | Date and time of the transaction |

### ~~QueryTransaction Request~~

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **~~Sl.no~~** | **~~Request Input Fields~~** | **~~Data type~~** | **~~Mandatory~~** | **~~Length~~** | **~~Description~~** |
|  | ~~txnID~~ | ~~String~~ | ~~M~~ | ~~Min: 1 Max: 35~~ | ~~Transaction ID received in response of Initiate collect request~~ |
|  | ~~entityID~~ | ~~String~~ | ~~M~~ | ~~7~~ | ~~It should be “centralbank” for UAT and “centralbank” for Prod~~ |

**~~QueryTransaction Response~~**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **~~Sl.No~~** | **~~Response~~** | **~~Data type~~** | **~~Mandatory~~** | **~~Length~~** | **~~Description~~** |
|  | ~~entityID~~ | ~~String~~ | ~~NA~~ | ~~7~~ | ~~It should be “scb” for UAT and “scbl” for Prod~~ |
|  | ~~appID~~ | ~~String~~ | ~~NA~~ | ~~7~~ | ~~e.g.: “ST00001”~~ |
|  | ~~mobileNo~~ | ~~String~~ | ~~NA~~ | ~~10~~ | ~~Customer mobile number~~ |
|  | ~~txnID~~ | ~~String~~ | ~~NA~~ | ~~Min 1 Max: 35~~ | ~~It is a unique transaction id generated and returned in the response each time the service is called.~~ |
|  | ~~txnType~~ | ~~String~~ | ~~NA~~ | ~~10~~ | ~~The transaction type can be 'Collect' or 'Pay'.~~ |
|  | ~~expireAfter~~ | ~~DateTime~~ | ~~NA~~ |  | ~~Request (Inititate Collect or Pay) validity in minutes.~~ |
|  | ~~payerAddr~~ | ~~String~~ | ~~NA~~ | ~~Min: 1 Max: 99~~ | ~~Customer VPA~~ |
|  | ~~payeeAddr~~ | ~~String~~ | ~~NA~~ | ~~Min: 1 Max: 99~~ | ~~Merchant VPA~~ |
|  | ~~requestTime~~ | ~~DateTime~~ | ~~NA~~ |  | ~~Date and time of the transaction~~ |
|  | ~~responseStatus~~ | ~~String~~ | ~~NA~~ |  | ~~The status of the transaction. It can be 'COMPLETED', 'PENDING', 'REJECT' or 'FAILED'.~~ |
|  | ~~txnAmount~~ | ~~Float~~ | ~~NA~~ | ~~fractionDigits: 5 minInclusive: 0 totalDigits: 18~~ | ~~Amount~~ |
|  | ~~txnNote~~ | ~~String~~ | ~~NA~~ |  | ~~This is a remark on the transaction~~ |
|  | ~~payeeName~~ | ~~String~~ | ~~NA~~ | ~~99~~ | ~~Merchant Name~~ |
|  | ~~refID~~ | ~~String~~ | ~~NA~~ | ~~35~~ | ~~It is the unique transaction Id received response of collect request. It’s sent back to query the transaction status.~~ |
|  | ~~payeeMobile~~ | ~~String~~ | ~~NA~~ | ~~12~~ | ~~Merchant Mobile number~~ |

# Sample Request and Response

***Note: API URLs***

*API URLs provided below are tentative and will be finalized before the commencement of work*

## Initiate Collect

|  |
| --- |
| *{*  *"payerName": "NA",*  *"references": [{*  *"key": "Order No",*  *"value": "UCB40000494105"*  *}],*  *"payerAddr": "infra1@centralbank",*  *"merchantCode": "8891",*  *"remarks": "payment made",*  *"txnAmount": "1.15",*  *"payerMobileNo": "NA",*  *"payeeMobileNo": "NA",*  *"authToken": "Test",*  *"payeeVPA": "satapp@centralbank",*  *"entityID": "centralbank",*  *"transactionID": "UCB40000494105",*  *"appId": "centralbank",*  *"refUrl": "NA",*  *"currencyCode": "INR"*  *}* |

**Sample URL: http://192.168.209.24:8090/upi/Merchant/Services/InitiateCollect/V1/billdesk**

**Sample Response**

|  |
| --- |
| *{*  *"ackStatus": "00",*  *"status": "00",*  *"responseMsg": "OPERATION\_SUCCESSFUL",*  *"transactionId": "* CBIfctl39g9v9l154hvgcenhgrccm*",*  *"amount": "1.15",*  *"trnDateTime": "2016-12-05 15:52:46.677",*  *"authToken": "4W1uzCIzmi6fMECrCRH0Yw==",*  *"responseParameter": [],*  *"responseParameterMap": {}*  *}* |

## ~~QueryTransaction~~

**~~Sample URL:~~**

[~~http://<Serverip>:<Serverport>/upi/Merchant/Services/InitiateCollect/V1/~~](http://180.92.171.234:8090/upi/Merchant/Services/InitiateCollect/V1/stfcl)~~billdesk~~

**~~Sample Request~~**

|  |
| --- |
| *~~{~~*  *~~"txnID": "~~*~~CBIfctl39g9v9l154hvgcenhgrccm~~*~~",~~*  *~~“entityID”: ”centralbank”,~~*  *~~}~~* |

**~~Sample Response~~**

|  |
| --- |
| *~~{~~*  *~~"entityID": "centralbank",~~*  *~~"appID": "centralbank",~~*  *~~"mobileNo": "9999999999",~~*  *~~"txnID": "~~*~~CBIfctl39g9v9l154hvgcenhgrccm~~*~~",~~*  *~~"txnType": "COLLECT",~~*  *~~"expireAfter": "21/04/2017 18:56:57",~~*  *~~"payerAddr": "reqpay@sc",~~*  *~~"payeeAddr": "stfc@sc",~~*  *~~"requestTime": "21/04/2017 16:15:57",~~*  *~~"responseStatus": "PENDING",~~*  *~~"txnAmount": "1.15",~~*  *~~"txnNote": "payment made",~~*  *~~"payeeName": "STFCL",~~*  *~~"refID": "~~*~~CBIfctl39g9v9l154hvgcenhgrccm~~*~~",~~*  *~~"payeeMobile": "8888888888"~~*  *~~}~~* |

**Message encryption Implementation:**

**RSA Algorithm**

RSA Key generation: Generating RSA public and private keys for sighing message using below mentioned java code:

System.out.println("-------GENRATE PUBLIC and PRIVATE KEY-------------");

KeyPairGenerator keyPairGenerator = KeyPairGenerator.getInstance("RSA");

keyPairGenerator.initialize(8192); //1024 used for normal securities

KeyPair keyPair = keyPairGenerator.generateKeyPair();

PublicKey publicKey = keyPair.getPublic();

PrivateKey privateKey = keyPair.getPrivate();

System.out.println("Public Key - " + publicKey);

System.out.println("Private Key - " + privateKey);

//Pullingout parameters which makes up Key

System.out.println("\n------- PULLING OUT PARAMETERS WHICH MAKES KEYPAIR----------\n");

KeyFactory keyFactory = KeyFactory.getInstance("RSA");

RSAPublicKeySpec rsaPubKeySpec = keyFactory.getKeySpec(publicKey, RSAPublicKeySpec.class);

RSAPrivateKeySpec rsaPrivKeySpec = keyFactory.getKeySpec(privateKey, RSAPrivateKeySpec.class);

System.out.println("PubKey Modulus : " + rsaPubKeySpec.getModulus());

System.out.println("PubKey Exponent : " + rsaPubKeySpec.getPublicExponent());

System.out.println("PrivKey Modulus : " + rsaPrivKeySpec.getModulus());

System.out.println("PrivKey Exponent : " + rsaPrivKeySpec.getPrivateExponent());

//Share public key with other so they can encrypt data and decrypt thoses using private key(Don't share with Other)

System.out.println("\n--------SAVING PUBLIC KEY AND PRIVATE KEY TO FILES-------\n");

RSAEncryptionDecryption1 rsaObj = new RSAEncryptionDecryption1();

rsaObj.saveKeys(PUBLIC\_KEY\_FILE, rsaPubKeySpec.getModulus(), rsaPubKeySpec.getPublicExponent());

rsaObj.saveKeys(PRIVATE\_KEY\_FILE, rsaPrivKeySpec.getModulus(), rsaPrivKeySpec.getPrivateExponent());

**Message Encryption:**

Request message will me encrypted with RSA public key using below mentioned Java code:

**private** **static** **byte**[] encryptData(String data) **throws** IOException {

System.***out***.println("\n----------------ENCRYPTION STARTED------------");

System.***out***.println("Data Before Encryption :" + data);

**byte**[] dataToEncrypt = data.getBytes();

**byte**[] encryptedData = **null**;

**try** {

PublicKey pubKey = *readPublicKeyFromFile*(***PUBLIC\_KEY\_FILE***);

Cipher cipher = Cipher.*getInstance*("RSA");

cipher.init(Cipher.***ENCRYPT\_MODE***, pubKey);

encryptedData = cipher.doFinal(dataToEncrypt);

System.***out***.println("Encryted Data: " + encryptedData);

} **catch** (Exception e) {

e.printStackTrace();

}

System.***out***.println("----------------ENCRYPTION COMPLETED------------");

**return** encryptedData;

}

**Message Decryption:**

Response message will decrypted with RSA private key using below mentioned java code:

**private** **static** **void** decryptData(**byte**[] data) **throws** IOException {

System.***out***.println("\n----------------DECRYPTION STARTED------------");

**byte**[] descryptedData = **null**;

**try** {

PrivateKey privateKey = *readPrivateKeyFromFile*(***PRIVATE\_KEY\_FILE***);

Cipher cipher = Cipher.*getInstance*("RSA");

cipher.init(Cipher.***DECRYPT\_MODE***, privateKey);

descryptedData = cipher.doFinal(data);

System.***out***.println("Decrypted Data: " + **new** String(descryptedData));

} **catch** (Exception e) {

e.printStackTrace();

}

System.***out***.println("----------------DECRYPTION COMPLETED------------");

}

RSA Certificates:



Certificate flow:

