UPI Web Collect

API Specification Document

1.0 Contents

2.0 Overview

This document provide details related to UPI web collect APIs. This APIs is used to initiate a collect request from the merchant website. Money request is from the merchant to customer in the form of UPI collect notification/ request in the respective UPI app. There is an expiry time after which request expires automatically.

3.0 Getting started

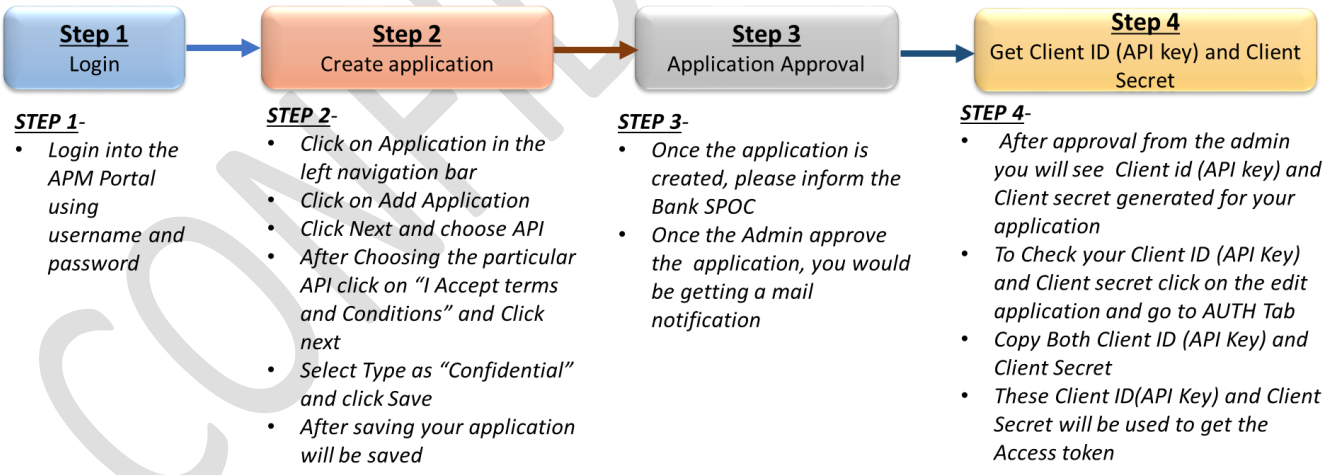
As your registration has been approved. Now you can access the APIs. For getting access to the APIs please follow below step:-

3.1 UAT

- Please register on Kotak Developer portal (https://api.kotak.com) and inform your Kotak SPOC
- Once your registration is approved (TAT-1 Day), please complete the UAT testing using UAT credential's provided by the Kotak team
- UAT testing need to be completed using OAuth token (Please refer section 3.3 for the same)

3.2 Production

Note: - For all the production request client need to send environmental tag in the Header. Please reach out to your Kotak SPOC to get the environmental tag value for production



Please refer help section on the portal for any help or Kotak Developer Guide shared by the Kotak team for more details and screen shots.

3.3 OAuth

Now you have got the Client ID (API key) and Client Secret. As a next step you need to generate the token by passing your client credentials (**Client Id** and **Client Secret**) in a simple call to OAuth server. The operation returns a token that's good for about 300 seconds.

For each API request, new OAuth token need to be generated and pass. For more details please check the document available on portal or click here

4.0 APIs Overview and sequence to call the APIs

The documentation consists of the request parameters, response parameters along with their examples for below APIs

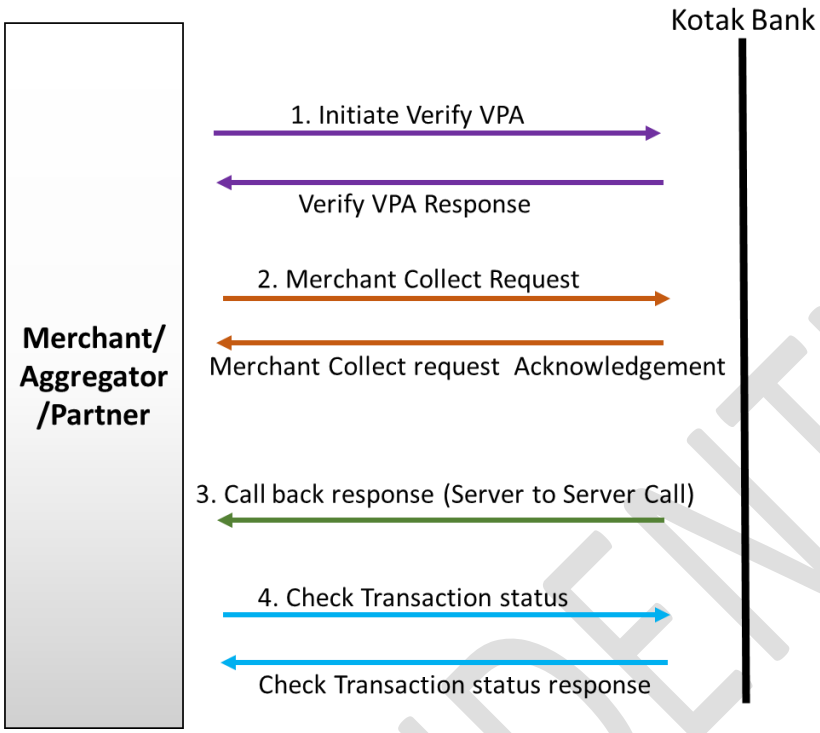
Below table provide brief description of the APIs

S. No.	API Name	Description
1.	VPA Verification*	This API is used to check whether a particular UPI ID is valid or not. It also fetches Customer name as registered in Account
2.	Merchant Collect	This API is used to create a money request from merchant to customer in the form of UPI collect notification/ request in the respective UPI app. There is an expiry time after which request expires automatically
3.	CheckTransactionStatus	This API is used to check the status of transaction whether it was successful or failure with the response code. It works based on fetching status against Transaction ID
4.	Merchant Cash Back	This is a Payment API and will serve the purpose of sending money from merchant account to a customer account via UPI. It can be used for cashbacks, refunds, disbursements etc.

For VPA verification API, please add separate API (Merchant_Verify_VPA) in the application along with UPI web-collect API. For any assistant please reach out to Kotak SPOC.

UPI web collect APIs need to be called in below sequence.

NOTE: - Depending on different merchant/Aggregator/Partner few APIs may not be required



5.0 VPA Verification

5.1 Description

This API is used to check whether a particular UPI ID is valid or not. It also fetches Customer name as registered in Account

5.2 Transport Method

HTTPS/ POST

5.3 API Request URL

UAT URL: https://apigwuat.kotak.com:8443/Merchant/VerifyVPA

Production URL: https://apigw.kotak.com:8443/Merchant/VerifyVPA

5.4 Access Requirements

OAuth

5.5 Request Parameters

Field	Data Type	Mandatory	Description	Parameter Constraints
Aggregator ID	String (20)	M	Aggregator id – This id will be created and shared by bank when aggregator gets on boarded for UPI transactions	No Special characters
Customer ID	Number(12)	M	Mobile number provided by merchant during on boarding	Only Numeric values allowed
MerchantID	String(20)	M	Merchant id generated and provided by bank during on boarding process	No Special characters
VPA	String(35)	M	Remitter VPA who's validity needs to be checked	

5.6 Sample Request JSON

Please be noted sample UAT values will be shared by the BSG team over the mail

```
{
  "aggregatorId": "ABC",
  "customerId": "91040000123",
  "merchantId": "ABC001",
  "vpa": "test@kotak"
}
```

5.7 Response Parameters

Field	Data Type	Mandatory	Description
Code	String	N	00 or ZH
Result	String	N	00-Success ZH-Invalid Virtual Address
Data	String	N	Remitter Name mapped against input vpa

5.8 Sample Response


```
{
  "code": "00",
  "result": "Success",
  "data": "Test Kumar"
}
```

6.0 Merchant Collect

6.1 Description

This API is used to create a money request from merchant to customer in the form of UPI collect notification/ request in the respective UPI app. There is an expiry time after which request expires automatically.

This API uses Checksum method (please refer checksum generation logic section for algorithm) and the same needs to be passed in the X-Check tag in the header

6.2 API Request URL

UAT URL: https://apigwuat.kotak.com:8443/upi/Merchant/Collect

Production URL: https://apigw.kotak.com:8443/upi/Merchant/Collect

6.3 Transport Method

HTTPS/ POST

6.4 Access Requirement

Pass x-check tag in the header which will contain the checksum value and OAuth

6.5 Request Parameters

Request Body	Data Type	Mandatory	Description
Merchant Collect	JSON Object	Y	

Merchant Collect JSON Object Field Descriptions:

Note : O - Optional, M - Mandatory

Field Name	Data Type	Size	Notes
amount	Number(M)	999999999	Amount should be in decimal format, eg: 12.45
customerId	String(M)	N12	Customer mobile number: 919876543210
orderId	String(0)	N	Any value eg:123456
payerVpa	String(M)	AN3..40	Payer vpa, eg:payer@kotak
submerchantVPA	String(O)	N	Optional field
remarks	String(M)	AN3..50	Remarks, eg: Merchant Collect
txnId	String(M)	AN35	Txn id,eg:KMB023432svsff234324sdvsvsvv234345 Note: Txnid logic has been provided in this document, please refer the same
aggregatorVPA	String(M)	AN3..40	This is merchant vpa, eg: merchantvpa@kotak
expiry	Number(M)	N5	Expiry is in minutes, eg: 18000

timeStamp	String(M)	N	Time is in 24 hr format, DD-MM-YYYY HH:MM:SS, Eg:01-06-2018 17:15:14
merchantReferenceCode	String(M)	N	Bank team will create unique merchant id for each merchant, eg: merm0001
submerchantReferenceid	String(O)	N	Optional field, if present pass merchant id, eg: merm0001
deviceDetails	Object(M)	N-AN	i. Mobile number is mandatory in the device details object, mobile=919876543210 ii. Bank team will create unique app id for each aggregator/merchant and is mandatory in the device details object, eg:app=com.appname.upi
debitAccount	Number(O)	N	Merchant debit account(Future enhancement)
ifsc	String(O)	AN11	Merchant IFSC code(Future enhancement)
refUrl	String(O)	AN3..4	Future enchancement

6.6 Sample Request JSON

X-Check: 3HRAMkpg8Dhz1Nig2DcxuvzzqcZlwQMUVGYUrq2dXww8+Uu7+Mf4H2tAuRWWWRSz

```
{
	"aggregatorVPA": "test@kotak",
	"amount": "10.00",
	"customerId": "91123456789",
	"deviceDetails":
		{
			"mobile":"919179440999",
			"app": "test.upi"
		},
	"expiry": "18000",
	"merchantReferenceCode": "test001",
	"orderId": "15001511",
	"payerVpa": "testuat@kotak",
	"referenceId": "804014035129",
	"remarks": "test",
	"timeStamp": "09-02-2017 11:05:15",
	"txnId": "KMBMCCAVLHCBBBZ9PGLAAHCC4BXJAK9R002",
	"debitAccount":"0412101009000309",
	“ifsc”:”KKBK0123456”,
	“refUrl”:”https://testing.com”
}
```

6.7 Response Parameters

Field Name	Data Type	Size	Notes
Code	Number(M)	N	Response Code (00)
Result	String(M)	AN3..40	Response Result
Data	Object(M)		Data object contains below fields
orderId	String(M)	N	Any value eg:123456
referenceId	String(M)	N	It is RRN and is 12digits in length.eg:804014035129
payerVpa	String(M)	AN3..40	Payer vpa, eg: payervpa@kotak
payerName	String(M)	A	Name of the payer
txnId	String(M)	AN35	Txn id,eg:KMB023432svsff234324sdvsvsvv234345 Note: Txnid logic has been provided in this document, please refer the same
aggregatorVPA	String(M)	AN3..40	This is merchant vpa, eg: merchantvpa@kotak
expiry	Number(M)	N5	Expiry is in minutes, eg: 18000
amount	Number(O)	999999999	Amount should be in decimal format, eg: 12.45
timeStamp	String(O)	N	Time is in 24 hr format, DD-MM-YYYY HH:MM:SS, Eg:01-06-2018 17:15:14

Possible values for Code in Merchant Collect Request:
00 – Accepted Collect Request

6.8 Sample Response JSON


```
{
	"code": "00",
	"result": "Accepted Collect Request",
	"data":
	{
		"orderId": "15001511",
		"referenceId": "804014035129",
		"payerVpa": "test@kotak",
		"payerName": null,
		"txnId": "KMBMCCAVLHCBBBZ9PGLFWHCC4BXJAK9R002",
		"aggregatorVPA": "test1@kotak",
		"expiry": "18000",
		"amount": "10.00",
		"timeStamp": "09-02-2017 11:05:15"
	}
}
```

7.0 Collect Call Back Response

Collect call back response will be send to the partner through direct server to server call on the call back URL configured against each partner

7.1 Response Parameters description

Field Name	Data Type	Size	Notes
transactionid	String(M)	AN35	Txn id, eg:KMB023432svsff234324sdvsvsvv234345 Note: This will be same as transactionid which is sent in collect request
aggregatorcode	String(M)	AN40	Code allocated to each aggregator, will be provided by bank. Eg: agr001
merchantcode	String(M)	AN40	Unique Id created against each merchant eg: merm0001
status	String(M)	A30	Status of the transaction, eg: SUCCESS
statusCode	String(M)	N2	Code for each transaction, eg: 00 for SUCCESS
description	String(M)	AN40	Description, eg: Testing
remarks	String(M)	AN40	Remarks, eg: Testing
transactionreference number	String(M)	N12	If merchant sends order id in the request then order id will be sent in this field, otherwise RRN will be sent, eg:721411361239
rrn	Number(M)	N12	RRN number, eg:721411361239
amount	Number(O)	999999999	Amount should be in decimal format, eg: 12.45
type	String(M)	A20	Type of the Transaction, COLLECT
payervpa	String(M)	AN3..40	Payer vpa, eg: payervpa@psp
payeevpa	String(M)	AN3..40	Payee vpa, eg: payeevpa@kotak
refurl	String(M)	AN30	ResponsePay Url, eg:https://mb.uat.kotak.com. The value passed in input will be sent back. In case input value is null default url will be sent.
refid	String(M)	N12	If merchant sends order id in the collect request then order id will be sent in this field, otherwise RRN will be sent, eg:721411361239
transactionTimestamp	String(M)	date	Time stamp should be 24hr format, YYYY-MM-DD HH:MM:SS.ZZZ eg:2017-12-13 19:18:33.617
checksum	String(M)	AN40	merm0001merchantvpa @psp customervpa@psp KMB01ba1363715005d82a453d223e b13b7772017-12-13 19:18:33.61719.0072141136123900Funds Transfer721411361239

7.2 Sample Response:

{
"transactionid":" KMB01ba1363715005d82a453d223eb13b777",
"aggregatorcode": "",
"merchantcode": "merm0001",
"status": "SUCCESS",
"statusCode": "00",
“description": "Test",
"remarks": "Test",
"transactionreferencenumber":"721411361239”,
"rrn": "721411361239",
“amount": "19.00",
"type": "COLLECT" ,
"payervpa": "Test@kotak",
"payeevpa": "abc@kotak",
"refurl":"https://mb.uat.kotak.com",
"refid": "10000468",
"transactionTimestamp":"2017-12-13 19:18:33.617",
"checksum": " merm0001siva@kotakkumar@kotakKMB01ba1363715005d82a453d223e
b13b7772017-12-13 19:18:33.61719.0072141136123900Funds Transfer721411361239"
}

7.3 Possible values for Code & Result in Merchant Callback Response:

Code	Result
00	SUCCESS
01	REJECTED
02	EXPIRED
03	FAILED

8.0 Check Transaction Status

8.1 Description

This API is used to check the status of transaction whether it was successful or failure with the response code. It works based on fetching status against Transaction ID

8.2 Transport method

HTTPS/POST

8.3 API Request URLs

UAT URL: https://apigwuat.kotak.com:8443/upi/Merchant/ChkTranStatus

Production URL: https://apigw.kotak.com:8443/upi/Merchant/ChkTranStatus

8.4 Access Requirements

OAuth

8.5 Request Parameters

Field Name	Data Type	Size	Notes
aggregatorVPA	String(M)	AN3..40	This is merchant vpa, eg: merchantvpa@kotak
amount	Number(O)	999999999	Amount should be in decimal format, eg: 12.45
customerId	String(M)	N12	Customer mobile number: 919876543210
dateTime	String(O)	N	Time is in 24 hr format, DD-MM-YYYY HH:MM:SS, Eg:01-06-2018 17:15:14
orderId	String(0)	N	Any value eg:123456
Refid	String(0)	N	It is RRN and is 12digits in length.eg:804014035129
tranid	String(M)	AN35	Txn id,eg:KMB023432svsff234324sdvsvsvv234345 Note: Txnid logic has been provided in this document, please refer the same. Txn ID should be same as passed in collect request
type	String(0)	AN35	Optional field

8.6 Sample Request JSON

```
{
        "aggregatorVPA": "test@kotak",
       "customerId": "919179440372",
        "tranid": "KMBMCCAVLHCBBBZ9PGLFWHCC4BXJAK9R002"
}
```

8.7 Response Parameters

Field Name	Data Type	Size	Notes
Code	Number(M)	N	Response Code
Result	String(M)	AN3..40	Response Result
Data	Object(M)		Data object contains below fields
status	String(M)	A1	Represents status of the transaction - 00/T01/T02
amount	Number(O)	999999999	Amount should be in decimal format, eg: 12.45
aggregatorVPA	String(M)	AN3..40	This is merchant vpa, eg: merchantvpa@kotak
payerVpa	String(M)	AN3..40	Payer vpa, eg: payervpa@kotak
txnId	String(M)	AN35	Txn id, eg:KMB023432svsff234324sdvsvsvv234345 Note: Txnid logic has been provided in this document, please refer the same
orderId	String(M)	N	Any value eg:123456
payerName	String(M)	A	Name of the payer
referenceId	String(M)	N	It is RRN and is 12digits in length.eg:804014035129
txntime	String(M)	N	Time is in 24 hr format, YYYY-MM-DD HH:MM:SS.SSS, Eg:2018-02-09 14:29:57.917

8.8 Sample Response JSON


```
{
  "code": "T01",
  "result": "Transaction Pending",
  "data":
        {
              "status": "P",
              "amount": "10.00",
              "aggregatorVPA": "ccav001@kotak",
              "payerVPA": "kumaruat2@kotak",
              "txnid": "KMBMCCAVLHCBBBZ9PGLFWHCC4BXJAK9R002",
              "orderid": "15001515",
              "payerName": null,
              "referenceId": "804014035129",
              "txntime": "2018-02-09 14:29:57.917"
        }
}
```

8.9 Possible values for Code & Result in Merchant Check Txn Status:

Code	Result	Status
00	SUCCESS	C (C mean completed)
T01	Transaction Pending	P (P mean pending)
T03	Status Not Found	Null
T04	Status Not Found	Null

9.0 Merchant Cash Back

9.1 Description

This is a Payment API and will serve the purpose of sending money from merchant account to a customer account via UPI. It can be used for cashbacks, refunds, disbursements etc.

9.2 Transport method

HTTPS/POST

9.3 API Request URLs

UAT URL: https://apigwuat.kotak.com:8443/upi/Merchant/Pay

Production URL: https://apigw.kotak.com:8443/upi/Merchant/Pay

9.4 Access Requirements

OAuth

9.5 Request Parameters

Note:- 1) M- Mandatory | O-Optional
2) Device Details Object mobilenumber Mandatory
3) X-Check validate the checksum for request body

Field Name	Data Type	Length	Sample
amount(M)	String	N	1.00
customerId(M)	String	12	919397123108
accountholdername (M)	String	AN	siva
Accrefnumber(M)	String	N	4211172426
Acctype(O)	String	AN3..40	SAVINGS
payerVpa(M)	String	AN3..40	merchantvpa@kotak
remarks(M)	String	AN3..50	Merchant Prepay Test
txnId(M)	String	AN35	KMBMTEST310118124749000000010000002
Payeevpa(M)	String	AN3..40	customervpa@kotak
mcc(M)	String	N10	1520
ifsc(M)	String	AN3..40	KKBK0000646
merchaentrefid (M)	String	AN3..40	test0001
approvalRef(0)	String	N	1341432155
customerRefid(0)	String	N	32412414141
deviceDetails(M)	Object	N	919397123108
X-check	String	M	OOtdkMGCMa1+Ca4S5tXH7DR4kPif+16nu+6

			P6gFNtwuTZPRD+dQkUW3RWFbnWHJD

9.6 Sample Request JSON


```
{
  "accountholdername": "merchant name",
  "accrefnumber": "4211172426",
  "acctype": "savings",
  "amount": "1.00",
  "approvalRef": "123456",
  "customerRefid": "4444444",
  "devicedetails": {
    "app": "string",
    "capability": "1111",
    "gcmid": "string",
    "geocode": "12.02,17.03",
    "id": "string",
    "ip": "127.0.0.1",
    "location": "string",
    "mobile": "919397123108",
    "os": "Android",
    "type": ""
  },
  "ifsc": "KKBK0000646",
  "mcc": "1520",
  "merchaentrefid": "merm0001",
  "mobilenumber": "919397123108",
  "payeevpa": "customervpa@kotak",
  "payervpa": "merchantvpa@kotak",
  "remarks": "Merchant Prepay Test",
  "txnid":
 "KMBMTEST3101181247490000000010000002"
}
```


X-Check
OOtdkMGCMa1+Ca4S5tXH7DR4kPif+16nu+6P6gFNtwuTZPRD+dQkUW3RWFbnWHJD

9.7 Response Parameters

Field Name	Data Type	Mandatory	Notes
Code	String	M	Response Code
Result	String	M	Response Result
Data	Object	M	{ "code": "string", "result": "string", "data": "string" }
Code	String	M	Response Code

9.8 Sample Response JSON

{ "code": "00",
"result": "Success",
"data": "719117018202 "
}

10.0 Algorithm for Checksum Generation

10.1 Algorithm for collect checksum generation:-

The following steps to be performed for a secure web collect request.

Step -1: Calculate the check sum using Collect object.

a) The order of checksum input and fields are mentioned as below,

```
public String getInput ()
{
        return  aggregatorVPA+amount+customerId+(deviceDetails!=null?
        deviceDetails.getInput() : "") +expiry +timestamp+merchantReferenceCode+
        payerVpa+orderId+remarks+(refurl!=null? refurl : "")+txnId;
}
```

< deviceDetails.getInput() >:

```
public String getInput ()
 {
   return app+capability+gcmid+geocode+id+ip+location+mobile+os+type;
}
```

b) Keep it as empty if any field is null

```
(refurl!=null? refurl : "")
(app!=null? app: "")
```

Step-2: Create digest with sha256 algorithm by using checksum input.

```
digest = Crypto.SHA256(mrc.getInput());
 public static byte[] SHA256(String paramString)throws Exception
{

        MessageDigest localMessageDigest =     MessageDigest.getInstance("SHA-
256");
        localMessageDigest.update(paramString.getBytes("UTF-8"));
        byte[] digest = localMessageDigest.digest();
        return digest;

 }
```

Step-3: Encrypt the digest by using AES secrete key specifications.

```
byte [] encData=   Crypto.encrypt(Crypto.hexStringToByteArray(skey),digest);
public static byte[] hexStringToByteArray(String s)
{
        byte[] b = new byte[s.length() / 2];
        for (int i = 0; i < b.length; i++) {
                int index = i * 2;
                int v = Integer.parseInt(s.substring(index, index + 2), 16);
                b[i] = (byte) v;
        }
        return b;
}
public static byte[] encrypt(byte[] key,byte[] data)throws Exception
{
        System.out.println(">>>>>>>>>KEY::"+key.length);
        SecretKeySpec secretKeySpec = new SecretKeySpec(key, "AES");
        byte[] iv = new byte[16];
        IvParameterSpec ivSpec = new IvParameterSpec(iv);
```

```
        Cipher acipher = Cipher.getInstance("AES/CBC/PKCS5Padding");
        acipher.init(Cipher.ENCRYPT_MODE, secretKeySpec,ivSpec);
        byte[] arrayOfByte1 = acipher.doFinal(data);
        return arrayOfByte1;
    }
```

Step-4: Calculate the check sum value and send as **X-Check header** along with the json object.

```
String checkSumval= Base64.encodeBase64String(encData);
```

10.2 Algorithm for callback checksum generation:-

```
public static String sha256(String base)
    {
        try
        {
        MessageDigest digest = MessageDigest.getInstance("SHA-256");
        byte[] hash = digest.digest(base.getBytes("UTF-8"));
        StringBuffer hexString = new StringBuffer();

        for (int i = 0; i < hash.length; i++)
            {
                String hex = Integer.toHexString(0xff & hash[i]);
                if(hex.length() == 1) hexString.append('0');
                hexString.append(hex);
            }

            return hexString.toString();
        }
        catch(Exception ex)
        {
            throw new RuntimeException(ex);
        }
    }
```

10.3 Checksum data for Collect Callback:

String data = merchantcode + payeevpa + payervpa + transactionid+ transactionTimestamp + amount + refid + statusCode + remarks + rrn;

11.0 Encrypted Callback response logic

For the merchants who want the callback response in encrypted form, need to share their public key with the bank to encrypted the request, callback will be sent in below encrypted form.

11.1 Encryption Logic:

```
String value= SPBahdzEXefS23AS // Merchant shared Key
jsonText = new String(Base64.getEncoder().encode(encrypt(value.getBytes(),
jsonText.getBytes())));
public byte[] encrypt(byte[] key,byte[] data)throws Exception
        {
                System.out.println(">>>>>>>>>>KEY::"+key.length);
                 SecretKeySpec secretKeySpec = new SecretKeySpec(key, "AES");
                 Cipher acipher = Cipher.getInstance("AES");
               acipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
              byte[] arrayOfByte1  = acipher.doFinal(data);
               return arrayOfByte1;
        }
```

11.2 Sample Response:

{"data":"GH9Q7aO8GTzt+91hPK+wNfHQ1VMywUhsEqiyzJ16rtmHzzNs3zZFjUsJsoabQOO
x5Oh7qMviOBCOjOFnbIciqDFN9LQ6eXDBmT\/hl6J86HP6K5l\/lE+xMvcZ0XR6039AnL6EW
qQ6w+XKBpdZxj9vX0u4Ldz4abnibC+qiFfUoxEVWAId7KmCf9gVkdORHobeZp3Fnk6CuHS
g1HDsHbpwTaNVRyFkQG7khMzaWgiS7E2JvFM4xn87mWyfQezs74zqlUOwoPHTXX3rVl0
BqxYAuZAi7gyhEk5PXbAfYH7dyavQbV1xI5luqed2c5\/QQiWW+SE9+0pNPjOj3Eug1xbAYZ
dHug0gGxA72oTKBsDwfXyxUE\/wVTDxwdJOCRXMI0TfvV6s7PClFNHjXpYsxpSFDcAqCrT
qLEQSmC0XslazVoYSUI9Pl07u4IhtOU8zuISUWxojL7GmfWPlJoU8EmeIARdiPM6r1paDu1
EcBf8gl+JGDGQHdQoC\W3eclcjNhNzPUi1n2qKWjbvv5\/7HxT9MbNDw08J325Jm6DbiZyD
QPYAFQSCZf\/aZ5pvHAYlLoWEV2tUe\/zy1zqnxoPqlU2T4PnxItmvtpEvdSUbVTRSWQRY2
6LHATj7\/3CihPPllB7r4zMmxNFLukVimvaAgTfxZxCHkOrXZyPiTbupnkjFMpoGgj4mRpPXx
KWxD7olECgYRPgs13TPC6OhLW4FKFB5iA=="}

12.0 Transaction ID Generation Logic

Please use the below format for generating the transaction id.

1-3: KMB
4-4 :M
5-8: ABCD(Bank team will provide the value)
9-35:UUID logic (Unique number maintained at merchant id)

Please click here to refer to NPCI UUID generation logic document or please refer UUID document available on portal

13.0 Error Codes

Please find below the list of error codes:

Error Code	Description
	Customer/Account
00	Success
OL01	Merchantrefid_notFound
OL02	Invalid_merchantrefid
OL03	Invalid_vpa
OL04	merchantTxnLimitsNotAvilable
OL05	PerDayCounterExceed
OL06	PerTxnAmountExceed
OL07	PerDayAmountExceed
OL08	InitiatePullNotAllowed
OL09	InitiatePaymentNotAllowed
OL10	MDRNotConfigured
OL11	PerWeekCounterExceed
OL12	PerMonthCounterExceed
OL13	MerchantRefAndVpaNotMatched
OL14	AmountNotInMdrRange
OL80	DatabaseError
OL91	Timeout
OL92	InvalidMobileNumber
OL93	MobilenumberAndVpaNotMatched
OL94	Customeridmismatched
OL95	IPAddressInvalid
OL96	KeyInvalid
OL97	IPAddressNotAvilable
S01	AggregatorVpaMandatory
S02	TransactionIdMandatory
S03	CustomerIdMandatory
99	InvalidCheckSumData
C00	CheckSumSuccess
C01	MobilenumberIsMismatched
C03	ExpiryMandatory
C04	MerchantReferencecodeMandatory
C05	PayerVpaMandatory
C06	TimestampMandatory
C07	TxnidMandatory
C08	AmountFiledIsMandatory
C09	RemarksMandatory
T01	TransactionPending
T02	StatusNotFound

14.0 Production escalation matrix

NOTE:- This escalation matrix strictly needs to be followed for Production issues only

Escalation matrix	Contact Person Name	Contact details	Email address
1st Escalation	channels and payments application support; App Mon Support	918061574183/91806154357/7760233877 918061574266	channel.applicationsupport@kotak.com; appmon.support@kotak.com
2nd Escalation	Prashant Kumar	918061574247/6366373473	Prashant.Kumar6@kotak.com
3rd Escalation	Group SM Channels	918061574146/6364580565	sm.channels@kotak.com
4th Escalation	Ritisha Thomas	918061574135/9980927439	ritisha.thomas@kotak.com
5th Escalation	Usha Govindan	918061574183/91806154357/7760233877	usha.govindan@kotak.com