



# COMPOSITE PAY API Documentation

## Document Revision

Date	Version	Description
14-Jun-2019	0.1	Initial Draft
17-Jun-2019	0.2	CIB API added
18-Jun-2019	0.3	Changed API process flow
01-Jul-2019	0.4	Inclusion of each flow request-response packet.
10-Jul-2019	0.5	UPI Fields Description updated.
17-Jul-2019	0.6	UPI fields addition
25-Jul-2019	0.7	Error Codes Addition- UPI, IMPS, NEFT
30-Jul-2019	0.8	Curl added
01-Aug-2019	0.9	UPI request packet correction
07-Aug-2019	0.10	IMPS API spec update
08-Aug-2019	1.01	IMPS payload changes
12-Aug-2019	1.02	IMPS Tran Inquiry API Added
16-Sep-2019	1.03	Addition Beneficiary Registration API
01-Oct-2019	1.04	Addition of bene reg. codes & change in UPI request packet
04-Nov-2019	1.05	Revision of error codes
16-Dec-2019	1.06	Inclusion of RTGS API
31-Dec-2019	1.07	Inclusion of Beneficiary APIs for UPI
02-Mar-2020	1.08	Revision of Error codes & Documentation
08-Mar-2020	1.09	Revision of FAQs
12-Mar-2020	1.10	Updating UPI request packet & introduction part
04-June-2020	1.11	Addition of Addendum - List of changes incorporated in v1.11
06-Aug-2020	1.12	Daily transaction MIS
26-Nov-2020	1.13	IMPS 360 API, CIB API under hybrid encryption and VPA Tag addition
12-Dec-2020	1.14	New APIGEE IPs added in composite API
30-Apr-2021	1.15	Status and 360 APIs integrated into composite status API
08-Sep-2021	1.16	IMPS recon API tranref number based search

## Contents

Sr. No.	Description	Page no
<b>1</b>	<b>INTRODUCTION</b>	<b>4</b>
	1.Brief	4
	2.Structure of payments	4
	3.Technical Pre-requisite required from client's end	6
	4. Debits allowed for Nodal Account holding clients	6
	5. Daily transaction MIS	6
<b>2</b>	<b>API DETAILS</b>	<b>7</b>
	1. Composite API	7
	2. Composite status - UPI	19
	3. Composite status - IMPS	22
	5. Status check - NEFT & RTGS	27
	6. NEFT Incremental Status API	29
	7. Beneficiary Registration API	31
	8. CIB Registration API	33
<b>3</b>	<b>CURL COMMAND</b>	<b>35</b>
<b>4</b>	<b>SECURITY</b>	<b>35</b>
<b>5</b>	<b>ENCRYPTION-DECRYPTION PROCESS</b>	<b>35</b>
<b>6</b>	<b>FAQs</b>	<b>39</b>
<b>7</b>	<b>Appendix &amp; Erratum</b>	<b>38</b>

# 1. INTRODUCTION:

## 1. Brief:

In today's digital world, large base of E-commerce companies and Payment Gateway Aggregators companies are moving towards API based payments and collection solutions and drifting away from the existing payment solutions which have a manual intervention or multiple hops in processing of the payments to the beneficiaries

In context to this document, we are introducing the "Composite pay API" that will cater to any of the payout types payments. The umbrella of composite API will also include the status check API and beneficiary services APIs for Nodal account holding merchants. Through this the customer will have all the payment modes in the API with them and wouldn't have to go into further iterations of integration with the Bank. Currently, IMPS, NEFT and UPI are a part of this API.

The API is designed in such a manner that it can accommodate multiple payment modes, has a provision of prioritizing a specific payment mode and the immediate payment modes like IMPS and UPI can also have a fallback option of NEFT or other payment modes once introduced. The client can also choose to use all or any one mode of payment. The Composite Pay API is hosted in the API Gateway of ICICI Bank. APIGW will route any request of Composite Pay API to the respective system of the payment mode depending on the priorities set in the payment request.

In the ERP Banking context, these APIs will be directly integrated into the ERP of the client and the client can directly carry out the transactions from the ERP. The transaction status will be fed back into the ERP as response to the fired API (more details given below). The entire maker-checker leg is at the customer's end without any bank's platform being involved in the process.

## 2. Structure of payments:

- 2.1 CIB Registration\*:

This is a mandatory one-time registration API that is required to access the Beneficiary Registration & Validation API, NEFT and RTGS payment mode APIs. CIB (Corporate Internet Banking) is the internal data-base in ICICI Bank for composite pay API and hence, for processing these APIs credentials are required to be created for the client in the CIB database.

- 2.2 Beneficiary Registration\*\*:

This is the first step for merchants holding the Nodal account. After CIB registration API is fired, this API is required to be fired which shall register beneficiaries on the fly to the CIB database.

- 2.2 Beneficiary Validation\*\*:

The bene validation service is a part of composite pay API and is structured in such a way that when the Composite Pay API is hit by the client, at the bank's end the bene validation service will be called first by APIGW which will check if the bene is registered in the CIB

database. If the validation is successful, the transaction leg is called which processes the transaction basis the payment mode. If the validation fails, the transaction is declined. The client is shown an appropriate error code as response that the beneficiary is not registered and the transaction is declined. Please note, this beneficiary validation leg will only be enabled for the Nodal account holding merchants only.

*\*CIB registration API is required for clients who want to access NEFT, RTGS or beneficiary registration API.*

*\*\*Beneficiary Registration & Validation is mandatory for Nodal account holding merchants. Please refer the section '4. Debits allowed for Nodal account holding clients' in this section below.*

- **2.3 Payments:**

The composite API will first hit the APIGW system from clients ERP system. Based on the priority set by the client, APIGW will route the transaction request to the respective payment mode systems.

- **2.4 Reverse Feed:**

The transaction status will be shared as response within the same API call for IMPS and UPI modes whereas for NEFT and RTGS' mode actual status of the transactions will be available once the transaction batch gets processed. After that, the client is required to fire the status check API which will be provided to the client to fetch the transaction status. These statuses essentially get captured back again in the ERP of the client against the respective transaction.

Please note, even though UPI and IMPS are real-time payment modes, but we recommend following timeline to our clients:

For UPI, the final status of the transaction will be available between 180 secs and T+2 days of initiating the transaction

For IMPS, the final status of the transaction will be available between 180 secs of initiating the transaction.

*(\*The exact timeline is subjective to purging activity at our end. Hence, it is recommended to check with ICICI Bank team before taking any action on back dated transaction. Ideally)*

- **2.5 Status Check:**

If the client doesn't receive the response of a transaction, they can use the Composite Status Check API of the respective payment mode X-priority to fetch it. Please note, until the status is confirmed, the client is not supposed to change the reference number and re-initiate the transaction.

### **3. Technical Pre-requisite required from client's end\*\*\*:**

**1. IP Address:** The IP needs to be whitelisted at our end to ensure the security.

**2. 4096 Bits Public Key Certificate:** Composite API will follow encryption and decryption mechanism to send and receive request & response from client's end. Every certificate file will have two components, i.e. Public Key certificate & Private key certificate. Public Key Certificates are used to encrypt and Private key certificates are used to decrypt.

The client will use ICICI Bank's public key certificate and encrypt the payment request. ICICI Bank will decrypt the payment request using our private key certificate and then process the transaction accordingly as per the payment mode preferred. After the request has been processed, ICICI Bank will encrypt the response using client's public key certificate and post it back. Client will use their corresponding private key certificate to decrypt the response received.

These certificate files are available with third party vendors. A self-signed certificate for UAT is acceptable, but for live environments a CA signed one is mandatory.

*\*\*\*Please note, we require the technical pre-requisites lists in both UAT and Production environment.*

#### **4. Debits allowed for Nodal Account holding clients:**

There are three types of debits allowed on nodal account that is:

1. Refunds/Cashback
2. Vendor payout
3. Commission

Beneficiary Registration and Validation is a mandatory API when the client wants to use the Nodal account for Vendor payouts only. The flow will be such that the client will first fire the Beneficiary Registration API and then include the registered beneficiary details in the composite API request packet and then subsequently fire the composite API. Composite API for Nodal account merchants is structured in such way that they shall first validate the beneficiary before processing the payment request.

#### **5. Daily transaction MIS:**

Daily transaction MIS will be made available on the API developer portal (<https://developer.icicibank.com>). Client is required to register on the API developer portal and share the User ID with business team. They will be able to download day-wise transaction MIS for last 15 days.

Path: Home Page > Account > MIS Download

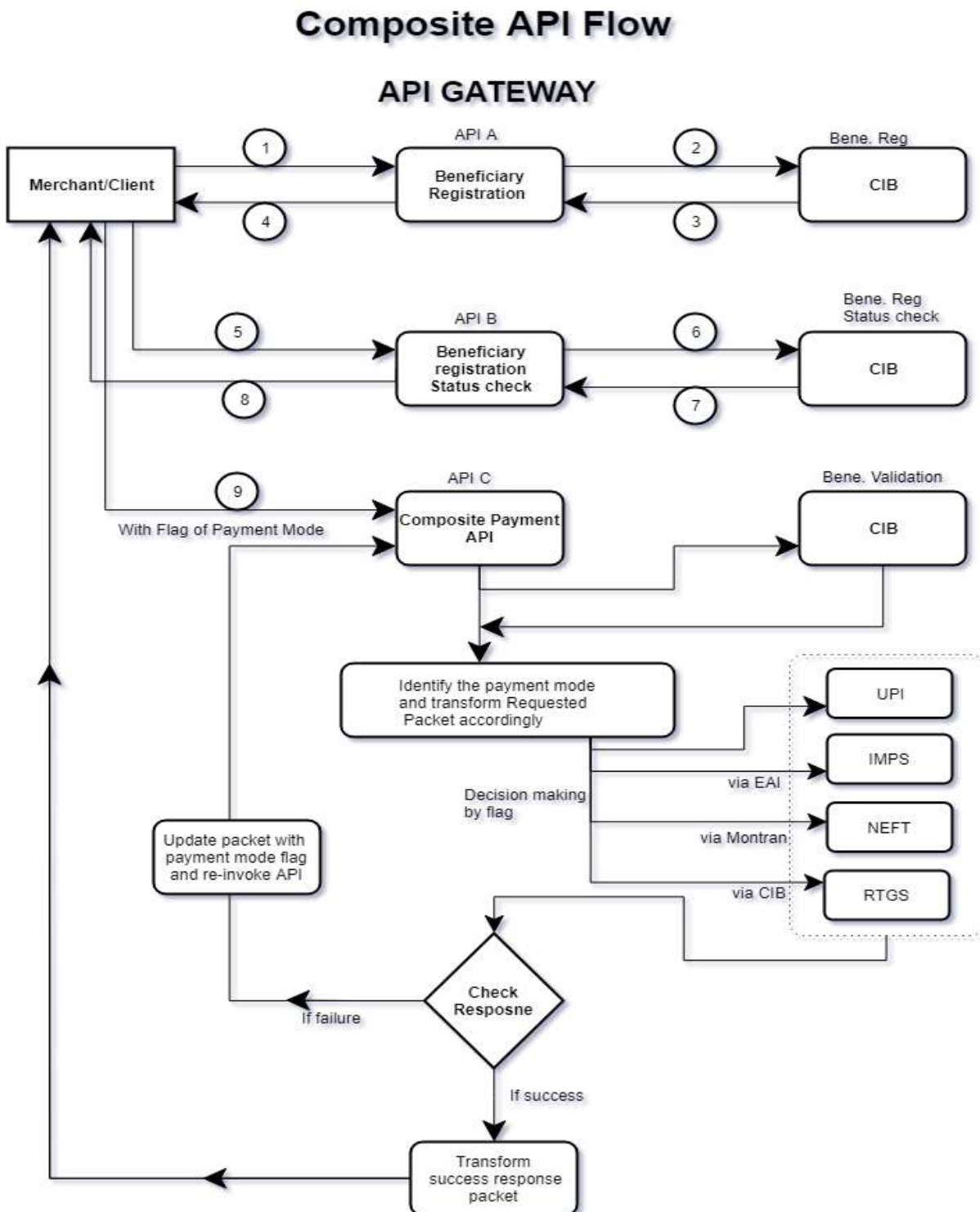
Clients with multiple accounts or merchant IDs can choose to receive a consolidated MIS on one single User ID or multiple MIS on multiple User ID.

## 2. API DETAILS

### 1. Composite Payment API

**1.1 Description:** This API will be used to make payment including priority order to identify and carry out the three types of payment options i.e., UPI, IMPS and NEFT.

**1.2 Flow Diagram:**



### 1.3 API Endpoint:

UAT: <https://apibankingonesandbox.icicibank.com/api/v1/composite-payment>

LIVE: <https://apibankingone.icicibank.com/api/v1/composite-payment>

Nodal Live: <https://apibankingone.icicibank.com/api/v2/composite-payment>

### 1.4 Input Headers

x-priority:- will come in request headers along with apikey.

### 1.5 Input Parameters:

Name	Type (max. char. Limit)	Description	Mandatory (Y-Yes, N-No, O-Optional & C- Conditional)	Provided by ICIC (Y/N)
API Header				
x-priority	String	Priority of Api sequencing eg: - 1000 for UPI (1); IMPS (0); NEFT (0); RTGS (0) 0100 for UPI (0); IMPS (1); NEFT (0); RTGS (0) 1320 for UPI (1); IMPS (3); NEFT (2); RTGS (0) 0001 for UPI (0); IMPS (0); NEFT (0); RTGS (1)	Y	N
UPI				
device-id	String (255)	Unique device Token. Token should be unique for per channel-user.  The channels which are unable to populate the device ID need to send mobile number as the value in this parameter. UPI switch will not be validating the device ID. It will just be forwarded to NPCI as it is a requirement.	Y	Y
mobile	String(10)	Mobile Number of the user	Y	Y
channel-code	String(15)	The code for the source application from which the transaction will be initiated.	Y	Y
profile-id	String(10)	ID of the profile returned in the response of the 'register mobile/store-acc-details' API. This will uniquely identify the user's profile.	Y	Y
seq-no	String(35)	This will be a txn-id generated by the Mobile APP. The value should be a input to the 'txn-id' in the library. This seq-no should be generated using java.util.UUID class.  This should start with 'ICI' and should not have any special characters.	Y	N
account-number	Number(35)	Account Number	C	N



use-default-acc	Char(1)	D=Use default account for transaction N=Use account No and IFSC provided in request for the transaction.	Y	Y
account-type	String(20)	Account type of the mapped account	O	Y
payee-va	String(255)	Alias name with which the payee can be identified by his registered entity.	Y	N
payer-va	String(255)	Alias name with which the payer can be identified by his registered entity.	Y	N
amount	Number(14)	Amount to be collected. (In Rupees, Integer value with 2 decimal) E.g. : 200.00 / 300.12	Y	N
pre-approved	Char(1)	A=Pre-approved, M=MPIN required, P=for M2P	Y	Y
default-debit	Char(1)	Flag 'D' will set this account as default debit account; Flag 'N' will not set this as default account; When use-default-acc has value 'D' then this value should be 'N'; When this value is 'D' then entered account no. will be set as default;	Y	Y
default-credit	Char(1)	Flag 'D' will set this account as default debit account; Flag 'N' will not set this as default account; When use-default-acc has value 'D' then this value should be 'N'; When this value is 'D' then entered account no. will be set as default;	Y	Y
currency	String(3)	Currency code for the transaction; if not passed default value of INR will be considered	O	Y
txn-type	String(50)	If 'merchantToPersonPay' then push payment from merchant to customer's VPA will be initiated; If 'payRequest' then pay to VA will be initiated; If 'payMerchantRequest' then pay to merchant will be initiated; If 'paytoGlobal' then global transaction will be initiated;	Y	Y
remarks	String(50)	Remarks entered by the payer for his reference.	Y	N
mcc	Number(4)	Merchant Category code	Y	Y
merchant-type	String(50)	For merchant transactions, merchant type should be 'ENTITY'	Y	Y
corpID	String	Corporate ID (mandatory only for Nodal account)	C	Y
userID	String	SMS Corporate User (mandatory only for Nodal account)	C	Y
aggrID	String	Aggregator ID (mandatory only for Nodal account)	C	Y

urn	String	URN Number (mandatory only for Nodal account)	C	N
global-address-type	String (20)	MOBILEMMID/ACCOUNTIFSC/AADHAR	C	Y
payee-account	String (20)	Payee account number	C	N
payee-ifsc	String (11)	Payee bank IFSC	C	N
VPA	String	Conditional for Nodal account user when doing VPA based transaction in UPI mode. This has to be same as "payee-va" tag	C	Y

initiation-mode	Numeric(02)	Initiation mode 00=Default, 01=QR Code, 02=Secure QR Code, 03=Bharat QR Code, 04=intent, 05=Secure Intent, 06=NFC, 07=BLE(Bluetooth), 08=UHF(Ultra High Frequency), 09=Aadhaar, 10=SDK, 11=UPI-Mandate, 12=FIR(Foreign Inward Remittance), 13=QR Mandate, 14=BBPS	C	Y
Purpose	Numeric(02)	The purpose field is used specially for SEBI txn 00-Default, 01=SEBI, 02=AMC, 03=Travel, 04=Hospitality, 05=Hospital, 06=Telecom, 07=insurance, 08=Education, 09=Gifting, 10=Other	C	Y
ref-id	String	Reference number Mandatory for creditcard payments	C	Y

#### IMPS

localTxnDtTime	String	Transaction Date & Time (YYYYMMDDHHmmss)	Y	N
beneAccNo	Number(35)	Beneficiary Account Number	Y	N
beneIFSC	String(11)	Beneficiary bank's IFSC Code- 11 digit	Y	N
amount	Number	Transaction amount to be transferred	Y	N
tranRefNo	String	Transaction Reference Number that uniquely identifies the transaction at BC end	Y	N
paymentRef	String(50)	Transaction Info, will be send in NFS message-max 50 char	Y	N
senderName	String(20)	BC customer's Name. This will be used in the message send to NFS, Max 20 char.	Y	N
mobile	String(10)	Remittance Mobile number- Max 10 char	Y	N
retailerCode	String	Retailer/CSP code (BC Specific Retailer/CSP Code).value to be passed as rcode	Y	Y
passCode	String	PassCode been generated by IMPS	Y	Y
bclID	String	Merchant's BCID	Y	Y
crpld	String	Corporate ID (mandatory only for Nodal account)	C	Y
crpUsr	String	SMS Corporate User (mandatory only for Nodal account)	C	Y
aggrld	String (100)	Aggregator ID	Y	Y

#### NEFT

tranRefNo	String (16)	Transaction Reference Number	Y	N
amount	Number (19)	Transaction Amount	Y	N

senderAcctNo	Number (35)	Sender's Bank Account No	Y	N
beneAccNo	Number (35)	Beneficiary Account Number	Y	N
beneName	String (50)	Beneficiary Name	Y	N
benefFSC	String (11)	Beneficiary Bank IFSC Code.  In case of ICICI Bank, this should be 'ICIC0000011'  In case of ICICI Bank enabled cards, this should be 'ICIC0001028' For "VAP" IFSC should be ICIC0000103, ICIC0000104, ICIC0000106	Y	N
narration1	String (35*4)	Narration 1 (Originator of Remittance)	Y	N
narration2	String (35*4)	Narration 2 (Remittance information)	O	N
crpld	String	Corporate ID	Y	Y
crpUsr	String	SMS Corporate User	Y	Y
aggrId	String (100)	Aggregator ID	Y	Y
aggrName	String	Aggregator Name	Y	Y
urn	String (40)	URN Number	Y	N
txnType	String	'TPA' for ICICI as beneficiary Banks; 'RGS' for other banks; For Virtual A/c payments Txn_type should be "VAP" & "RGS" and IFSC should be ICIC0000103, ICIC0000104 & ICIC0000106 depending on the client codes created for the service of virtual account number based collection. This is communicated during setup of this service for any client. IMPS & RTGS txn will not allowed for virtual payments.	Y	Y
WORKFLOW_REQD	String	Conditional for Nodal account user with PWT flow	C	Y
<b>RTGS</b>				
AGGRID	String	Aggregator Id	Y	Y
CORPID	String	Corporation Id	Y	Y
USERID	String	User Id	Y	Y
URN	String	URN Number	Y	N
AGGRNAME	String	Aggregator Name	Y	Y
UNIQUEID	String	Unique Id	Y	N
DEBITACC	Number	Debtor Account no.	Y	N
CREDITACC	Number	Creditor Account no.	Y	N
IFSC	String	Beneficiary Bank IFSC Code.  In case of ICICI Bank, this should be 'ICIC0000011'  In case of ICICI Bank enabled cards, this should be 'ICIC0001028'	Y	N
AMOUNT	Number	Transaction Amount	Y	N
CURRENCY	String	Currency in which transaction held	Y	Y

TXNTYPE	String	'TPA' for ICICI as beneficiary Banks; 'RTG' for other banks;	Y	Y
PAYEENAME	String	Payee name of transaction	Y	N
REMARKS	String	Remark by payee	Y	N
WORKFLOW_REQD	String	Conditional for Nodal account user with PWT flow	C	Y

## 1.6 Sample Request Packets:

### (1) Sample Request Packet for UPI:

```
{
  "mobile": "7000000023",
  "device-id": "190160190160190160190160",
  "seq-no": "ICI5DC866EA6ADC427",
  "account-provider": "74",
  "payee-va": "moto2.4@icici",
  "payer-va": "uat@icici",
  "profile-id": "2995692",
  "amount": "1.00",
  "pre-approved": "P",
  "use-default-acc": "D",
  "default-debit": "N",
  "default-credit": "N",
  "payee-name": "MEHUL",
  "mcc": "6011",
  "merchant-type": "ENTITY",
  "txn-type": "merchantToPersonPay",
  "channel-code": "MICICI",
  "remarks": "none",
  "corpID": "API3",
  "aggrID": "AGGR0008",
  "userID": "USER2",
  "vpa": "moto2.4@icici"
}
```

Please note sample details for UAT testing:

- Please note, corpID, aggrID & userID is only mandatory in case of merchants opting for beneficiary registration & validation API services)
- Vpa tag is only mandatory in case of Nodal account when vpa is [pre](#)-registered through bene registration
- Sequence tag value should always start with a prefix "ICI".

### (2) Sample Request Packet for IMPS

```
{
  "localTxnDtTime": "20190808161038",
  "beneAccNo": "001700000004",
  "beneIFSC": "HDFC0922915",
  "amount": "287.00",
  "tranRefNo": "1236456",
  "paymentRef": "FTTransferP2A",
  "senderName": "Ram singh",
  "mobile": "9999988888",
}
```

```
"retailerCode": "rcode",  
"passCode": "3f4dbbdad1a241bca994339c0d3f3efd",  
"bcId": "IBCFLi00044",  
"crpId": "CIBNEXT",  
"crpUsr": "BALAKIRAN",  
"aggrId": "AGGRID"  
}
```

(Please note, ,aggrId, 'crpId' & 'crpUsr' are only mandatory in case of merchants opting for beneficiary registration & validation API services)

### (3) Sample Request Packet for NEFT

```
{  
  "tranRefNo": "abc12345",  
  "amount": "1",  
  "senderAcctNo": "000451000301",  
  "beneAccNo": "000405002777",  
  "beneName": "Mehul",  
  "beneIFSC": "SBIN0003060",  
  "narration1": "Test",  
  "crpId": "PRACHICIB1",  
  "crpUsr": "USER3",  
  "aggrId": "AGGRID",  
  "urn": "759969775cff4dcc8b8c63402e645da3",  
  "aggrName": "AGGRNAME",  
  "txnType": "RGS",  
  "WORKFLOW_REQD": "N"  
}
```

Please note sample details for UAT testing:

- Debit Account: 000451000301
- Credit Account: 000405002777
- For ICICI as beneficiary bank use IFSC as ICIC0000011 and 'txnType' as "TPA" and for Non-ICICI Beneficiaries use IFSC as DLXB0000092 and 'txnType' as "RGS", virtual accounts "VAP"
- WORKFLOW\_REQD is an optional parameter in Nodal account flow for users with transaction processing is allowed without workflow.

### (4) Sample Request Packet for RTGS

```
{  
  "AGGRID": "AGGRID",  
  "CORPID": "PRACHICIB1",  
  "USERID": "USER3",  
  "URN": "7ef3bc0d03ec495b9f5fa320224ce94e",  
  "AGGRNAME": "AGGRNAME",  
  "UNIQUEID": "ICI01234",  
  "DEBITACC": "000405001611",  
  "CREDITACC": "000405002777",  
  "IFSC": "ICIC00000004",  
  "AMOUNT": "1.00",  
  "CURRENCY": "INR",  
  "TXNTYPE": "RTG",  
  "PAYEENAME": "ASDAS",  
}
```

```
"REMARKS": "Salary for Payroll Jan"  
"WORKFLOW_REQD": "N"
```

```
}
```

Please note sample details for UAT testing:

- Debit Account: 000451000301
- Credit Account: 000405002777
- For ICICI as beneficiary bank use IFSC as ICIC0000011 and 'txnType' as "TPA" and for Non-ICICI Beneficiaries use IFSC as SBIN0003060 and 'txnType' as "RTG".
- WORKFLOW\_REQD is an optional parameter in Nodal account flow for users with transaction processing is allowed without workflow.

#### (5) Sample Request Packet for UPI - IMPS (priority: - 1200)

```
{  
  "device-id": "c16e561d7512ac43",  
  "mobile": "999998888888",  
  "channel-code": "GOOGLE",  
  "profile-id": "7759403",  
  "seq-no": "IC1bb60062b4754450db400209affb30bb3",  
  "mpin": "NPC1,201508zrw==",  
  "account-provider": "508534",  
  "account-number": "AXe1720015d6ce948bae2cc55382925629",  
  "senderIFSC": "ICIC0006246",  
  "account-type": "UNKNOWN",  
  "payee-account": "000123456789",  
  "payee-ifsc": "ICIC00000001",  
  "global-address-type": "ACCOUNTIFSC",  
  "payer-va": "merchant2@icici",  
  "amount": "287.00",  
  "pre-approved": "A",  
  "use-default-acc": "N",  
  "default-credit": "N",  
  "default-debit": "N",  
  "currency": "INR",  
  "txn-type": "paytoGlobal",  
  "remarks": "I paid cash bcz this was pending",  
  "mcc": "6011",  
  "merchant-type": "ENTITY",  
  "ref-id": "12345678",  
  "expire-after": "1439",  
  "deliveryChannelCtrlID": "CIB",  
  "localTxnDtTime": "20170109161038",  
  "beneAccNo": "0017000000004",  
  "beneIFSC": "HDFC9229154",  
  "tranRefNo": "c16e561d7512ac43",  
  "paymentRef": "123456789",  
  "senderName": "GOOGLE",  
  "retailerCode": "0123",  
  "passCode": "0000",  
  "bcID": "IBCFli00044",  
}
```

**(6) Sample Request Packet for UPI - NEFT (priority: - 1020)**

```
{
  "mobile": "7000000023",
  "device-id": "190160190160190160190160",
  "seq-no": "5DC866EA6ADC427",
  "account-provider": "74",
  "payee-va": "moto2.4@icici",
  "payer-va": "uat@icici",
  "profile-id": "2995692",
  "amount": "1.00",
  "pre-approved": "P",
  "use-default-acc": "D",
  "default-debit": "N",
  "default-credit": "N",
  "payee-name": "MEHUL",
  "mcc": "6011",
  "merchant-type": "ENTITY",
  "txn-type": "merchantToPersonPay",
  "channel-code": "MICICI",
  "remarks": "none",
  "tranRefNo": "abc12345",
  "amount": "1",
  "senderAcctNo": "000405001257",
  "beneAccNo": "000451000001",
  "beneName": "Mehul",
  "beneIFSC": "SBIN0003060",
  "narration1": "Test",
  "crpld": "PRACHICIB1",
  "crpUsr": "ABC",
  "aggrId": "AGGR0028",
  "urn": "759969775cff4dcc8b8c63402e645da3",
  "aggrName": "Abc",
  "txnType": "RGS"
}
```

**(7) Sample Request Packet for IMPS - NEFT (priority: - 0120)**

```
{
  "localTxnDtTime": "20190808161038",
  "beneAccNo": "001700000004",
  "beneIFSC": "HDFC9229154",
  "amount": "287.00",
  "tranRefNo": "1236456",
  "paymentRef": "FTTransferP2A",
  "senderName": "Ram singh",
  "mobile": "99999888888",
  "retailerCode": "rcode",
  "passCode": "3f4dbbdad1a241bca994339c0d3f3efd",
  "bclID": "IBCFLi00044",
  "tranRefNo": "abc12345",
  "amount": "1",
  "senderAcctNo": "000405001257",
}
```

```
"beneAccNo": "000451000001",
"beneName": "Mehul",
"benelFSC": "SBIN0003060",
"narration1": "Test",
"crpld": "PRACHICIB1",
"crpUtr": "ABC",
"aggrld": "AGGR0028",
"urn": "759969775cff4dcc8b8c63402e645da3",
"aggrName": "Abc"
"txnType": "RGS"

}
```

#### (8) Sample Request Packet for UPI-IMPS-NEFT (priority: - 1230)

```
{
  "mobile": "7000000023",
  "device-id": "190160190160190160190160",
  "seq-no": "5DC866EA6ADC427",
  "account-provider": "74",
  "payee-va": "moto2.4@icici",
  "payer-va": "uat@icici",
  "profile-id": "2995692",
  "amount": "1.00",
  "pre-approved": "P",
  "use-default-acc": "D",
  "default-debit": "N",
  "default-credit": "N",
  "payee-name": "MEHUL",
  "mcc": "6011",
  "merchant-type": "ENTITY",
  "txn-type": "merchantToPersonPay",
  "channel-code": "MICICI",
  "remarks": "none"
  "localTxnDtTime": "20190808161038",
  "beneAccNo": "001700000004",
  "benelFSC": "HDFC9229154",
  "amount": "287.00",
  "tranRefNo": "1236456",
  "paymentRef": "FTTransferP2A",
  "senderName": "Ram singh",
  "mobile": "99999888888",
  "retailerCode": "rcode",
  "passCode": "3f4dbbdad1a241bca994339c0d3f3efd",
  "bcdID": "IBCFLi00044",
  "tranRefNo": "abc12345",
  "amount": "1",
  "senderAcctNo": "000405001257",
  "beneAccNo": "000451000001",
  "beneName": "Mehul",
  "benelFSC": "SBIN0003060",
  "narration1": "Test",
  "crpld": "PRACHICIB1",
  "crpUtr": "ABC",
}
```



```
"aggrId": "AGGR0028",
"urn": "759969775cff4dcc8b8c63402e645da3",
"aggrName": "Abc"
"txnType": "RGS"
```

```
}
```

## 1.7 Output Parameters:

Name	Type	Description
UPI		
success	Boolean	True/False
response	Number	Response Code
message	String	Response Message
BankRRN	String	Bank RRN number
UpiTranlogId	String	UPI Transaction Log Id
UserProfile	String	User Profile
SeqNo	Number	Sequence No.
MobileAppData	String	Mobile App Data
PayerRespCode	Number	Payee Response Code
IMPS		
Response	String	ActCode Description
ActCode	String	ActCode Value
TransRefNo	Number	Transaction reference no.
BankRRN	String	Bank Transaction Reference Number
BeneName	String	Beneficiary Name
NEFT & RTGS (SUCCESS)		
REQID	Number	Request Id of transaction
RESPONSE	String	Response message of transaction
STATUS	String	Status of transaction
UNIQUEID	Number	Unique transaction id
URN	Number	URN Number of transaction
UTR	Number	UTR number generated by Bank
NEFT & RTGS (FAILURE)		
RESPONSE	String	Response message of transaction
ERRORCODE	String	Error code of the failed transaction
STATUS	String	Status of transaction
RESPONSECODE	String	Response code
MESSAGE	Number	Transaction message

### (1) Response Packet of UPI (Success)

```
{
  "success" : true,
```

```
"response" : "0",
"message" : "Transaction Successful",
"BankRRN" : "821911303721",
"UpiTranlogId" : "1275303721",
"UserProfile" : "15980666",
"SeqNo" : "ICI820f8967824f4fdbb9d11e72b0f7566a",
"MobileAppData" : "SUCCESS",
"PayerRespCode" : "00",
"PayeeRespCode" : "00"
}
```

## (2) Response Packet of IMPS (Success)

```
{
  "success": true,
  "Response": "Transaction Successful",
  "ActCode": "0",
  "TransRefNo": "1238456",
  "BankRRN": "922019796797",
  "BeneName": "BENE & CUSTOMER NA"
}
```

## (3) Response Packet of NEFT (Success)

```
{
  "URN": "2634AB",
  "STATUS": "SUCCESS",
  "UNIQUEID": "312342543",
  "RESPONSE": "SUCCESS",
  "REQID": "275843"
  "UTR": "455849"
}
```

## (4) Response Packet of RTGS (Success)

```
{
  "URN": "2634AB",
  "STATUS": "SUCCESS",
  "UNIQUEID": "312342543",
  "RESPONSE": "SUCCESS",
  "REQID": "275843"
  "UTR": "455849"
}
```

## (5) Response Packet of UPI (Failure)

```
{
  "success" : fail,
  "response" : "39",
  "message" : "duplicate seq no from channel",
  "BankRRN" : "44564564564",
  "UpiTranlogId" : "115151",
  "UserProfile" : "11481217",
  "SeqNo" : "ICla978816e9f914de689791fd35430da6b",
  "MobileAppData" : ""
}
```

## (6) Response Packet of IMPS (Failure)

```
{
  "success" : false,
  "ActCodeDesc": "Duplicate transaction",
}
```

```
"ActCode": "094",  
"TransRefNo": "803112626398"  
}
```

### (7) Response Packet of NEFT & RTGS (Failure)

```
{  
  
  "RESPONSECODE": "100042",  
  "MESSAGE": "Transaction with reference id 346712386 failed during processing, due to System Malfunction",  
  "STATUS": "FAILURE",  
  "ERRORCODE": "106910",  
  "RESPONSE": "FAILURE"  
}
```

### (8) Response packet of Gateway Failure, common to all payment modes (failure)

```
{  
  "success": false,  
  "errorCode": "997",  
  "description": "Note : Initiate Status check after Some time"  
}
```

\*\* Please note, the sample response packet mentioned above for NEFT & RTGS is standard. But, in some failure cases, parameters RESPONSECODE, STATUS & ERRORCODE might not appear. RESPONSE and MESSAGE parameters will always appear.

## 2. Composite Status - UPI

**2.1 Description:** This API will be used to check the transaction status of the UPI Payment.

X-priority - 1000, to be passed in headers for accessing UPI status or recon360

**2.2 Endpoint:**

**UAT:** <https://apibankingonesandbox.icicibank.com/api/v1/composite-status>

**LIVE:** <https://apibankingone.icicibank.com/api/v1/composite-status>

**2.3 Input Parameters:**

Name	Type	Description	Mandatory (Y/N)
device-id	String	Unique device Token. Token should be unique for per channel-user.	Y
Mobile	Number	Mobile Number of the user.	Y
seq-no	String	This will be a txn-id generated by the Mobile APP. This id will be used in the NPCI Common Library at the time of encrypting the OTP/MPIN.	Y
channel-code	String	The code for the source application from which the transaction will be initiated.	Y
profile-id	Number	ID of the profile returned in the response of the 'register mobile/store-acc-details' API. This will uniquely identify the user's profile.	Y
ori-seq-no	String	Seq No of the original transaction for which we are checking the status.	Y
Date	Date	MM/DD/YYYY format, Date to be of transaction initiated date	Y
Recon360	String	Fixed value "Y" or "N" For Value "N", API will be routed to status API, API will fetch the status available over switch For value "Y", API will be routed to Recon API for knowing the Deemed approved or pending transaction status	Y

## 2.4 Request Packet:

### Sample request packet for UPI status enquiry

```
{
  "date": "02/22/2021",
  "recon360": "N",
  "seq-no": "Unique id every time you hit the API",
  "channel-code": "imobile",
  "ori-seq-no": "ICIf5DfdC8c6ddgdcfdffdfhfdj2",
  "mobile": "9956979792",
  "profile-id": "3239272",
  "device-id": "19ee1bee5d349d11"
}
```

### Sample request packet for UPI Recon status

```
{
  "date": "02/22/2021",
  "recon360": "Y",
  "seq-no": "Unique id every time you hit the API",
  "channel-code": "imobile",
  "ori-seq-no": "SBIbb8319eef7c74f16aa8c92640552d2b0",
  "mobile": "9956979792",
}
```

```
"profile-id":"3239272",
"device-id":"19ee1bee5d349d11"
}
```

Please note sample details for UAT testing:

- Client has to pass “N” value in recon360 tag to get the UPI transaction status response and for Recon360 the value to be passed is “Y”
- Client will receive two different responses for status and UPI360.
- Recon360 to be used for “Deemed” cases post using the status API and on T+2 working days of the original transactions.
- Last 5 days transaction status will be present over UPI status enquiry at any point of time.
- 

## 2.5 Output Parameters:

Name	Type	Description
<b>UPI status API response</b>		
Success	String	Tag denoting service, not to be considered for Transaction
Response	Number	Response code of the API. Response code "0" indicates the 'success response'. Not to be considered fir
Message	String	Response code description message
BankRRN	String	The reference number of the transaction.
UpiTranlogId	String	The Transaction ID generated by Switch.
UserProfile	String	Profile Id of the user.
MobileAppData	String	Denotes current transaction status of the RRN
SeqNo	String	Seq-no input parameter will be echoed back.

<b>Recon 360 API response</b>		
PayeeAccount	Numeric	Beneficiary account number
TransactionType	Alphanumeric	Default value as PAY for UPI payouts
PayeeIFSC	Alphanumeric	IFSC of the payee account
OriginatingChannel	Alpha	Default value “ <b>UPI.NPCI</b> ”
OriginalTransactionStatus	Alpha	Original transaction status
PayeeName	Alpha	Payee Name
PayerIFSC	Alphanumeric	Payer IFSC
PayerName	Alpha	Payer Name
PayerVPA	Alphanumeric	Payer VPA
MerchantID	Alphanumeric	Merchant ID
PayeeVPA	Alphanumeric	Payee VPA
RRN	Numeric	RRN
TransactionAmount	Numeric	Transaction Amount
PayerAccountNumber	Numeric	Payer Account Number
DateTimeStatusChange	Alphanumeric	Date and Time of Status Change

CurrentTransactionStatus	Alphanumeric	Current Transaction Status/updated transaction status of Original transaction Success responses: Credit confirmation received, Success, Completed Failure response: Failed and Returned received Pending response: Deemed approved, Suspect, Timeout
SequenceNumber	Alphanumeric	Original Sequence number
IRC	Numeric	IRC
DisplayMessage	Alphanumeric	Display Message on API call
OriginalTransactionDateTime	Alphanumeric	Original Transaction Date and Time

## 2.6 Response Packet:

UPI Status API Response:

```
{
  "success": true,
  "response": "0",
  "message": "Transaction Successful",
  "BankRRN": "104714151385",
  "UpiTranlogId": "304844161",
  "UserProfile": "2996298",
  "SeqNo": "123",
  "MobileAppData": { "original-txn-rrn":
106719317736
  "original-txn-response-code": 0
  "original-txn-message": "SUCCESS"
}
  "PayerRespCode": "00",
  "PayeeRespCode": "00"
}
```

UPI recon360 API response:

```
{
  "PayeeAccount": "XXXXX7639",
  "TransactionType": "PAY",
  "PayeeIFSC": "SBIN0003160",
  "OriginatingChannel": "UPI.NPCI",
  "OriginalTransactionStatus": "Deemed Approved",
  "PayeeName": "Mr S VIMALAN",
  "PayerIFSC": "",
  "PayerName": "Vignesh .",
  "PayerVPA": "vignesh18895@oksbi",
  "MerchantID": "SBIbb8319eef7c74f16aa8c92640552d2b0",
  "PayeeVPA": "vimalan281096@okicici",
  "RRN": "000320706502",
  "TransactionAmount": "500",
  "PayerAccountNumber": "XXXXXXXXXXXX4976",
  "DateTimeStatusChange": "2020-01-29 15:14:32.0",
  "CurrentTransactionStatus": "Credit Confirmation Received",
  "SequenceNumber": "SBIbb8319eef7c74f16aa8c92640552d2b0",
}
```

```
"IRC": "0000",
"DisplayMessage": "Success. 223813",
"OriginalTransactionDateTime": "2020-01-03 20:38:48.0"
}
```

### 3. Composite Status- IMPS

**3.1 Description:** This API will be used to check the transaction status for IMPS.

**3.2 API Endpoint:**

**UAT:** <https://apibankingonesandbox.icicibank.com/api/v1/composite-status>

**Live:** <https://apibankingone.icicibank.com/api/v1/composite-status>

**3.3 Input Parameters:**

Name	Type	Description	Mandatory (Y/N)
transRefNo	Number	Transaction Reference Number that uniquely identifies the transaction at BC end	Y
Passcode	String	PassCode been generated by IMPS	Y
bcID	String	Merchant's BCID	Y
Channel-code	Alphanumeric	Channel code	Conditional
Date	Date	MM/DD/YYYY format, Date has to be transaction initiated date	Y
Recon360	String	Fixed value "Y" or "N" For Value "N", API will be routed to status API, API will	Recon360

**3.4 Request Packet:**

IMPS Status enquiry request packet:

```
{
  "transRefNo": "134255",
  "passCode": "3f4dbbdad1a241bca994339c0d3f3efd",
  "bcID": "IBCFli00044",
  "recon360": "N",
  "date": "02/16/2021"
}
```

IMPS recon 360 request packet:

```
{
  "date": "02/16/2021",
  "recon360": "Y",
  "transRefNo": "8442310086",
}
```

}

Please note sample details for UAT testing:

- Client has to pass “N” value in recon360 tag to get the IMPS transaction status response and for Recon360 the value to be passed is “Y”
- Recon360 to be used for error codes 11&30 post using the status API and on T+2 working days of the original transactions.

### 3.5 Output Parameters:

Name	Type	Description
IMPS status enquiry response		
ActCode	Number	Response code of the API. Response code "0" indicates the success response.
Response	String	Transaction Response
BankRRN	Number	Bank RRN number
BeneName	String	Beneficiary Name
TranRefNo	Number	Transaction Reference Number that uniquely identifies the transaction at BC end
PaymentRef	String	Payment Reference
TranDateTime	Date	Transaction Date and time
Amount	Number	Amount
BeneMMID	Number	Beneficiary MMID
BeneMobile	Number	Beneficiary Mobile Number
BeneAccNo	Number	Beneficiary Account Number
BeneIFSC	String	Beneficiary IFSC
RemMobile	Number	Sender Mobile Number
RemName	String	Sender Name
RetailerCode	String	Retailer Code
IMPS recon 360 response parameters		
TranRefNo	Number	Transaction Reference Number that uniquely identifies the transaction at BC end
PaymentRef	String	Payment Reference
Display Message	String	Transaction original status
RRN	Number	Bank RRN number
OriginalTransactionStatus	Number	Original Transaction Status



CurrentTransactionStatus	String	Current Transaction Status/updated transaction status Success responses: Credit confirmation received, Success, Completed Failure responses: Failed, Returned received. Pending responses: Deemed approved, Suspect, Timeout
OriginalTransactionDate Time	String	DDMMYYYY
TransactionAmount	Number	Transaction Amount
BeneficiaryName	String	Beneficiary Name
BeneficiaryAccountNumber	Number	Beneficiary Account Number
BeneficiaryMobileNo	Number	Beneficiary Mobile No
BeneficiaryAadharNumber	Number	Beneficiary Aadhar Number
BeneficiaryIFSC	String	Beneficiary IFSC
Remitter Name	String	Alphanumeric(50) Remitter Name
RemitterAccountNumber	Numeric	Remitter Account Number
RemitterMobileNumber	Numeric	Remitter Mobile Number
OriginatingChannel	String	Originating Channel

### 3.6 Response Packet:

IMPS status enquiry response:

```
{
  "ImpsResponse": {
    "ActCode": "11",
    "Response": "Time out at NPCI",
    "BankRRN": "921011806647",
    "BeneName": "",
    "TranRefNo": "3247197400487395",
    "PaymentRef": "FTTransferP2A",
    "TranDateTime": "29-07-2019 11:37:13",
    "Amount": "2000",
    "BeneMMID": "9028001",
    "BeneMobile": "",
    "BeneAccNo": "06340110001646",
    "BeneIFSC": "UCBA0000634",
    "RemMobile": "8800683242",
    "RemName": "ACTAS TECHNOLOGIES PRIVATE LIMITED",
    "RetailerCode": "rcode"
  }
}
```

IMPS recon 360 response:

```
"recon-api-response": {
  "headers": {
    "message-type": 1001,
    "proc-code": 455002,
    "channel-code": "mobile",
```

```
"stan": "STAN000002",
"response-code": "0000",
"response-message": "Success."
},
"parameters": {
  "index": {
    "id": 0,
    "parameter": [
      {
        "name": "BeneficiaryMobileNo",
        "value": {}
      },
      {
        "name": "RemitterAccountNumber",
        "value": "XXXXXXXX0190"
      },
      {
        "name": "BeneficiaryIFSC",
        "value": "ABHY0065032"
      },
      {
        "name": "OriginatingChannel",
        "value": "Internet"
      },
      {
        "name": "OriginalTransactionStatus",
        "value": "Suspect"
      },
      {
        "name": "BeneficiaryName",
        "value": {}
      },
      {
        "name": "BeneficiaryAadharNumber",
        "value": {}
      },
      {
        "name": "RRN",
        "value": "003813455317"
      },
      {
        "name": "TransactionAmount",
        "value": 1.84
      },
      {
        "name": "BeneficiaryAccountNumber",
        "value": "XXXXXXXXXXXX2259"
      },
      {
        "name": "MobileNumber",
        "value": 8898526409
      },
      {
```

```
        "name": "CurrentTransactionStatus",
        "value": "Credit Confirmation Received"
    },
    {
        "name": "TranRefNo",
        "value": 8442310086
    },
    {
        "name": "IRC",
        "value": "0091"
    },
    {
        "name": "DisplayMessage",
        "value": "Time out at NPCI"
    },
    {
        "name": "OriginalTransactionDateTime",
        "value": "12-FEB-2000.00.00"
    },
    {
        "name": "RemitterName",
        "value": "Paytm01"
    }
    ]
}
}
```

## 4. Status Check for NEFT & RTGS

NEFT transaction processing essentially takes place in two steps - a) There is a debit in to remitter account first, b) Followed by transaction getting submitted to RBI for processing with the time lag.

For NEFT status enquiry there are two sets of APIs available from ICICI bank given below as -

- 1) NEFT debit confirmation API - Provides the status of the debit in remitter a/c
- 2) NEFT credit confirmation API - Provides the status of the credit in beneficiary a/c

### Status Enquiry -

#### a. For money transferred to ICICI Bank account bene -

For the amount transferred within ICICI bank the remitter needs to only use the first API ie. Debit confirmation API.

The remitter needs to fire this API and if the response received in status field is “SUCCESS” in that case the transaction can be marked as successfully processed.

For the status received as failure(Response tag value to be success) the same can be marked as failed and that particular transaction can be re-initiated. For any other status remitter should recheck the status after 2 hours. This re-check is required to be done only once. In case the status does not change to a definite success or failure, in that case no further action is possible and this transaction is to be marked as ‘pending’.

All pending transactions are to be reconciled with the remitter bank statement to ascertain the final status of the transaction. In case of no respective debit is found the transaction can be marked a failed & fresh transaction can be re-initiated. In case the debit is present in the account the transaction can be marked as **successful**.

Additionally, this API can also be utilized to fetch the UTR number for the transaction in case the same is not received with initial

#### **b. For money transferred to non ICICI Bank account bene -**

It is a two-step process to identify the final status for these transaction.

1. The remitter needs to fire the first API (Debit) and if the response received in status field is “SUCCESS” only and only in that case the remitter should proceed to step 2 listed below.

For the status received as failure the same can be marked as failed and that particular transaction can be re-initiated. **For any other status remitter should recheck the status after 2 hours.** This re-check is required to be **done only once**. In case the status does not change to a definite success or failure, in that case no further action is possible and this transaction is to be marked as ‘pending’.

All pending transactions are to be reconciled with the remitter bank statement to ascertain the final status of the transaction. In case of no respective debit is found the transaction can be marked a failed & fresh transaction can be re-initiated. **In case the debit is present in the account then the remitter can proceed to step 2.**

2. Under step 2, the remitter should invoke the second API (Credit API or Incremental status API) with the requisite request packet as documented in API document. Once the status is successfully received as “amount credited to beneficiary” transaction can be marked as **successful**.
3. It is possible that in some scenarios the status is received as “posted to RBI”. This status is provided for the transaction which the beneficiary bank has not provided the ultimate status and after 5hrs of the transaction the status would be changed “paid”. In such a scenario RBI has provisioned for 72 hrs for the beneficiary bank to update the final status of the transaction. However, if the status does not change post 72 hrs the remitter can mark these transactions as successful.

#### **4.1. Description: RTGS response received over composite status API is a terminal response.**

## 4.2. Endpoint URL:

UAT: <https://apibankingonesandbox.icicibank.com/api/v1/composite-status>

Live: <https://apibankingone.icicibank.com/api/v1/composite-status>

## 4.3. Input Header:

x-priority = '0010' for NEFT and '0001' for RTGS

## 4.4. Input Parameters:

AGGRID	String	Aggregator Id	Y
CORPID	String	Corporation Id	Y
USERID	String	User Id	Y
URN	String	URN Number	Y
UNIQUEID	String	Unique Id (Pass the value of 'tranRefNo' in this parameter for NEFT status check)	Y

Sample Request:

```
{
  "AGGRID": "AGGR0001",
  "CORPID": "PRACHICIB1",
  "USERID": "USER3",
  "URN": "1288883",
  "UNIQUEID": "abc12dk3d7dd4hc5ducdfc"
}
```

## 4.5. Output Parameters:

URN	Number	URN Number of transaction
STATUS	String	Status of transaction
UNIQUEID	Number	Unique transaction id
RESPONSE	String	Response message of transaction
UTRNUMBER	Number	UTR Number of the transaction

Sample Response:

```
{ "XML": {
  "STATUS": "SUCCESS",
  "URN": 1288883,
  "UNIQUEID": "abc12dk3d7dd4hc5ducdfc",
  "UTRNUMBER": "056931871",
  "RESPONSE": "SUCCESS"
}}
```

```
}
```

## 5. Incremental Status API for NEFT

**5.1. Description :** To be used after 30 minutes of original transaction once the NEFT batch is processed. API gives the terminal status of transactions for txntype “RGS”

**Note:** it is mandatory to use status API before firing incremental API.

### 5.2. Endpoint URL:

**UAT:** <https://apibankingonesandbox.icicibank.com/api/v1/CIBNEFTStatus>

**Live:** <https://apibankingone.icicibank.com/api/v1/CIBNEFTStatus>

### 5.3. Input Parameters:

Name	Type	Description
AGGRID	String	Aggregator Id
CORPID	String	Corporation Id
USERID	String	User Id
UTRNUMBER	String	UTR Number
URN	String	URN Number

### 5.4. Sample request

```
{
  "URN": "2634AB",
  "AGGRID": " Cust0XXX ",
  "CORPID": "ICICXXX",
  "USERID": "Deepak",
  "UTRNUMBER": "455849"
}
```

### 5.5. Output Parameter

Name	Type	Description
STATUS	String	Status of the UTR. This is the field which needs to be referred, below value to be considered for final status Success: Amount credited to Beneficiary

		Failure: Amount refunded to Remitter Pending: paid Retry with valid details: FAILURE, Invalid Transaction ID, UTR missing
UTRNUMBER	String	UTR Number
CreditDate	String	Date of Credit
Response	String	Success/Failure
REASON	String	Reason for transaction failure applicable only in case of a failure transaction

## 2.4 Sample Response

```
{
  "STATUS": "Amount credited to Beneficiary.",
  "UTRNUMBER": "XXXXXXXXXX",
  "CreditDate": "DD,MM,YYYY HH:MM:SS AM/PM",
  "Response": "SUCCESS",
  "REASON": ""
}
```

### When Txn id or UTR no is not exist

```
{
  "STATUS": "FAILURE",
  "ErrorCode": "999922",
  "Message": "Invalid Transaction Id.",
  "Response": "Failure"
}
```

## 6. API Name: Beneficiary Registration

- 6.1. Description:** Beneficiary Registration API is mandatory for Merchants transacting from Nodal Account. Hence, before any composite API request, the clients are required to first fire this API. Our systems will configure the clients in such a way that first we shall validate the beneficiary registration, and then process the payment request.

There are two such APIs. One registers the beneficiary on the basis of account number & IFSC which would suffice the payment needs for IMPS, NEFT & RTGS modes. Other API registers beneficiary on the basis of Virtual Payment address which will suffice payment needs for UPI mode.

For clients with existing ALIAS ID, the default configuration for this API will not work. Client needs to delete their existing ALIAS ID through CIB portal and create a new ALIAS ID similar to the existing corporate user ID. For example, a client has ALIAS ID as 'ICICIBank' and corporate user ID as 'Bank12345', then client needs to delete the existing ALIAS ID i.e. 'ICICIBank' through CIB portal and create a new ALIAS ID as 'Bank12345' which has to be similar to the corporate user ID.

## 6.2. Beneficiary Registration through account & IFSC

### 7.2.1 API Endpoint:

**UAT:** <https://apibankingonesandbox.icicibank.com/api/Corporate/CIB/v1/BeneAddition>

**Live:** <https://apibankingone.icicibank.com/api/Corporate/CIB/v1/BeneAddition>

### 7.2.2 Input Parameters:

Name	Type (Max char limit)	Description	Mandatory (Y/N)
Crpld	NVARCHAR2 (32)	Client ID in Corporate Internet Banking.	Y
CrpUsr	NVARCHAR2 (32)	Customer ID under Client ID in Corporate Internet Banking	Y
BnfName	NVARCHAR2 (80)	Beneficiary name.	Y
BnfNickName	NVARCHAR2 (80)	Beneficiary Nick name. Unique for every request.	Y
BnfAccNo	NVARCHAR2 (32)	Beneficiary Account number	Y
PayeeType	CHAR (1)	Client need to pass respective Payee type for bene registration 'O' for other banks & 'W' for within bank WIB. Y	Y
IFSC	NVARCHAR2 (32)	Bene user IFSC. But for FT static IFSC should be pass by client ICIC0000011	Y
AGGR_ID	NVARCHAR2 (100)	ID of the partner to be provided and hardcoded by ICICI Bank.	Y
URN	NVARCHAR2 (40)	This a unique value that partner will assign to each registration from his end for security and recon.	Y

### 7.2.3 Sample Request:

```
{
  "Crpld" : "PRACHICIB1",
  "CrpUsr": "USER4",
  "BnfName" : "AnshumanMishra",
  "BnfNickName" : "Ansh",
  "BnfAccNo" : "000405001257",
  "PayeeType" : "W",
  "IFSC" : "ICIC0000011",
  "AGGR_ID" : "AGGR0002",
  "URN" : "3kCy4sPuSqDNi4kggXJIoE568221"
}
```

### 7.2.4 Output Parameters:

Name	Type	Description
Message	NVARCHAR2	Status message of request



Response	NVARCHAR2	Response SUCCESS OR FAILURE.
ErrorCode	String	Error Code

### Response:Success

```
{
  BNF_ID:"4836",
  Message:"You may not be able to make fund transfer immediately with newly added payee, please
  check payee status before proceeding",
  Response:"SUCCESS"
}
```

### Response:Failure

```
{
  Message:"Invalid Corp Id or User Id or Aggregator Id is passed.",
  ErrorCode:"9906",
  Response:"Failure"
}
```

## 7.3 Beneficiary Registration API through VPA

**7.3.1 Description:** This API will register beneficiary on the basis of VPA essential for UPI payment mode for Nodal account holding merchants.

### 7.3.2 API Endpoint:

**UAT:** <https://apibankingonesandbox.icicibank.com/api/Corporate/CIB/VPA/v1/BeneAddition>

**Live:** <https://apibankingone.icicibank.com/api/Corporate/CIB/VPA/v1/BeneAddition>

### 7.3.3 Input Parameters:

Name	Type (Max char limit)	Description	Mandatory (Y/N)
AGGRID	String	Aggregator Id	Y
CORPID	String	Corporation Id	Y
USERID	String	User Id	Y
URN	String	URN Number	Y
VPA	String	ALIAS ID	Y

### 7.3.4 Request Packet:

```
{
  "aggrID" : "PRACHICIB1",
  "corpID":"USER4",
  "userID" : "AnshumanMishra",
  "urn" : "1234567",
  "vpa" : "test0@icici",
}
```

### 7.3.4 Output Parameters:

Name	Type (Max char limit)	Description
Message	String	Message
Response	String	Response will be 'Failure' or 'Success'
ErrorCode	String	Error Code

### 7.3.5 Request Packet:

#### Response:Success

```
{
  BNF_ID:"4836",
  Message:"You may not be able to make fund transfer immediately with newly added
  payee, please check payee status before proceeding",
  Response:"SUCCESS"
}
```

#### Response:Failure

```
{
  Message:"Invalid Corp Id or User Id or Aggregator Id is passed.",
  ErrorCode:"9906",
  Response:"Failure"
}
```

## 7. CIB Registration API

**7.1. Description:** This is a one-time API which is required to access NEFT, RTGS & Beneficiary Registration APIs.

### 7.2. API Endpoint:

**UAT:** <https://apibankingonesandbox.icicibank.com/api/Corporate/CIB/v1/Registration>

**Live:** <https://apibankingone.icicibank.com/api/Corporate/CIB/v1/Registration>

#### Input Parameters:

Name	Type (Max char limit)	Description	Name
------	-----------------------	-------------	------

AGGRNAME	String	Aggregator name	Y
AGGRID	String	Aggregator Id	Y
CORPID	String	Corporation Id	Y
USERID	String	User Id	Y
URN	String	URN Number	Y
ALIASID	String	ALIAS ID	Y

### 7.3. Sample Request

```
{
  "CORPID": "<Corp Id>"
  "USERID": "<user id>"
  "AGGRNAME": "<Partner
name>"
  "AGGRID": "<Partner
id>"
  "URN": "<URN>"
  "BANKID": "ICI"
}
```

### Output Parameters:

Name	Type (Max char limit)	Description
AGGRNAME	String	Aggregator name
AGGRID	String	Aggregator Id
CORPID	String	Corporation Id
USERID	String	User Id
URN	String	URN Number
Response	String	Response
Message	String	Message

## 3. CURL COMMAND

Sample Curl command for only UPI:

```
curl -X POST \
https://apigwuat.icicibank.com:8443/api/v1/composite-payment \
-H 'apikey: <apikey>' \
-H 'content-type: application/json' \
-H 'x-priority: 1000' \
-d '{
  "device-id": "a2fa806f843b3538",
  "mobile": "9046780520",
  "channel-code": "GOOGLE",
  "profile-id": "1147846",
  "seq-no": "ICl6a732b2eb1264b1f8da0fb62ed9116ae",
  "pre-approved": "M",
  "payer-va": "ankan.k.e.s.h0045-40@okicici",
  "amount": "120.00",
  "remarks": "UPI",
  "account-provider": "508534",
  "account-number": "AX61c7d43c47fc0c513ad9283dc0539d90",
  "senderIFSC": "ICIC0001937",
  "account-type": "SAVINGS",
  "use-default-acc": "N",
  "default-credit": "N",
  "default-debit": "N",
  "currency": "INR",
  "ref-url": "https://www.example.org",
  "payee-va": "good@icici",
  "txn-type": "payRequest"
}'
```

## 4. SECURITY

- API Key needs to be passed in the header (apikey) and merchant IP will also be required for IP whitelisting.
- **API Key:** *to be shared by ICICI team.*
- API request and response to Merchant is secured using advanced and agreed upon encryption algorithm agreed to maintain data confidentiality and integrity.
- API Gateway uses the standard authenticating and authorizing process for the incoming request from merchant and for maintaining the integrity and confidentiality we apply state of art Encryption/ Decryption algorithm.
- Please refer below given steps for Encryption & Decryption process.

### 4.1 Encryption & Decryption Process

Sample Encrypted Request to be sent to Client (JSON):

```
{
  "requestId": "",
  "service": "PaymentApi",
  "encryptedKey":
  "oG5mU1JJNBuwQaSLKb3wfrZks/ cT2Vo2yBNNuqjNHDWEC144WxC8iKqBpJAgq7reFKC4sHNUmNPRDya1AvmQ7x1L+3EA
  dEs9FEWNurZuWTVzPk4y7JrGhg0rz9KptBf+JfJUKSMo7NR3Saxel6EYtckkDr3AGW7WJZmhcEoAMMXRws/hLVmaNHC/nOj
  CNqqBd4lOOAzdJh/HADRVl+YAJKT8dE4x9NTl+UX1zAooWhza+TsWEHfzxQJa7zai7WSa/wiJD3uD7mk5vT1WY/fKJBquCuz
  M7l35vigDhmb7dLVLuX8VMiNQrtErWNI0uVaag1jg+uZUtyDSxjPFi5yEpKVvc7+T503lDnCvkCFDyqgasDsPL24qOjYk4XavTZv
  wGuPAdYNNkVnLzVElEhg4zS2ye+fa/8fZiMt/3fwYeN9dgn9i5R6VOFbXSuZJYPSci9k0oqz73h1nzFtps60rUEDoGikGvm9waJ
  U3W78VH5mldGfGvvJjiKluVHmi/huzEX9v4w3mW7RDGgmOuKlImkqki+XWgyB0JvVmsLdO+cBaym/seZP3+zdfhO9AWSI2t
  DLD4Vf0jDjzoDSFN2mzUFgHK9mbtbXgvsNReoGqx/KsivzmZNLmDmtg8eR4Z9LnLnI4rl4OtkDv5y/mxMtL3MBUUUajkw6OS
  6NnhEG895yo=",
  "oaepHashingAlgorithm": "NONE",
  "iv": "",
  "encryptedData":
  "wBJSeFsnJVlobh1cJR553w6Ay6b8/2frCjxvdZ1Bsnxztul7Ha8lFl4PoZD+IhdLRShWdKgZ3yJYlisGV/KKpyMSY3DILOpbkqEa
  0Qq0g=",
  "clientInfo": "",
  "optionalParam": ""
}
```

### 4.2 Field Description:

Field	Description	Data Type	Max length	Mandatory	Permitted Values
requestId	Unique identifier for the request. Not being stored at any level.	String	64	No	

service type	Service Name ; Identifying the backend service name	String		Yes	
encryptedKey	One-time use AES key encrypted by the Client's public key. Requirement is for a 128-bit key (with 256-bit key supported as an option).	String. Base64- encoded data (case- insensi- tive)		Yes	
oaepHashingAlgo- rithm	Describes the algorithm used for Asymmetric Encryption of one time AES key.	String	6	Yes	Value : NONE, Meaning : RSA/ECB/PKCS1 is used for Asymmetric Encryption Value : SHA1, Meaning : RSA/NONE/ OAEPWithSHA1AndMGF 1Padding OR RSA/ECB/ OAEPWithSHA1AndMGF 1Padding

lv	The initialization vector used when encrypting data using the one-time use AES key.	String. Base64-encoded data (case-insensitive)	24	Yes, if IV is not part of encrypted data itself. Leave blank otherwise. For the response data encrypted at ICICI end, IV will always be part of encryptedData itself and it is randomly generated for each request. Can be retrieved by Client by retrieving the first 16 bytes of Base64 decoded encryptedData	Exactly 16 bytes actual value to match the block size
encryptedData	Contains the encrypted dataPayload object containing the business information. Encrypted by the ephemeral AES key using AES/CBC/PKCS5Padding. Sample unencrypted object: Please refer first section of document for a sample object.	String. Base64-encoded data (case-insensitive)		Yes	
clientInfo	ClientIP or other information	String		No	
optionalParam	Reserved for future use	String		No	

## 5. ENCRYPTION-DECRYPTION PROCESS

### 5.1 For Composite Payment, Status Check APIs and CIB APIs

For encryption of request at ICICI:

```
SesionKey = Randomly generated string of length 16 (OR 32).  
encryptedKey = Base64Encode(RSA/ECB/PKCS1Encryption(SesionKey,ICICIPubKey.cer))  
IV = Initialization Vector- Exactly 16 bytes actual value to match the block size  
encryptedData = Base64Encode(AES/CBC/PKCS5Padding(Request,SessionKey, IV))
```

For encryption of response at ICICI:

```
encryptedKey = Base64Encode(RSA/ECB/PKCS1Encryption(SesionKey,ClientPubKey.cer))  
Session key is nothing but randomly generated string of length 16 (OR 32).  
encryptedData = Base64Encode(AES/CBC/PKCS5Padding(Response,SessionKey))
```

For decryption of response at Client:

```
IV= getFirst16Bytes(Base64Decode(encryptedData)  
SessionKey = Base64Decode(RSA/ECB/PKCS1Decryption(encryptedKey,ClientPrivateKey.p12,))  
Session key is nothing but randomly generated string of length 16 (OR 32) .  
Response = Base64Decode (AES/CBC/PKCS5Padding Decryption(encryptedData,SessionKey, IV))  
Or
```

Steps for Encryption

- 1) Generate 16 digit random number. Say RANDOMNO.
- 2) Encrypt RANDOMNO using RSA/ECB/PKCS1Padding and encode using Base64. Say encryptedKey.
- 3) Perform AES/CBC/PKCS5Padding encryption on request payload using RANDOMNO as key and iv- initialization vector. Say encryptedData.
- 4) Now client may choose to send IV in request from one of below two options.
  - a. Send Base64 Encoded IV in "iv" tag. (Recommended Approach)
  - b. Send IV as a part of encryptedData itself.

```
bytes[] iv = IV;  
bytes[] cipherText = symmetrically encrypted Bytes (step3)  
bytes[] concatB = iv + cipherText;  
encryptedData = B64Encode(concatB);
```

- 5) Perform AES/CBC/PKCS5Padding encryption on DATA using RANDOMNO as key and Base64encoded RANDOMNO as IV. Say ENCR\_DATA.

Steps for Decryption

- 1) Get the IV- Base64 decode the encryptedData and get first 16 bytes and rest is encrypted response.  
bytes[] IV= getFirst16Bytes(Base64Decode(encryptedData))
- 2) Decrypt encryptedKey using algo (RSA/ECB/PKCS1Padding) and Client's private key.
- 3) Decrypt the response using algo (AES/CBC/PKCS5Padding).
- 4) Ignore first 16 bytes of response, as it contains IV.

## 6. FAQs

### 1. What is composite API?



Ans: One single API which will accommodate payments for UPI, IMPS, NEFT & RTGS.

**2. How do you know the status of the transaction if there is no response received for the fired API?**

Ans: There are status check API.

**3. How do you know the preferred payment mode in the request?**

Ans: Through Input Header 'x-priority'.

**4. What is 'x-priority'?**

Ans: 'x-priority' is a 4 digit input header parameter which determines the payment mode. The first, second, third and fourth digit sets the priority for UPI, IMPS, NEFT & RTGS respectively.

**5. Can we process a single payment mode?**

Ans: Yes. Put the value '1' against the preferred payment mode and rest all as '0'. Therefore, in case of just UPI, IMPS, NEFT & RTGS, x-priority shall be 1000, 0100, 0010 & 0001.

**6. Can we process more than one payment modes?**

Ans: Yes. In case, UPI is preferred and IMPS is fallback, then x-priority shall be '1200'. Similarly, 1020 is UPI preferred and NEFT as fallback and so on.

**7. Will the request parameter change for more than one payment mode API?**

Ans: Yes. The request packet needs to include the mandatory parameters for all the payment modes selected.

**8. Is Beneficiary Registration API Mandatory?**

Ans: No. It is only mandatory for clients transacting from Nodal account.

**9. How do you do the status check of Beneficiary Registration API?**

Ans: Fire Beneficiary registration API again, and you'll receive a response, 'Beneficiary already registered'.

**10. How much time does beneficiary registration API takes to register the beneficiaries?**

Ans: Beneficiary Registration API is real time API. As soon as the API response is received, Beneficiary is registered on the Data base and the transfer API can be processed.

**11. What is CIB registration API?**

Ans: This is a one-time API that registers client on CIB which is our internal system. This is required for clients to use the Beneficiary registration, NEFT & RTGS APIs.

**12. How approval is done for CIB registration in UAT or production.**

Ans: In UAT approval is done at back end post firing the registration API. In production client has to login CIB portal, select connected banking and approve from pending approvals.

**13. What is required from clients' end for testing?**

Ans: IP addresses & 4096 Bits Public key certificate.

**14. Why is IP address required?**

Ans: In order to maintain the security, we whitelist client's IP addresses.

**15. What is 4096 Bits Public key certificate?**

Ans: Composite API will follow encryption and decryption mechanism to send and receive request & response from client's end. Every certificate file will have two components, i.e. Public Key certificate & Private key certificate. Public Key Certificates are used to encrypt and Private key certificates are used to decrypt.

The client will use ICICI Bank's public key certificate and encrypt the payment request. ICICI Bank will decrypt the payment request using our private key certificate and then process the transaction accordingly as per the payment mode preferred. After the request has been processed, ICICI Bank will encrypt the response using client's public key certificate and post it back. Client will use their corresponding private key certificate to decrypt the response received.

These certificate files are available with third party vendors. A self-signed certificate for UAT is acceptable, but for live environments a CA signed one is mandatory.

**16. What are the details provided by ICICI Bank after the configuration is completed?**

Ans: We provide the API Key & ICICI Bank's Public key certificate separately for both UAT & production.

**17. What is API Key?**

Ans: 'API Key' is a unique value generated for each client which is required to be passed in the header.

**18. What is Device-ID, profile-ID, channel code, account provider, merchant type, etc. in UPI request packet?**

Ans: These are fixed values which are generated when the client is configured in UAT & Production environment and the same shall be provided after the configuration.

**19. What is BC ID, passcode and r-code in the IMPS request packet?**

Ans: These are fixed values which are generated when the client is configured in UAT & Production environment and the same shall be provided after the configuration.

**20. What is AGGRID, AGGRNAME, CORPID & USERID in the NEFT & RTGS request packet?**

Ans: These are fixed values which are generated when the client is configured in UAT & Production environment and the same shall be provided after the configuration.

**21. What will be the actionable at client's end against the error received?**

Ans: Actionable has been mapped against each error codes in the error code document.

**22. What is the ideal response time for UPI transaction?**

Ans: For UPI, ideally the final status of the transaction will be available within 180 secs of initiating the transaction.

**23. What is the ideal response time for IMPS transaction?**

Ans: For IMPS, ideally the final status of the transaction will be available within 180 secs of initiating the transaction.

**24. What is the limit for number of UPI transactions?**

Ans: No transaction limit on volume per day. Can manage 2-3 transactions per second.

**25. What is limit for number of IMPS transactions?**

Ans: No transaction limit on volume per day. Can manage 2-3 transactions per second.

**26. What is limit for number of NEFT and RTGS?**

Ans: No transaction limit on volume per day. Can manage 2-3 transactions per second.

**27. Will NEFT 24X7 be implemented for corporate clients?**

Ans: Yes. But during 07:00PM to 01:00AM ICICI Banks will limit the number of transactions to '5' and consolidate value per user as 2 Lacs. This limit will also hold on bank holidays and alternate weekends.

**28. What will be the status of NEFT transaction, if it is initiated after the NEFT time slot is over for that day?**

Ans: The transaction will be kept in pending state. No amount shall be debited until the NEFT window reopens.

**29. What is 500 error code?**

Ans: The request fired in from a non-whitelisted IP.

**30. What is 403 forbidden error code?**

Ans: One of the parameters in the request packet isn't correct or the x-priority is incorrect.

**31. What is 8000 error code?**

Ans: The encryption mechanism is incorrect at client's end.

**32. How to check the encryption-decryption mechanism at clients' end?**

Ans: Client should encrypt the request using their own public key and decrypt using the corresponding private key to understand the mechanism.

**33. When will MIS be received?**

Ans: The MIS will be received on T+1 day.

**34. Where will the MIS be received?**

Ans: Client will have to download daily transaction MIS from the API developer portal.

**35. Will the MIS include failed transactions?**

Ans: Optional. Depends on the clients' requirements.

**36. What are the encryption steps for CIB registration & Beneficiary Registration APIs?**

Ans: Refer Section 5.2

**37. What are the encryption-decryption mechanism for Composite API and their respective status check APIs?**

Ans: Refer Section 5.1

**38. When can I use status check of an API and for how many days the status is available**

Ans. You can use the status check API for response not received and for pending scenario. Status check to be used upto 5 days from the original transaction day

### 39. What is the Chargeback request TAT for wrong fund transfer

Ans: within 60 days from the original request.

## 7. Appendix & Modification details~~Erratum~~:

### Changes incorporated from Composite API Document v1.11 onwards

#### 1) 1, INTRODUCTION

##### 2.4 Reverse Feed:

- Modified text: 'For UPI, the final status of the transaction will be available between 180 secs and T+2 days of initiating the transaction'.
- Modified text: 'For IMPS, the final status of the transaction will be available between 180 secs & 6 months\* of initiating the transaction.'

*(\*The exact timeline is subjective to purging activity at our end. Hence, it is recommended to check with ICICI team before taking any action on back dated transaction)*

##### 3.2 4096 Bits public key certificate:

- Added text: 'A self-signed certificate for UAT is acceptable, but for live environments a CA signed one is mandatory.'

#### 2) 2. API Details

##### 1.5 Input Parameters:

- Addition of 'Provided by ICICI' column for all payment modes

##### UPI:

- Removal of MPIN, Ref-ID, Expire-after & Notes from UPI request packet section as these fields are not mandatory
- Seq-no: This should start with 'ICI' and should not have any special characters.

##### NEFT:

- Removal of following optional parameters:  
senderName, senderAddress, senderContactDtls, senderIFSC, senderBankName,  
senderBankAddress, senderBankContactDtls, beneAddress, beneContactDtls,  
beneBankName, beneBankAddress, beneBankContactDtls

##### 1.6 Sample Request Packets:

- Sample request packets have been changed for UPI.
- For NEFT & RTGS, details required for UAT testing have been updated.

### 1.7 Output Parameters

- NEFT & RTGS out parameter table have been combined
- Sample response packet for NEFT & RTGS have been updated.
- In case of success response packet, UTR field has been added
- In case of failure response packet, ERRORCODE field has been added.
- Sample response code packet have also been added along with a disclaimer 'Please note, the sample response packet mentioned above is standard. But, in some failure cases, parameters RESPONSECODE, STATUS & ERRORCODE might not appear. RESPONSE and MESSAGE parameters will always appear.'

## 3) 5. API Name: Beneficiary Registration

### 5.1 Description:

- Added text: 'For clients with existing ALIAS ID, the default configuration for this API will not work. Client needs to delete their existing ALIAS ID through CIB portal and create a new ALIAS ID similar to the existing corporate user ID. For example, a client has ALIAS ID as 'ICICIBank' and corporate user ID as 'Bank12345', then client needs to delete the existing ALIAS ID i.e. 'ICICIBank' through CIB portal and create a new ALIAS ID as 'Bank12345' which has to be similar to the corporate user ID.'

## Changes incorporated from Composite API Document v1.12 onwards

- VPA Tag added to the UPI request parameter for Nodal account flow
- New CIB APIs under one hybrid encryption, CIB Registration, Bene addition, Bene Validation and CIB Neft Status
- IMPS Status API added to check the terminal status of final transactions

## Changes incorporated from Composite API Document v1.14 onwards

- IMPS and UPI, status and recon360 APIs are integrated into composite status
- Standalone status and imps recon API is removed
- Change done to IMPS status API with response starting with {"ImpsResponse":

## Changes incorporated from Composite API Document v1.16 onwards

- Changes done to composite status IMPS&UPI for recon360 API
- Changes done to for what all error codes IMPS 360 API can be invoked. I.e. for 11&30 error codes.
- Below changes in draft over status API of NEFT

### **Original Text:**

- ICICI to ICICI fund transfer (IFT fund transfer) - Composite status API shows the final response
- RGS transaction- Composite status API to be used for checking debit at account level and retrieving UTR number in case of pending transaction.
- RGS Incremental status- Incremental API to be used for final NEFT response

### **Modified Text:**

NEFT transaction processing essentially takes place in two steps - a) There is a debit in to remitter account first, b) Followed by transaction getting submitted to RBI for processing with the time lag.

For NEFT status enquiry there are two sets of APIs available from ICICI bank given below as -

- 3) NEFT debit confirmation API - Provides the status of the debit in remitter a/c
- 4) NEFT credit confirmation API - Provides the status of the credit in beneficiary a/c

#### Status Enquiry -

##### c. For money transferred to ICICI bank account bene -

For the amount transferred within ICICI bank the remitter needs to only use the first API ie. Debit confirmation API.

The remitter needs to fire this API and if the response received in status field is “SUCCESS” in that case the transaction can be marked as successfully processed.

For the status received as failure the same can be marked as failed and that particular transaction can be re-initiated. For any other status remitter should recheck the status after 2 hours. This re-check is required to be done only once. In case the status does not change to a definite success or failure, in that case no further action is possible and this transaction is to be marked as ‘pending’.

All pending transactions are to be reconciled with the remitter bank statement to ascertain the final status of the transaction. In case of no respective debit is found the transaction can be marked a failed & fresh transaction can be re-initiated. In case the debit is present in the account the transaction can be marked as successful.

Additionally, this API can also be utilized to fetch the UTR number for the transaction in case the same is not received with initial

##### d. For money transferred to non ICICI bank account bene -

It is a two-step process to identify the final status for these transaction.

- 4. The remitter needs to fire the first API (Debit) and if the response received in status field is “SUCCESS” only and only in that case the remitter should proceed to step 2 listed below.

For the status received as failure the same can be marked as failed and that particular transaction can be re-initiated. **For any other status remitter should recheck the status after 2 hours.** This re-check is required to be **done only once**. In case the status does not change to a definite success or failure, in that case no further action is possible and this transaction is to be marked as ‘pending’.

All pending transactions are to be reconciled with the remitter bank statement to ascertain the final status of the transaction. In case of no respective debit is found the transaction can be marked a failed & fresh transaction can be re-initiated. **In case the debit is present in the account then the remitter can proceed to step 2.**

5. Under step 2, the remitter should invoke the second API (Credit API) with the requisite request packet as documented in API document. Once the status is successfully received as “amount credited to beneficiary” transaction can be marked as successful.
6. It is possible that in some scenarios the status is received as “posted to RBI”. This status is provided for the transaction which the beneficiary bank has not provided the ultimate status. In such a scenario RBI has provisioned for 72 hrs for the beneficiary bank to update the final status of the transaction. However, if the status does not change post 72 hrs the remitter can mark these transactions as successful.

- Neft incremental API request packet parameter changed from “UTR” to UTRNUMBER”.
- Expected status values updated in NEFT incremental API\_output parameters

**New Response added to gate way sample responses:**

**Response packet of Gateway Failure, common to all payment modes (failure)**

```
{
"success":false,
"errorCode":"997",
"description":"Note : Initiate Status check after Some time"
}
```