DasshPe Payout API(Version 2.0)

CONFINDENTIALITY NOTICE

This entire document is private and confidential. The document and the information contained therein must not be given or communicated to any other parties without the express written permission of DasshPe. This document is meant solely for assessing our competence in the IT services and cannot be utilized. The entire document is under an NDA restricting its use, copying, distribution, and recompilation.

API INTRODUCTION

Use DasshPe Payout API to access various modules like Authenticate, Account, Beneficiary, Transfers, UPI validation.

COMMON HOST URL

Requests sent to our API are **POST** over **HTTPs**.
Content Type: **application/json**

payId :provided by DasshPe

Use the following host URLs based on your specific environment.

Environment	Host URL
Test	**https://secure.dasshpe.com/payoutV2test/api**
Prod	**https://secure.dasshpe.com/payoutV2live/api**

Access Control

Your IP has to be whitelisted to communicate with DasshPe. Submit an IPv4 address to whitelisting for both environment **Test** and **Prod** as well. You can add up to **5** IP addresses to whitelisting for each environment.

Reach out to the DasshPe team at **alerts@dasshpe.com** from your registered email-id to approve an added IP for whitelisting.

API Guidelines & Testing

Learn more about DasshPe Payouts APIs.

Guidelines

Below are some of the points you must be aware of while calling Payouts APIs:

- All API responses are in JSON format.
- POST requests should include *ContentType: application/json*
- All API response have **status**, **subCode**, **message**, and **data**.
- Subcode is the status subcode of the response-All requests to DasshPe that are processed by the server return *HTTP 200*. Use the status flag to determine if the request was successfully processed.

We recommended you to scan error sub-code and not error messages.

Authenticate

Calling the Authenticate APIs allows you to get and verify bearer tokens returned by DasshPe . DasshPe require these token for all further communication.

- Do not store the token in an insecure manner. Regenerating a new token does not invalidate the already generated token.
- Token generated is valid for 300 seconds. Please ensure that you recall the authorize API once the token has expired.
- Ensure your IPv4 is whitelisted.

1. Authorize and generate access token for every request

URL: /authorize

To authenticate with the DasshPe system and obtain the authorization token, call the authorize API. All other API calls must have this token as **authorization** JSON key for them to get processed.

Required Input

Key	Data type	Description
payId	string	16 characters alphanumeric value (provided by DasshPe)

Request:

```
{
    "payId": "{{}}"
}
```

Response:

SUCCESS

```
{
    "status": "SUCCESS",
    "subCode": "200",
    "message": "Token generated",
    "data": {
        "token":     "{{}}",
        "expiry": 1600342124
    }
}
```

ERROR

```
{
    "status": "ERROR",
    "subCode": "400",
    "message": "payId does not exists"
}
```

2. Verify Token

URL: /verifyToken

Verify the token generated. If the token does not exist, is invalid, or has expired, the response "Token is not valid" is returned. Regenerate token in case of token expiry for making API calls (use /authorize for this).

Required Inputs

Key	Data type	Description
payId	string	16 characters alphanumeric value (provided by DasshPe)
authorization	string	token to be verified

Request:

```
{
    "authorization": "{{}}",
    "payId": "{{}}"
}
```

Response:

SUCCESS
```
{
    "status": "SUCCESS",
    "subCode": "200",
    "message": "Token is valid"
}
```
ERROR
```
{
    "status": "ERROR",
    "subCode": "400",
    "message": "payId does not exists"
}

{
    "status": "ERROR",
    "subCode": "403",
    "message": "Token is not valid"
}
```

Account

1. Get Balance

URL: /account/getBalance

Get available balance of your account. Available balance is balance minus the sum of all pending transfers (transfers triggered and processing or pending now).

Required Inputs

Key	Data type	Description
payId	string	16 characters alphanumeric value (provided by DasshPe)
authorization	string	auth token

Request:
```
{
    "authorization": "{{}}",
    "payId": "{{}}"
}
```
Response:

SUCCESS
```
{
    "status": "SUCCESS",
    "subCode": "200",
    "message": "Balance for the account",
    "data": {
        "availableBalance": "73980.50"
    }
}
```

ERROR

```
{
	"status": "ERROR",
	"subCode": "400",
	"message": "payId does not exists"
}
```

Beneficiary

1. Add Beneficiary

URL: /beneficiary/addBeneficiary

Add a beneficiary to your DasshPe account by providing the bank account number, IFSC, and other required details. You can only request a transfer if the account has been successfully added as a beneficiary already. Please wait 30 minutes after adding the beneficiary

Required Inputs

Key	Data type	Description
payId	string	16 characters alphanumeric value (provided by DasshPe)
authorization	string	auth token
beneId	string	Unique Beneficiary Id to identify the beneficiary. Alphanumeric and underscore (_) allowed (50 character limit)
name	string	Beneficiary name, only alphabets and white space (100 character limit)
email	string	Beneficiaries email, string in email Id format (Ex: johndoe_1@example.com) - should contain @ and dot (.) - (200 character limit)
phone	string	Beneficiaries phone number, phone number registered in India (only digits, 8 - 12 characters after stripping +91)
address1	string	Beneficiaries address, alphanumeric and space allowed (but script, HTML tags gets sanitized or removed) (150 character limit)

Optional Inputs

Key	Data type	Description
bankAccount	string	Beneficiaries bank account number, alphanumeric (>=6 and <= 40 characters)
ifsc	string	Accounts IFSC (standard IFSC format) - length 11, first four bank IFSC and 5th digit 0
vpa	string	Beneficiary VPA, alphanumeric, dot (.), hyphen (-), at sign (@), and underscore (_) allowed (100 character limit). Note: underscore (_) and dot (.) gets accepted before and after at sign (@), but hyphen (-) get only accepted before at sign (@)
cardNo	string	Beneficiaries card number, only digits. Starting with 4 or 5 only (16 character limit)
address2	string	Beneficiary address, alphanumeric and space allowed (but script, HTML tags gets sanitized or removed) (150 character limit)
city	string	Beneficiary city, only alphabets and white space (50 character limit)
state	string	Beneficiary state, only alphabets and white space (50 character limit)
pincode	string	Beneficiaries pincode, only numbers (6 character limit)

Request:

```
{
    "authorization" : "{{}} ",
    "payId" : "{{}}",
    "beneId" :"{{}}",
    "name" : "{{}}",
    "email" : "{{}}",
    "phone" : "{{}}",
    "bankAccount" : "{{}}",
    "ifsc" : "{{}}",
    "address1" : "{{}}",
    "city" : "{{}}",
    "state" : "{{}}",
    "pincode" : "{{}}"
}
```

Response:

SUCCESS

```
{
     "status": "SUCCESS",
     "subCode": "200",
     "message": "Beneficiary added successfully"
}
```

ERROR

```
{
    "status": "ERROR",
    "subCode": "400",
    "message": "payId does not exists"
}

{
    "status": "ERROR",
    "subCode": "409",
    "message": "Beneficiary Id already exists"
}
```

2. Get Beneficiary Details

URL: /beneficiary/getBeneficiary

Get the details of a particular beneficiary.

Required Inputs

Key	Data type	Description
payId	string	16 characters alphanumeric value (provided by DasshPe)
authorization	string	auth token
beneId	string	Beneficiary Id

Request:

```
{
    "authorization":"{{}}",
    "payId" : "{{}}",
    "beneId" : "{{}}"
}
```

Response:

SUCCESS

```
{
    "status": "SUCCESS",
    "subCode": "200",
    "message": "Details of beneficiary",
    "data": {
        "beneId": "JOHN18011",
        "name": "John",
        ..................
        ................
    }
}
```

ERROR

```
{
    "status": "ERROR",
    "subCode": "404",
    "message": "Beneficiary does not exist"
}
```

```
    {
        "status": "ERROR",
        "subCode": "400",
        "message": "payId does not exists"
    }
```

3. Get Beneficiary ID

URL: /beneficiary/getBeneId

Get the beneficiary id by providing the bank account number and ifsc.

Required Inputs

Key	Data type	Description
payId	string	16 characters alphanumeric value (provided by DasshPe)
authorization	string	auth token
bankAccount	string	Beneficiaries bank account number, alphanumeric (>=6 and <= 40 characters)
ifsc	string	Beneficiaries bank's IFSC code (standard IFSC format) - length 11, first four IFSC and 5th 0

Request:

```
{
    "authorization":"{{}}",
    "payId" : "{{}}",
    "bankAccount" : "{{}}",
    "ifsc" : "{{}}"
}
```

Response:

SUCCESS

```
{
    "status": "SUCCESS",
    "subCode": "200",
    "message": "beneId retrieved successfully",
    "data": {
        "beneId": "JOHN18011"
    }
}
```

ERROR

```
{
        "status": "ERROR",
        "subCode": "404",
        "message": "Beneficiary not found with given bank account
                    details"
    }

    {
        "status": "ERROR",
        "subCode": "400",
        "message": "payId does not exists"
    }
```

4. Remove Beneficiary

URL: /beneficiary/removeBeneficiary

Remove an existing beneficiary from a list of added beneficiaries.

Required Inputs

Key	Data type	Description
payId	string	16 characters alphanumeric value (provided by DasshPe)
authorization	string	auth token
beneId	string	Beneficiaries Id to delete, alphanumeric and underscore allowed (50 character limit)

Request:

```
{
    "authorization":"{{}}",
    "payId" : "{{}}",
    "beneId" : "{{}}"
}
```

Response:

SUCCESS

```
{
    "status": "SUCCESS",
    "subCode": "200",
    "message": "Beneficiary removed"
}
```

ERROR

```
{
    "status": "ERROR",
    "subCode": "400",
    "message": "payId does not exists"
}
```

Transfers

1. Async Transfer

URL: /transfer/requestAsyncTransfer

Request an amount transfer at DasshPe by providing beneficiary id, amount, and transfer id. This is an async transfer request.

Once you trigger the requestAsyncTransfer API, DasshPe verifies your request and returns the DasshPe referenceId. The transfer to beneficiary account will be attempted within the next 60 seconds and you may query the transfer status after 60 seconds.

Required Inputs

Key	Data type	Description
payId	string	16 characters alphanumeric value (provided by DasshPe)
authorization	string	auth token
beneId	string	Beneficiary Id, alphanumeric allowed
amount	string	Amount to be transferred, decimal (>= 1.00)
transferId	string	A unique id to identify this transfer, alphanumeric and underscore (_) allowed (40 character limit)

Optional Inputs

Key	Data type	Description
transferMode	string	Mode of transfer, **banktransfer** by default. Allowed values are **upi**, **paytm**, **amazonpay**, and **card**
remarks	string	Additional remarks, if any. alphanumeric and white spaces allowed (70 characters limit)

Request:

```
{
    "authorization" : "{{}} ",
    "payId" : "{{}}",
    "beneId" :"{{}}",
    "amount" : "{{}}",
    "transferId" : "{{}}",
    "transferMode" : "{{}}",
    "remarks" : "{{}}"
}
```

Response:

SUCCESS

```
{
    "status": "SUCCESS",
    "subCode": "201",
    "message": "Transfer Initiated",
    "data": {
        "referenceId": "10***0"
    }
}
```

ERROR

```
{
    "status": "ERROR",
    "subCode": "400",
    "message": "payId does not exists"
}
```

2. Get Transfer Status

URL: /transfer/getTransferStatus

Get details of a particular transfer. You can pass referenceId and transferId to fetch the details.

Required Inputs

Key	Data type	Description
payId	string	16 characters alphanumeric value (provided by DasshPe)
authorization	string	auth token
referenceId	string	referenceId of the transaction
transferId	string	transferId of the transaction

Request:

```
{
    "authorization":"{{}}",
    "payId" : "{{}}",
    "referenceId" : "{{}}",
    "transferId" : "{{}}"
}
```

Response:

SUCCESS

```
{
    "status": "SUCCESS",
    "subCode": "200",
    "message": "Details of transfer {{}}",
        "data": {"transfer":
            {
                "referenceId": 1***3,
                "beneId": "ABCD_123",
                "amount": "20.00",
                "status": "SUCCESS",
                ..................
                ..................
            }
    }
}
```

ERROR

```
{
    "status": "ERROR",
    "subCode": "404",
    "message": "{{}} is invalid or doesnot exist"
}
{
    "status": "ERROR",
    "subCode": "400",
    "message": "payId does not exists"
}
```

UPI Validation

1. Validation for UPI

URL: /validation/upiDetails

Verify whether a UPI handle exists. You receive a customer name at the bank in the response for valid UPIs.

Required Inputs

Key	Data type	Description
payId	string	16 characters alphanumeric value (provided by DasshPe)
authorization	string	auth token
name	string	name of the account to be validated, only alphabets and white space (100 character limit)
vpa	string	VPA of account to be validated, alphanumeric, period (.), hyphen (-), at sign (@) and underscore (_) allowed (100 character limit) **Note**: Underscore (_) and dot (.) gets accepted before and after at sign (@), but hyphen (-) get accepted only before the at sign (@)

Request:

```
{
    "authorization":"{{}}",
    "payId" : "{{}}",
    "name" : "JANE DOE",
    "vpa" : "success@upi"
}
```

Response:

SUCCESS

```
{
    "status": 1,
    "subCode": "200",
    "message": "VPA verification successful",
        "data": {
            {
                "nameAtBank": "JANE DOE",
                "accountExists": "Yes"
            }
    }
}
```

```
{
    "status": 1,
    "subCode": "200",
    "message": "No Account linked with VPA",
        "data": {
            {
                "accountExists": "No"
            }
    }
}
```

ERROR

```
{
    "status": 0,
    "subCode": "422",
    "message": "Either VPA or name invalid"
}

{
    "status": "ERROR",
    "subCode": "400",
    "message": "payId does not exists"
}
```

Payout Methods

Learn about various payment methods supported by DasshPe.

DasshPe supports payouts to your party using various payout methods. You can payout depending on the preferred payment method for the party.
Currently, the following payment methods are supported:

- Bank transfers
- Card transfers
- UPI
- Wallets

Bank Transfers

Bank transfers via DasshPe support both the IMPS and NEFT payouts. By default, IMPS is the configured mode for payouts, since deposits made using IMPS are instantly credited to the beneficiary account.
Below is the comparison between IMPS and NEFT. You can select the required payment method as per your need.

IMPS	**NEFT**
IMPS is an instant transfer the amount is credited to the beneficiary account immediately.	NEFT ideally takes up to 2 hours to reflect in the beneficiary account.
IMPS is available at all times.	NEFT is available between 1 AM and 6:45 PM on all NEFT working days (Monday to Saturday, except 2nd and 4th Saturday).
IMPS has a limit of Rs. 2 lakhs per payout.	NEFT is the default payout method for any amount higher than Rs. 2 lakhs.

You need the beneficiary ID, name, email, phone, bank account number, and IFSC to add a beneficiary for bank payouts. If the bank account details or IFSC is incorrect, the transfer fails.

Card Payouts

Card payouts can be made at all times, including weekends and holidays. With instant payouts, you can immediately send funds to credit card.

Credit Cards

Payouts support all the credit cards. Payouts are instant for the banks that support IMPS on credit cards and for others it may take up to 48 hours to credit the transferred amount.

You need the beneficiary ID, name, email, phone, and card number to add a beneficiary for card payouts.

UPI Payouts

UPI payouts can be made at all times including weekends and holidays. With instant payouts, you can immediately send funds to all valid UPI VPAs.

Note:

- The transaction limit per UPI payouts is Rs. 1 lakh. Although the transaction limit is Rs. 1 Lakh, the upper limit might vary from bank-to-bank. This limit ranges from Rs. 10,000 to Rs. 1 lakh.
- You need to provide a valid UPI VPA for the payouts to go through. No other beneficiary details are required. If the provided VPA is incorrect, then the payout fails.

You need the beneficiary ID, name, email, phone, and VPA to add a beneficiary for UPI payouts. If the VPA is incorrect, the transfer fails.

Wallet Payouts

Wallet payouts can be made at all times, including weekends and holidays. With instant payouts, you can immediately send funds to all available wallet accounts. Currently, wallet payouts support Paytm and Amazon Pay.

Paytm

- Beneficiary should have completed the KYC validation for the Paytm payout to go through, or it fails.
- Payouts are done instantly and reflect in the statement for the beneficiary.
- The maximum limit is up to Rs. 1 Lakh if the customer has completed the KYC registration.
- You need the beneficiary ID, name, email, phone number for making the paytm payouts.

Amazon Pay

- Unlike Paytm, payouts via Amazon Pay does not require the beneficiary to have the KYC validation.
- Payouts are done instantly and reflect in the statement for the beneficiary.
- The maximum limit per transfer is Rs. 10000.
- You need the beneficiary ID, name, email, phone number for making the Amazon payouts.

Testing

Learn about various ways to test your integration before going live.

For all transactions, use the following test bank, card, UPI, and wallet numbers to trigger all validations and transfers for payouts. Add these details while adding the beneficiary, and mock the transfer responses to the provided results.
Please note that transfers to any other details other than the ones mentioned below fail. Test mode payouts and validations simulate a live payout but don't get processed with the bank.

Bank Numbers
For banks, the primary parameters for transfer would be the bank account and IFSC number included while adding the beneficiary. Use these test bank numbers to test payouts to a card. Utilized only with test API keys.

Account Number	IFSC	Remarks
026291800001191	YESB0000262	Success
00011020001772	HDFC0000001	Success
000890289871772	SCBL0036078	Success
000100289877623	SBIN0008752	Success
2640101002729	CNRR0002640	Failure – Invaid IFSC code
026291800001190	YESB0000262	Failure – Invalid Account number
007711000031	HDFC0000077	Pending
00224412311300	YESB0000001	Pending (later to Success)
7766666351000	YESB0000001	Pending (later to Failure)
02014457596969	CITI0000001	Success (later to Reversed)
34978321547298	KKBK0000001	Timeout - 100s (later to Success) Test with a timeout of 30s and 100s (gateway timeout)

Card Numbers
For cards, the primary parameter for transfer would be the card number included while adding the beneficiary. Use these test card numbers to test payouts to a card. Utilized only with test API keys.

Card Number	Remarks
4434260000000000	Successful card transfer
4434260000000001	Failed card transfer

Wallet Numbers
For wallets, the primary parameter for transfer would be the phone number included while adding the beneficiary. Use these wallet numbers to test payouts to a wallet. Utilized only with test API keys.

Phone Number	**Remarks**
9999999999	Paytm successful wallet transfer
8888888888	Paytm successful wallet transfer
7777777777	AmazonPay successful wallet transfer
6666666666	AmazonPay successful wallet transfer

UPI Numbers
For UPI, the primary parameter for transfer would be the UPI VPA included while adding the beneficiary. Use these UPI VPA to test payouts to an account. Utilized only with test API keys.

VPA	**Remarks**
success@upi	Successful UPI transfer
failure@upi	Failed UPI transfer

While in TEST mode, as long as valid external bank information and other relevant conditions get covered, it never requires real identity verification or other interactive steps that are part of the custom account workflow.

The security is on the Token approach, and everything is on https. Also, as another security feature, we accept requests only from whitelisted servers.

API Error Response

1) /authorize

Error Code	Message
400	payId can’t be null
400	invalid payId
400	payId does not exists
400	ip not whitelisted
401	Unauthorized
500	Internal Server Error

2) /verifyToken

Error Code	Message
400	payId can’t be null
400	invalid payId
400	payId does not exists
400	ip not whitelisted
400	authorization can’t be null
401	Unauthorized
403	Token is not valid
500	Internal Server Error

3) /account/getBalance

Error Code	Message
400	payId can’t be null
400	invalid payId
400	payId does not exists
400	ip not whitelisted
400	authorization can’t be null
401	Unauthorized
500	Internal Server Error

4) /beneficiary/addBeneficiary

Error Code	Message
400	payId can’t be null
400	invalid payId
400	payId does not exists
400	ip not whitelisted
400	authorization can’t be null
400	beneId can’t be null
400	beneId not more than 50 characters

400	name can’t be null
400	name not more than 100 characters
400	email can’t be null
400	invalid email
400	email not more than 200 characters
400	phone can’t be null
400	invalid phone number
400	address1 can’t be null
400	address1 not more than 150 characters
400	cardNo not more than 16 characters
400	invalid pincode
401	Unauthorized
400	Not added
400/409	Beneficiary Id already exists
409	Entered bank Account is already registered
412	Post data is empty or not a valid JSON
412	Please provide a valid Bank IFSC code.
422	Please provide a valid Beneficiary Id
422	Invalid details provided
422	Please provide a valid MasterCard or Visa card number
422	Please provide a masked card number of a valid MasterCard or Visa card
500	Internal Server Error
520	Adding beneficiary Failed

5) /beneficiary/getBeneficiary

Error Code	Message
400	payId can’t be null
400	invalid payId
400	payId does not exists
400	ip not whitelisted
400	authorization can’t be null
400	beneId can’t be null
400	beneId not more than 50 characters
400	beneId does not exist
404	Beneficiary does not exist
500	Internal Server Error
520	Unknown error occurred

6) /beneficiary/getBeneId

Error Code	Message
400	payId can’t be null
400	invalid payId
400	payId does not exists
400	ip not whitelisted
400	authorization can’t be null
400	bankAccount can’t be null
400	bankAccount not less than 6 characters
400	bankAccount not more than 40 characters
400	Ifsc can’t be null
400	Beneficiary does not exist
422	Please provide a valid bank account and ifsc
500	Internal Server Error
520	Error while fetching beneId

7) /beneficiary/removeBeneficiary

Error Code	Message
400	payId can’t be null
400	invalid payId
400	payId does not exists
400	ip not whitelisted
400	authorization can’t be null
400	beneId can’t be null
400	beneId not more than 50 characters
400	Beneficiary does not exist
400	Something went wrong, please try again.
500	Internal Server Error
520	Unknown error occurred

8) /transfer/requestAsyncTransfer

Error Code	Message
400	payId can’t be null
400	invalid payId
400	payId does not exists
400	ip not whitelisted
400	authorization can’t be null
400	beneId can’t be null
400	beneId not more than 50 characters
400	invalid transferMode

400	invalid amount
400	transferId can't be null
400	transferId not more than 40 characters
400	transferId already exist
400	insufficient balance
400	Seems something worng. Please content to your relationship manager if does not get the refund in 1 bussines day.
400	Something went wrong, please try again.
412	Please wait 30 minutes after adding the beneficiary
500	Internal Server Error
520	Unknown error occurred

9) /transfer/getTransferStatus

Error Code	Message
400	payId can't be null
400	invalid payId
400	payId does not exists
400	ip not whitelisted
400	authorization can't be null
400	referenceId & transferId can't be null
400	Something went wrong, please try again.
500	Internal Server Error
520	Unknown error occurred

10) /validation/upiDetails

Error Code	Message
400	payId can't be null
400	invalid payId
400	payId does not exists
400	ip not whitelisted
400	authorization can't be null
400	vpa can't be null
400	vpa not more than 100 characters
400	name can't be null
400	name not more than 100 characters
400	Something went wrong, please try again.
422	Please provide a valid name
422	Please provide a valid UPI VPA
422	Invalid UPI VPA provided
422	Either VPA or name invalid

422	Please provide a valid Virtual Payee Address.
500	Internal Server Error
520	Unknown error occurred