Select these steps to send the necessary configuration to Okta.



Using Okta Workspace to Verify

CustomizareNew- French



CONTENTS

1. **titlejhbnj** **3**
2. **Test** **161**
3. **Untitled** **162**
4. **c'est un titre h1** **163**
5. **Baguette parce que car cela ouais** **165**
    - 5.1. Test doc 166
6. **Indenting** **172**
    - 6.1. Une autre page 172
        - 6.1.1. Filtrage avancé 173
    - 6.2. Nouvelle page 181
7. **page de test** **182**
8. **Reminder** **183**
    - 8.1. Langage Toxique 183
    - 8.2. ASDASDSDA 192
    - 8.3. Test de numérotation sur les images problème (cloné) 193
        - 8.3.1. asdsadas 212
        - 8.3.2. Langage Toxique 216
    - 8.4. asdsadasdas (cloné) 224



1. titlejhbnj

6.23 OAS 3.0

Judopay Transaction API

Servers

https://api-sandbox.judopay.com - Sandbox environment

Authorize

Registering Cards

POST /transactions/savecard

If you do not want to perform a pre-authorisation check on a customer's account, use this call to tokenize the card information into an encrypted string.

Note: The Card Token is not tested until it is used.

Parameters | Try it out

Name	Description
Api-Version * required string *(header)*	The version of this API. 6.23

Request body required | application/json

Request is sent with full card details

Examples: SaveCardWithCardHolderName

Example Value | Schema

```
{
  "cardNumber": "4111111111111111",
  "expiryDate": "01/28",
  "yourConsumerReference": "2b45fd3f-cee5-4e7e-874f-28051db65408",
  "judoId": "100100100",
  "cardHolderName": "John Doe"
}
```

Responses



Code	Description	Links
200	Successful response	*No links*

Media type: application/json

Controls `Accept` header.

Examples: SaveCardWithCardHolderNameResponse

Example Value | Schema


```
{
  "receiptId": "1001131610340495360",
  "yourPaymentReference": "6482c678-cad3-4efd-b081-aeae7a89a134",
  "type": "Save",
  "createdAt": "2025-08-27T10:47:46.5157+01:00",
  "result": "Success",
  "message": "Register Card",
  "judoId": "100100100",
  "merchantName": "Test Merchant",
  "appearsOnStatementAs": "APL*/TestMerchant       ",
  "originalAmount": "0.00",
  "netAmount": "0.00",
  "amount": "0.00",
  "currency": "GBP",
  "acquirerTransactionId": "33666277256892154000",
  "externalBankResponseCode": "",
  "authCode": "",
  "riskScore": 0,
```


400	Bad request if any of the required request parameters are not supplied	*No links*
401	Unauthorized	*No links*

Media type: application/json

Examples: Unauthorized

Example Value | Schema


```
{
  "requestId": "68ea7fab5879f147d51103c3eb265780",
  "message": "Sorry, we were unable to authorize this request. Please check your details and permissions before trying again.",
  "code": 7,
  "category": 1
}
```


403	Forbidden if token used does not have required permission	*No links*

Media type: application/json

Examples: Forbidden

Example Value | Schema


```
{
  "requestId": "68ea7fab5879f147d51103c3eb265780",
  "message": "Sorry, it looks like permissions are not enabled to perform this request. Application permissions can be configured in the judo Dashboard.",
  "code": 84,
  "category": 1
}
```



Pre-Authorization

POST `/transactions/checkcard`

Conduct a zero amount pre-authorisation in the customer's account. Use this endpoint to check if the card is valid to perform a 0 Auth on a customer's account. 0 Auth means it does not hold any funds on the customer's account.

Note: The card information is tokenized into an encrypted string and tested as part of the pre-authorisation process. Not all gateways support this feature.

Warning: The Issuing Bank will not check funds.

Parameters | Try it out

Name	Description
Api-Version * required `string` *(header)*	The version of this API. 6.23

Request body required — application/json

Request is sent with full card details

Examples: ThreeDSecure2CheckCard

Example Value | Schema


```
{
  "cardNumber": "4111111111111111",
  "cv2": "123",
  "expiryDate": "01/28",
  "cardAddress": {
    "address1": "CardHolder House",
    "address2": "1 CardHolder Street",
    "town": "CardHolder Town",
    "postCode": "AB1 2CD",
    "countryCode": "826"
  },
  "judoId": "100100100",
  "yourConsumerReference": "2b45fd3f-cee5-4e7e-874f-28051db65408",
  "yourPaymentReference": "6482c678-cad3-4efd-b081-aeae7a89a134",
  "cardHolderName": "John Doe",
  "mobileNumber": "7999999999",
  "phoneCountryCode": "44",
  "emailAddress": "test.user@judopay.com",
```


Responses

Code	Description	Links
200	Successful response will be one of • transaction receipt • request for 3DS2 device details	*No links*



Code	Description	Links

- request for 3DS2 challenge

Media type: application/json

Examples: PaymentReceipt

Controls `Accept` header.

Example Value | Schema


```
{
  "receiptId": "1001131610340495360",
  "yourPaymentReference": "6482c678-cad3-4efd-b081-aeae7a89a134",
  "type": "PreAuth",
  "createdAt": "2025-08-27T10:47:46.5157+01:00",
  "result": "Success",
  "message": "AuthCode: 123456",
  "judoId": "100100100",
  "merchantName": "Test Merchant",
  "appearsOnStatementAs": "APL*/TestMerchant       ",
  "originalAmount": "10.99",
  "netAmount": "10.99",
  "amount": "10.99",
  "currency": "GBP",
  "acquirerTransactionId": "33666277256892154000",
  "externalBankResponseCode": "A",
  "authCode": "123456",
  "riskScore": 0,
```


400	Bad request if any of the required request parameters are not supplied	*No links*
401	Unauthorized	*No links*

Media type: application/json

Examples: Unauthorized

Example Value | Schema


```
{
  "requestId": "68ea7fab5879f147d51103c3eb265780",
  "message": "Sorry, we were unable to authorize this request. Please check your details and permis
sions before trying again.",
  "code": 7,
  "category": 1
}
```


403	Forbidden if token used does not have required permission	*No links*

Media type: application/json

Examples: Forbidden

Example Value | Schema


```
{
  "requestId": "68ea7fab5879f147d51103c3eb265780",
  "message": "Sorry, it looks like permissions are not enabled to perform this request. Application
permissions can be configured in the judo Dashboard.",
  "code": 84,
  "category": 1
}
```


POST ~~/transactions/registercard~~



Warning: Deprecated

Conduct a pre-authorisation, reserving a pre-configured amount in the customer's account. The amount is set during onboarding (default is 1.01). Registercard will tokenise the card information into an encrypted string. The POST /transactions/voids endpoint can be used to cancel this pre-authorisation so it does not appear in the customer's statement

Note: The card information is tokenised into an encrypted string and tested as part of the pre-authorisation process.

This call has been deprecated in favour of POST /transactions/checkcard or POST /transactions/preauths

Parameters Try it out

Name	Description
Api-Version * required `string` *(header)*	The version of this API. 6.23

Request body required application/json

Request is sent with full card details

Examples:

SimpleNon3dsRegisterCard

Example Value | Schema

```
{
  "cardNumber": "4111111111111111",
  "cv2": "123",
  "expiryDate": "01/28",
  "judoId": "100100100",
  "yourConsumerReference": "2b45fd3f-cee5-4e7e-874f-28051db65408",
  "yourPaymentReference": "6482c678-cad3-4efd-b081-aeae7a89a134"
}
```

Responses

Code	Description	Links
200	Successful response will be one of • transaction receipt • request for 3DS2 device details • request for 3DS2 challenge Media type: application/json Controls `Accept` header. Examples: PaymentReceipt **Example Value** | Schema	*No links*

```
{
  "receiptId": "1001131610340495360",
  "yourPaymentReference": "6482c678-cad3-4efd-b081-aeae7a89a134",
  "type": "PreAuth",
```



Code	Description	Links


```
  "createdAt": "2025-08-27T10:47:46.5157+01:00",
  "result": "Success",
  "message": "AuthCode: 123456",
  "judoId": "100100100",
  "merchantName": "Test Merchant",
  "appearsOnStatementAs": "APL*/TestMerchant       ",
  "originalAmount": "10.99",
  "netAmount": "10.99",
  "amount": "10.99",
  "currency": "GBP",
  "acquirerTransactionId": "33666277256892154000",
  "externalBankResponseCode": "A",
  "authCode": "123456",
```


400 Bad request if any of the required request parameters are not supplied — *No links*

401 Unauthorized — *No links*

Media type: application/json

Examples: Unauthorized

Example Value | Schema


```
{
  "requestId": "68ea7fab5879f147d51103c3eb265780",
  "message": "Sorry, we were unable to authorize this request. Please check your details and permissions before trying again.",
  "code": 7,
  "category": 1
}
```


403 Forbidden if token used does not have required permission — *No links*

Media type: application/json

Examples: Forbidden

Example Value | Schema


```
{
  "requestId": "68ea7fab5879f147d51103c3eb265780",
  "message": "Sorry, it looks like permissions are not enabled to perform this request. Application permissions can be configured in the judo Dashboard.",
  "code": 84,
  "category": 1
}
```


POST /transactions/preauths

Use preauths to reserve funds on a customer's account and postpone completing the payment until the goods have been delivered or the service fulfilled. This is also known as reserve and collection of funds. **Caution** The expiry time of the reserved pre-authorised funds differs across issuers.

Note To reduce your PCI scope, you can store a card token instead of the card number. The card token is a randomly generated string linked to a card saved securely within the Judopay Card Vault. You can also use a card token to pre-authorise repeat payments without the consumer having to re-enter their details.

Parameters — Try it out



Name	Description
Api-Version * required `string` *(header)*	The version of this API. 6.23

Request body required — application/json

Request can be sent with

- full card details,
- a cardToken generated with a call to POST /transactions/savecard, or
- wallet details from Apple Pay or Google Pay

Examples: SimpleNon3dsPreAuth

Example Value | Schema

```
{
  "cardNumber": "4111111111111111",
  "cv2": "123",
  "expiryDate": "01/28",
  "judoId": "100100100",
  "yourConsumerReference": "2b45fd3f-cee5-4e7e-874f-28051db65408",
  "yourPaymentReference": "6482c678-cad3-4efd-b081-aeae7a89a134",
  "currency": "GBP",
  "amount": 1.01
}
```

Responses

Code	Description	Links
200	Successful response will be one of • transaction receipt • request for 3DS2 device details • request for 3DS2 challenge Media type: application/json Controls `Accept` header. Examples: PaymentReceipt Example Value | Schema	*No links*

```
{
  "receiptId": "1001131610340495360",
  "yourPaymentReference": "6482c678-cad3-4efd-b081-aeae7a89a134",
  "type": "PreAuth",
  "createdAt": "2025-08-27T10:47:46.5157+01:00",
  "result": "Success",
  "message": "AuthCode: 123456",
  "judoId": "100100100",
  "merchantName": "Test Merchant",
  "appearsOnStatementAs": "APL*/TestMerchant       ",
  "originalAmount": "10.99",
  "netAmount": "10.99",
  "amount": "10.99",
```



Code	Description	Links


```
  "currency": "GBP",
  "acquirerTransactionId": "33666277256892154000",
  "externalBankResponseCode": "A",
  "authCode": "123456",
```


400 Bad request if any of the required request parameters are not supplied — *No links*

401 Unauthorized — *No links*

Media type: application/json | Examples: Unauthorized

Example Value | Schema


```
{
  "requestId": "68ea7fab5879f147d51103c3eb265780",
  "message": "Sorry, we were unable to authorize this request. Please check your details and permissions before trying again.",
  "code": 7,
  "category": 1
}
```


403 Forbidden if token used does not have required permission — *No links*

Media type: application/json | Examples: Forbidden

Example Value | Schema


```
{
  "requestId": "68ea7fab5879f147d51103c3eb265780",
  "message": "Sorry, it looks like permissions are not enabled to perform this request. Application permissions can be configured in the judo Dashboard.",
  "code": 84,
  "category": 1
}
```


409 Conflict - unable to find a route for the transactions — *No links*

Media type: application/json | Examples: Judold Blocked

Example Value | Schema


```
{
  "requestId": "68ea7fab5879f147d51103c3eb265780",
  "message": "Sorry, but we are currently unable to accept payments to this account. Please contact customer services.",
  "code": 54,
  "category": 3
}
```


POST /transactions/incrementalAuth

Increment the amount on a successful customer initiated pre-auth (before it is fully collected).

Parameters | Try it out



Name	Description
Api-Version * required `string` *(header)*	The version of this API. 6.23

Request body required — application/json

Example Value | Schema


```
{
  "receiptId": "1001131610340495360",
  "amount": 0.99,
  "yourPaymentMetaData": {
    "internalLocationRef": "Example",
    "internalId": 99
  },
  "yourPaymentReference": "aa648425-2230-4e4e-8a61-27f06df54542"
}
```


Responses

Code	Description	Links
200	Successful response Media type: application/json Controls `Accept` header. **Example Value** | Schema	*No links*
401	Unauthorized Media type: application/json Examples: Unauthorized	*No links*


```
{
  "receiptId": "1001131610340495360",
  "originalReceiptId": "1001124307998347264",
  "yourPaymentReference": "6482c678-cad3-4efd-b081-aeae7a89a134",
  "type": "PreAuth",
  "createdAt": "2025-02-05T16:28:32.8596+00:00",
  "result": "Success",
  "message": "AuthCode: 123456",
  "judoId": 100100100,
  "merchantName": "Test Merchant",
  "appearsOnStatementAs": "APL*/TestMerchant       ",
  "originalAmount": "10.99",
  "amountCollected": "1.99",
  "netAmount": "1.00",
  "amount": "1.00",
  "currency": "GBP",
  "recurringPaymentType": "MIT",
  "acquirerTransactionId": "33666277256892153705",
```



Code	Description	Links

Example Value | Schema

```
{
  "requestId": "68ea7fab5879f147d51103c3eb265780",
  "message": "Sorry, we were unable to authorize this request. Please check your details and permissions before trying again.",
  "code": 7,
  "category": 1
}
```

403 Forbidden if token used does not have required permission — *No links*

Media type: application/json

Examples: Forbidden

Example Value | Schema

```
{
  "requestId": "68ea7fab5879f147d51103c3eb265780",
  "message": "Sorry, it looks like permissions are not enabled to perform this request. Application permissions can be configured in the judo Dashboard.",
  "code": 84,
  "category": 1
}
```

POST /transactions/collections

Collect funds previously reserved using the pre-authorisation request. You can choose to let your customer know when to expect their card to be charged:

> You have not yet been charged for this item. We won't debit any funds from your payment card until your journey is complete.

Caution The pre-authorised funds will expire in three days if not collected, releasing the reserved funds.

Parameters — Try it out

Name	Description
Api-Version * required `string` *(header)*	The version of this API. 6.23

Request body required — application/json

Example Value | Schema

```
{
  "receiptId": "1001131610340495360",
  "amount": 10.99,
  "yourPaymentMetaData": {
    "internalLocationRef": "Example",
    "internalId": 99
  },
```




```
  "yourPaymentReference": "aa648425-2230-4e4e-8a61-27f06df54542"
}
```


Responses

Code	Description	Links
200	Successful response Media type application/json Controls `Accept` header. **Example Value** | Schema	*No links*


```
{
  "receiptId": "1001131610340495360",
  "originalReceiptId": "1001124307998347264",
  "yourPaymentReference": "6482c678-cad3-4efd-b081-aeae7a89a134",
  "type": "PreAuth",
  "createdAt": "2025-02-05T16:28:32.8596+00:00",
  "result": "Success",
  "message": "AuthCode: 123456",
  "judoId": 100100100,
  "merchantName": "Test Merchant",
  "appearsOnStatementAs": "APL*/TestMerchant       ",
  "originalAmount": "10.99",
  "amountCollected": "1.99",
  "netAmount": "1.00",
  "amount": "1.00",
  "currency": "GBP",
  "recurringPaymentType": "MIT",
  "acquirerTransactionId": "33666277256892153705",
```


401	Unauthorized Media type: application/json Examples: Unauthorized **Example Value** | Schema	*No links*


```
{
  "requestId": "68ea7fab5879f147d51103c3eb265780",
  "message": "Sorry, we were unable to authorize this request. Please check your details and permissions before trying again.",
  "code": 7,
  "category": 1
}
```


403	Forbidden if token used does not have required permission Media type: application/json Examples: Forbidden **Example Value** | Schema	*No links*


```
{
  "requestId": "68ea7fab5879f147d51103c3eb265780",
  "message": "Sorry, it looks like permissions are not enabled to perform this request. Application permissions can be configured in the judo Dashboard.",
```



Code	Description	Links

```
  "code": 84,
  "category": 1
}
```

POST `/transactions/voids`

Use void to cancel a pre-authorised transaction if the funds have not yet settled. **Note** Not all banks support voiding transactions. Check with your Acquirer. Voids cannot be performed on transactions that have been collected (partial or full).

Caution The pre-authorised funds will expire in three days if not collected, releasing the reserved funds.

Parameters Try it out

Name	Description
Api-Version * required `string` *(header)*	The version of this API. 6.23

Request body required — application/json

Example Value | Schema

```
{
  "receiptId": "1001131610340495360",
  "amount": 10.99,
  "yourPaymentReference": "aa648425-2230-4e4e-8a61-27f06df54542"
}
```

Responses

Code	Description	Links
200	Successful response Media type: application/json Controls `Accept` header. Example Value | Schema	*No links*

```
{
  "receiptId": "1001131610340495360",
  "originalReceiptId": "1001124307998347264",
  "yourPaymentReference": "6482c678-cad3-4efd-b081-aeae7a89a134",
  "type": "PreAuth",
  "createdAt": "2025-02-05T16:28:32.8596+00:00",
  "result": "Success",
  "message": "AuthCode: 123456",
  "judoId": 100100100,
```



Code	Description	Links


```
  "merchantName": "Test Merchant",
  "appearsOnStatementAs": "APL*/TestMerchant       ",
  "originalAmount": "10.99",
  "amountCollected": "1.99",
  "netAmount": "1.00",
  "amount": "1.00",
  "currency": "GBP",
  "recurringPaymentType": "MIT",
```


401 Unauthorized — *No links*

Media type: application/json | Examples: Unauthorized

Example Value | Schema


```
{
  "requestId": "68ea7fab5879f147d51103c3eb265780",
  "message": "Sorry, we were unable to authorize this request. Please check your details and permissions before trying again.",
  "code": 7,
  "category": 1
}
```


403 Forbidden if token used does not have required permission — *No links*

Media type: application/json | Examples: Forbidden

Example Value | Schema


```
{
  "requestId": "68ea7fab5879f147d51103c3eb265780",
  "message": "Sorry, it looks like permissions are not enabled to perform this request. Application permissions can be configured in the judo Dashboard.",
  "code": 84,
  "category": 1
}
```


Payments

POST `/transactions/payments`

Note To reduce your PCI scope, you can store a card token instead of the card number. The card token is a randomly generated string linked to a card saved securely within the Judopay Card Vault. You can also use a card token to pre-authorise repeat payments without the consumer having to re-enter their details.

Parameters — Try it out

Name	Description
Api-Version * required `string` *(header)*	The version of this API. 6.23

Request body required — application/json



Request can be sent with

- full card details,
- a cardToken generated with a call to POST /transactions/savecard, or
- wallet details from Apple Pay or Google Pay

Examples:

SimpleNon3dsPayment

Example Value | Schema


```
{
  "cardNumber": "4111111111111111",
  "cv2": "123",
  "expiryDate": "01/28",
  "judoId": "100100100",
  "yourConsumerReference": "2b45fd3f-cee5-4e7e-874f-28051db65408",
  "yourPaymentReference": "6482c678-cad3-4efd-b081-aeae7a89a134",
  "currency": "GBP",
  "amount": 1.01
}
```


Responses

Code	Description	Links
200	Successful response will be one of • transaction receipt • request for 3DS2 device details • request for 3DS2 challenge Media type: application/json Examples: PaymentReceipt Controls `Accept` header. **Example Value** | Schema	*No links*


```
{
  "receiptId": "10011316103404953 60",
  "yourPaymentReference": "6482c678-cad3-4efd-b081-aeae7a89a134",
  "type": "PreAuth",
  "createdAt": "2025-08-27T10:47:46.5157+01:00",
  "result": "Success",
  "message": "AuthCode: 123456",
  "judoId": "100100100",
  "merchantName": "Test Merchant",
  "appearsOnStatementAs": "APL*/TestMerchant       ",
  "originalAmount": "10.99",
  "netAmount": "10.99",
  "amount": "10.99",
  "currency": "GBP",
  "acquirerTransactionId": "33666277256892154000",
  "externalBankResponseCode": "A",
  "authCode": "123456",
  "riskScore": 0,
```


401	Unauthorized Media type: application/json Examples: Unauthorized	*No links*



Code	Description	Links

Example Value | Schema

```
{
  "requestId": "68ea7fab5879f147d51103c3eb265780",
  "message": "Sorry, we were unable to authorize this request. Please check your details and permissions before trying again.",
  "code": 7,
  "category": 1
}
```

403 Forbidden if token used does not have required permission — *No links*

Media type: application/json

Examples: Forbidden

Example Value | Schema

```
{
  "requestId": "68ea7fab5879f147d51103c3eb265780",
  "message": "Sorry, it looks like permissions are not enabled to perform this request. Application permissions can be configured in the judo Dashboard.",
  "code": 84,
  "category": 1
}
```

409 Conflict - unable to find a route for the transactions — *No links*

Media type: application/json

Examples: Judold Blocked

Example Value | Schema

```
{
  "requestId": "68ea7fab5879f147d51103c3eb265780",
  "message": "Sorry, but we are currently unable to accept payments to this account. Please contact customer services.",
  "code": 54,
  "category": 3
}
```

POST /transactions/refunds

Use refund to return funds to the customer. You can process a full or partial refund:

- Full Refund. Returns the total of the original transaction back to the customer.
- Partial Refund. Returns a specified amount of the original transaction back to the customer. For example, if a customer has been overcharged, or where part of an order can't be satisfied by the business.

** Note ** Not all Acquiring banks support partial refunds on the same date as the original transaction.

Parameters — Try it out

Name	Description
Api-Version * required `string` *(header)*	The version of this API. 6.23



Request body required

application/json

Example Value | Schema


```
{
  "receiptId": "1001131610340495360",
  "amount": 10.99,
  "yourPaymentReference": "6482c678-cad3-4efd-b081-aeae7a89a134"
}
```


Responses

Code	Description	Links
200	Successful response	*No links*

Media type: application/json

Examples: RefundReceipt

Controls `Accept` header.

Example Value | Schema


```
{
  "receiptId": "1001131610340495360",
  "originalReceiptId": "1001124307998347264",
  "yourPaymentReference": "6482c678-cad3-4efd-b081-aeae7a89a134",
  "type": "Refund",
  "createdAt": "2025-08-27T10:47:46.5157+01:00",
  "result": "Success",
  "message": "Refund successful",
  "judoId": "100100100",
  "merchantName": "Test Merchant",
  "appearsOnStatementAs": "APL*/TestMerchant       ",
  "originalAmount": "10.99",
  "netAmount": "10.00",
  "amount": "0.99",
  "currency": "GBP",
  "acquirerTransactionId": "33666277256892154000",
  "externalBankResponseCode": "A",
  "authCode": "123456",
```


401	Unauthorized	*No links*

Media type: application/json

Examples: Unauthorized

Example Value | Schema


```
{
  "requestId": "68ea7fab5879f147d51103c3eb265780",
  "message": "Sorry, we were unable to authorize this request. Please check your details and permissions before trying again.",
  "code": 7,
  "category": 1
}
```


403	Forbidden if token used does not have required permission	*No links*

Media type | Examples



Code	Description	Links

application/json | Forbidden

Example Value | Schema


```
{
  "requestId": "68ea7fab5879f147d51103c3eb265780",
  "message": "Sorry, it looks like permissions are not enabled to perform this request. Application permissions can be configured in the judo Dashboard.",
  "code": 84,
  "category": 1
}
```


3DSecure

PUT `/transactions/{receiptId}/resume3ds`

Use /resume3ds when authenticating with 3D Secure 2, following the 3D Secure 2 device details being successfully completed by the consumer.

Parameters | Try it out

Name	Description
Api-Version * required `string` *(header)*	The version of this API. 6.23
receiptId * required `string` *(path)*	Judopay's reference, that identifies the transaction. Get the receiptId from the response of the initial request. 1001131610340495360

Request body required — application/json

Examples: Resume3DS2

Example Value | Schema


```
{
  "cv2": "123",
  "threeDSecure": {
    "methodCompletion": "Yes"
  }
}
```


Responses

Code	Description	Links
200	Successful response Media type	*No links*



Code	Description	Links

application/json

Controls `Accept` header.

Example Value | Schema


```
{
  "receiptId": "1001131610340495360",
  "originalReceiptId": "1001124307998347264",
  "yourPaymentReference": "6482c678-cad3-4efd-b081-aeae7a89a134",
  "type": "PreAuth",
  "createdAt": "2025-02-05T16:28:32.8596+00:00",
  "result": "Success",
  "message": "AuthCode: 123456",
  "judoId": 100100100,
  "merchantName": "Test Merchant",
  "appearsOnStatementAs": "APL*/TestMerchant       ",
  "originalAmount": "10.99",
  "amountCollected": "1.99",
  "netAmount": "1.00",
  "amount": "1.00",
  "currency": "GBP",
  "recurringPaymentType": "MIT",
  "acquirerTransactionId": "33666277256892153705",
```


400	Bad request (Api-Version header missing, or request body attributes invalid)	*No links*
401	Unauthorized	*No links*

Media type: application/json

Examples: Unauthorized

Example Value | Schema


```
{
  "requestId": "68ea7fab5879f147d51103c3eb265780",
  "message": "Sorry, we were unable to authorize this request. Please check your details and permis
sions before trying again.",
  "code": 7,
  "category": 1
}
```


PUT `/transactions/{receiptId}/complete3ds`

Use this endpoint when authenticating with 3D Secure 2, following the 3D Secure 2 challenge being successfully completed by the consumer.

Parameters Try it out

Name	Description
Api-Version * required `string` *(header)*	The version of this API. 6.23



Name	Description
receiptId * required `string` *(path)*	Judopay's reference, that identifies the transaction. Get the receiptId from the response of the initial request. 1001131610340495360

Request body required — application/json

Examples:

Complete3DS2

Example Value | Schema

```
{
  "cv2": "123",
  "version": "2.2.0"
}
```

Responses

Code	Description	Links
200	Successful response Media type: application/json Controls `Accept` header. Example Value | Schema	*No links*
400	Bad request (Api-Version header missing, or request body attributes invalid)	*No links*
401	Unauthorized Media type: application/json Examples: Unauthorized Example Value | Schema	*No links*

```
{
  "receiptId": "1001131610340495360",
  "originalReceiptId": "1001124307998347264",
  "yourPaymentReference": "6482c678-cad3-4efd-b081-aeae7a89a134",
  "type": "PreAuth",
  "createdAt": "2025-02-05T16:28:32.8596+00:00",
  "result": "Success",
  "message": "AuthCode: 123456",
  "judoId": 100100100,
  "merchantName": "Test Merchant",
  "appearsOnStatementAs": "APL*/TestMerchant       ",
  "originalAmount": "10.99",
  "amountCollected": "1.99",
  "netAmount": "1.00",
  "amount": "1.00",
  "currency": "GBP",
  "recurringPaymentType": "MIT",
  "acquirerTransactionId": "33666277256892153705",
```



Code	Description	Links


```
{
  "requestId": "68ea7fab5879f147d51103c3eb265780",
  "message": "Sorry, we were unable to authorize this request. Please check your details and permissions before trying again.",
  "code": 7,
  "category": 1
}
```


Receipts

GET `/transactions/{receiptId}`

Retrieve details of a single transaction. Get the receiptId from the response of the initial payment/preauth request.

Parameters | Try it out

Name	Description
Api-Version * required `string` *(header)*	The version of this API. 6.23
receiptId * required `string` *(path)*	Judopay's reference, that identifies the transaction. Get the receiptId from the response of the initial request. 1001131610340495360

Responses

Code	Description	Links
200	Successful response Media type application/json Controls `Accept` header. **Example Value** | Schema	*No links*


```
{
  "receiptId": "1001131610340495360",
  "originalReceiptId": "1001124307998347264",
  "yourPaymentReference": "6482c678-cad3-4efd-b081-aeae7a89a134",
  "type": "PreAuth",
  "createdAt": "2025-02-05T16:28:32.8596+00:00",
  "result": "Success",
  "message": "AuthCode: 123456",
  "judoId": 100100100,
  "merchantName": "Test Merchant",
  "appearsOnStatementAs": "APL*/TestMerchant       ",
  "originalAmount": "10.99",
  "amountCollected": "1.99",
```



Code	Description	Links


```
  "netAmount": "1.00",
  "amount": "1.00",
  "currency": "GBP",
  "recurringPaymentType": "MIT",
```


401 Unauthorized — *No links*

Media type: application/json | Examples: Unauthorized

Example Value | Schema


```
{
  "requestId": "68ea7fab5879f147d51103c3eb265780",
  "message": "Sorry, we were unable to authorize this request. Please check your details and permissions before trying again.",
  "code": 7,
  "category": 1
}
```


403 Forbidden if token used does not have required permission — *No links*

Media type: application/json | Examples: Forbidden

Example Value | Schema


```
{
  "requestId": "68ea7fab5879f147d51103c3eb265780",
  "message": "Sorry, it looks like permissions are not enabled to perform this request. Application permissions can be configured in the judo Dashboard.",
  "code": 84,
  "category": 1
}
```


404 Not Found — *No links*

Media type: application/json | Examples: Receipt Not Found

Example Value | Schema


```
{
  "requestId": "68ea7fab5879f147d51103c3eb265780",
  "message": "Sorry, the transaction you've specified has not been found. Please check your details and try again.",
  "code": 82,
  "category": 2
}
```


GET /transactions

Return a list of transactions associated with the given query. Please note: The records returned by this endpoint should not be relied upon for real-time processing.

Parameters | Try it out



Name	Description
Api-Version * required **string** *(header)*	The version of this API. 6.23
pageSize **integer** *(query)*	The number of records to display per page (1-500). *Default value* : 10 10
offset **integer** *(query)*	Zero-based index of the first record in page. This should be a multiple of pageSize. e.g. with a pageSize of 100, offset would be 0,100,200 etc. *Default value* : 0 0
sort **string** *(query)*	Determines how the list is sorted. *Available values* : time-descending, time-ascending *Default value* : time-descending time-descending
from **string** *(query)*	Earliest date used to find transactions (DD/MM/YYYY). Start of day in Europe/London timezone is assumed. If from and to are both not specified, then transactions from the past month are queried. 28/01/2025
to **string** *(query)*	Latest date used to find transactions (DD/MM/YYYY). Start of day in Europe/London timezone is assumed. If from and to are both not specified, then transactions from the past month are queried. 15/02/2025
yourPaymentReference **string** *(query)*	If specified, will only match transactions with this yourPaymentReference. d991e31f-81f0-49e0-82aa-80b7e5ffaacd
yourConsumerReference **string** *(query)*	If specified, will only match transactions with this yourConsumerReference. 61588279-231e-4413-a8bc-4ff08088278c

Responses

Code	Description	Links
200	Successful response Media type application/json Controls `Accept` header. **Example Value** | Schema	*No links*

```
{
  "resultCount": 135,
  "pageSize": 10,
  "offset": 0,
```



Code	Description	Links


```
  "results": [
    {
      "receiptId": "1001131610340495360",
      "originalReceiptId": "1001124307998347264",
      "yourPaymentReference": "6482c678-cad3-4efd-b081-aeae7a89a134",
      "type": "PreAuth",
      "createdAt": "2025-02-05T16:28:32.8596+00:00",
      "result": "Success",
      "message": "AuthCode: 123456",
      "judoId": 100100100,
      "appearsOnStatementAs": "APL*/TestMerchant       ",
      "originalAmount": "10.99",
      "amountCollected": "1.99",
```


401 Unauthorized — *No links*

Media type: application/json

Examples: Unauthorized

Example Value | Schema


```
{
  "requestId": "68ea7fab5879f147d51103c3eb265780",
  "message": "Sorry, we were unable to authorize this request. Please check your details and permissions before trying again.",
  "code": 7,
  "category": 1
}
```


403 Forbidden if token used does not have required permission — *No links*

Media type: application/json

Examples: Forbidden

Example Value | Schema


```
{
  "requestId": "68ea7fab5879f147d51103c3eb265780",
  "message": "Sorry, it looks like permissions are not enabled to perform this request. Application permissions can be configured in the judo Dashboard.",
  "code": 84,
  "category": 1
}
```


GET /transactions/{transactionType}

Return a list of transactions of the given transactionType associated with the given query. Please note: The records returned by this endpoint should not be relied upon for real-time processing.

Parameters — Try it out

Name	Description
Api-Version * required `string` *(header)*	The version of this API. 6.23



Name	Description
transactionType * required `string` *(path)*	Type of transactions that are to be returned. *Available values* : payments, preauths, refunds, collections payments
pageSize `integer` *(query)*	The number of records to display per page (1-500). *Default value* : 10 10
offset `integer` *(query)*	Zero-based index of the first record in page. This should be a multiple of pageSize. e.g. with a pageSize of 100, offset would be 0,100,200 etc. *Default value* : 0 0
sort `string` *(query)*	Determines how the list is sorted. *Available values* : time-descending, time-ascending *Default value* : time-descending time-descending
from `string` *(query)*	Earliest date used to find transactions (DD/MM/YYYY). Start of day in Europe/London timezone is assumed. If from and to are both not specified, then transactions from the past month are queried. 28/01/2025
to `string` *(query)*	Latest date used to find transactions (DD/MM/YYYY). Start of day in Europe/London timezone is assumed. If from and to are both not specified, then transactions from the past month are queried. 15/02/2025
yourPaymentReference `string` *(query)*	If specified, will only match transactions with this yourPaymentReference. d991e31f-81f0-49e0-82aa-80b7e5ffaacd
yourConsumerReference `string` *(query)*	If specified, will only match transactions with this yourConsumerReference. 61588279-231e-4413-a8bc-4ff08088278c

Responses

Code	Description	Links
200	Successful response Media type application/json Controls `Accept` header. **Example Value** | Schema	*No links*

```
{
  "resultCount": 135,
  "pageSize": 10,
  "offset": 0,
```



Code	Description	Links


```
  "results": [
    {
      "receiptId": "1001131610340495360",
      "originalReceiptId": "1001124307998347264",
      "yourPaymentReference": "6482c678-cad3-4efd-b081-aeae7a89a134",
      "type": "PreAuth",
      "createdAt": "2025-02-05T16:28:32.8596+00:00",
      "result": "Success",
      "message": "AuthCode: 123456",
      "judoId": 100100100,
      "appearsOnStatementAs": "APL*/TestMerchant       ",
      "originalAmount": "10.99",
      "amountCollected": "1.99",
      "netAmount": "1.00"
```


401 Unauthorized — *No links*

Media type: application/json — Examples: Unauthorized

Example Value | Schema


```
{
  "requestId": "68ea7fab5879f147d51103c3eb265780",
  "message": "Sorry, we were unable to authorize this request. Please check your details and permissions before trying again.",
  "code": 7,
  "category": 1
}
```


403 Forbidden if token used does not have required permission — *No links*

Media type: application/json — Examples: Forbidden

Example Value | Schema


```
{
  "requestId": "68ea7fab5879f147d51103c3eb265780",
  "message": "Sorry, it looks like permissions are not enabled to perform this request. Application permissions can be configured in the judo Dashboard.",
  "code": 84,
  "category": 1
}
```


Payment Sessions

POST `/paymentsession` Create a generic payment session

Create a payment session without specifying a payment operation. webPaymentOperation will be 0.

Parameters — Try it out

Name	Description
Api-Version * required	The version of this API.



Name	Description
`string` *(header)*	6.23

Request body required — application/json

Examples: Session without 3DS

Example Value | Schema


```
{
  "judoId": "100100100",
  "yourConsumerReference": "2b45fd3f-cee5-4e7e-874f-28051db65408",
  "yourPaymentReference": "6482c678-cad3-4efd-b081-aeae7a89a134",
  "yourPaymentMetaData": {
    "internalLocationRef": "Example",
    "internalId": 99
  },
  "currency": "GBP",
  "amount": 10.99,
  "cardAddress": {
    "address1": "CardHolder House",
    "address2": "1 CardHolder Street",
    "town": "CardHolder Town",
    "postCode": "AB1 2CD",
    "countryCode": "826",
    "cardHolderName": "John Doe"
  },
```


Responses

Code	Description	Links
200	Successful response Media type: application/json Controls `Accept` header. Example Value | Schema	*No links*
400	Bad request (Api-Version header missing, or request body attributes invalid)	*No links*
401	Unauthorized Media type: application/json Examples: Unauthorized Example Value | Schema	*No links*


```
{
  "expiryDate": "2026-02-05T16:28:32.8596+00:00",
  "reference": "5QcAAAQAAAAPAAAACAAAABtGgvhBrF9BHTN7nqn1e0J4hVVmi-y27dGPjWBMtls3Gj_XDg"
}
```



```
{
  "requestId": "68ea7fab5879f147d51103c3eb265780",
```



Code	Description	Links


```
  "message": "Sorry, we were unable to authorize this request. Please check your details and permis
sions before trying again.",
  "code": 7,
  "category": 1
}
```


403	Forbidden if token used does not have required permission	*No links*

Media type: application/json

Examples: Forbidden

Example Value | Schema


```
{
  "requestId": "68ea7fab5879f147d51103c3eb265780",
  "message": "Sorry, it looks like permissions are not enabled to perform this request. Application
permissions can be configured in the judo Dashboard.",
  "code": 84,
  "category": 1
}
```


PUT /paymentsession/{reference}/cancel

Cancel an open payment session

Update the status of an Open payment session to Cancelled, preventing it from being used in the future.

Parameters | Try it out

Name	Description
Api-Version * required `string` *(header)*	The version of this API. 6.23
reference * required `string` *(path)*	Judo reference for a payment session, returned in response to creation of session. 5QcAAAMAAAAKAAAACwAAADFDclhHlx3-c`

Request body required — application/json

Example Value | Schema


```
{}
```


Responses

Code	Description	Links
200	Successful response	*No links*



Code	Description	Links

Media type

application/json

Controls `Accept` header.

Example Value | Schema


```
{
  "reference": "5QcAAAQAAAAPAAAACAAAABtGgvhBrF9BHTN7nqn1e0J4hVVmi-y27dGPjWBMtls3Gj_XDg",
  "status": "Cancelled"
}
```


400 Bad request (Api-Version header missing, or payment session is not in Open status) *No links*

Media type: application/json | Examples: Payment Session not open

Example Value | Schema


```
{
  "requestId": "68ea7fab5879f147d51103c3eb265780",
  "message": "Status update only allowed for open payment sessions. Please refer to Judo API documentation for more details.",
  "code": 176,
  "category": 1
}
```


401 Unauthorized *No links*

Media type: application/json | Examples: Unauthorized

Example Value | Schema


```
{
  "requestId": "68ea7fab5879f147d51103c3eb265780",
  "message": "Sorry, we were unable to authorize this request. Please check your details and permissions before trying again.",
  "code": 7,
  "category": 1
}
```


403 Forbidden if token used does not have required permission *No links*

Media type: application/json | Examples: Forbidden

Example Value | Schema


```
{
  "requestId": "68ea7fab5879f147d51103c3eb265780",
  "message": "Sorry, it looks like permissions are not enabled to perform this request. Application permissions can be configured in the judo Dashboard.",
  "code": 84,
  "category": 1
}
```


404 Payment session reference not found *No links*

Media type: application/json | Examples: Reference not found

Example Value | Schema



Code	Description	Links

```
{
  "requestId": "68ea7fab5879f147d51103c3eb265780",
  "message": "Sorry, but it appears the web payment reference you've specified is not valid.",
  "code": 76,
  "category": 2
}
```

GET `/transactions/{receiptId}/webpayment` Retrieve details of a completed payment session

Retrieve details of a completed payment session matching the given receiptId.

Parameters Try it out

Name	Description
Api-Version * required `string` *(header)*	The version of this API. 6.23
receiptId * required `string` *(path)*	Judopay's reference, that identifies the transaction. Get the receiptId from the response of the initial request. 1001131610340495360

Responses

Code	Description	Links
200	Successful response Media type application/json Controls `Accept` header. **Example Value** | Schema	*No links*

```
{
  "amount": 10.99,
  "cardAddress": {
    "address1": "CardHolder House",
    "address2": "1 CardHolder Street",
    "town": "CardHolder Town",
    "postCode": "AB1 2CD",
    "state": "FL",
    "countryCode": 826,
    "cardHolderName": "John Doe"
  },
  "clientIpAddress": "1.2.3.4",
  "clientUserAgent": "Mozilla/5.0",
  "companyName": "Test Merchant",
  "currency": "GBP",
  "expiryDate": "2026-02-05T16:28:32.8596+00:00",
```



Code	Description	Links
	"judoId": "100100100", "paymentCancelUrl": "https://my.web.site/cancel",	
400	Bad request (Api-Version header missing)	*No links*

401	Unauthorized
	Media type: application/json
	Examples: Unauthorized
	Example Value | Schema
	{
	"requestId": "68ea7fab5879f147d51103c3eb265780",
	"message": "Sorry, we were unable to authorize this request. Please check your details and permissions before trying again.",
	"code": 7,
	"category": 1
	}
		No links

403	Forbidden if token used does not have required permission
	Media type: application/json
	Examples: Forbidden
	Example Value | Schema
	{
	"requestId": "68ea7fab5879f147d51103c3eb265780",
	"message": "Sorry, it looks like permissions are not enabled to perform this request. Application permissions can be configured in the judo Dashboard.",
	"code": 84,
	"category": 1
	}
		No links

404	ReceiptId not found, or not associated with a payment session, or doesn't match authentication details.	*No links*

GET `/webpayments` Return a list of payment sessions

Return a list of payment sessions matching the given query.

Parameters | Try it out

Name	Description
Api-Version * required `string` *(header)*	The version of this API. 6.23
pageSize `integer` *(query)*	The number of records to display per page (1-500). *Default value* : 10 10
offset `integer`	Zero-based index of the first record in page. This should be a multiple of pageSize. e.g. with a pageSize of 100, offset would be 0,100,200 etc.



Name	Description
(query)	*Default value* : 0 0
sort **string** *(query)*	Determines how the list is sorted. *Available values* : time-descending, time-ascending *Default value* : time-descending time-descending

Responses

Code	Description	Links
200	Successful response Media type application/json Controls `Accept` header. **Example Value** | Schema	*No links*

```
{
  "resultCount": 135,
  "pageSize": 10,
  "offset": 0,
  "results": [
    {
      "amount": 10.99,
      "cardAddress": {
        "address1": "CardHolder House",
        "address2": "1 CardHolder Street",
        "town": "CardHolder Town",
        "postCode": "AB1 2CD",
        "state": "FL",
        "countryCode": 826,
        "cardHolderName": "John Doe"
      },
      "clientIpAddress": "1.2.3.4",
      "clientUserAgent": "Mozilla/5.0",
```

400	Bad request (Api-Version header missing)	*No links*
401	Unauthorized Media type: application/json Examples: Unauthorized **Example Value** | Schema	*No links*

```
{
  "requestId": "68ea7fab5879f147d51103c3eb265780",
  "message": "Sorry, we were unable to authorize this request. Please check your details and permis
sions before trying again.",
  "code": 7,
```



Code	Description	Links

```
  "category": 1
}
```

403 Forbidden if token used does not have required permission — *No links*

Media type: application/json

Examples: Forbidden

Example Value | Schema

```
{
  "requestId": "68ea7fab5879f147d51103c3eb265780",
  "message": "Sorry, it looks like permissions are not enabled to perform this request. Application permissions can be configured in the judo Dashboard.",
  "code": 84,
  "category": 1
}
```

GET /webpayments/{reference} Retrieve the details of a single payment session

Return the details of a single payment session with the given reference.

Parameters — Try it out

Name	Description
Api-Version * required `string` *(header)*	The version of this API. 6.23
reference * required `string` *(path)*	Judo reference for a payment session, returned in response to creation of session. 5QcAAAMAAAAKAAAACwAAADFDcIhHlx3-c`

Responses

Code	Description	Links

200 Successful response — *No links*

Media type: application/json

Controls `Accept` header.

Example Value | Schema

```
{
  "amount": 10.99,
  "cardAddress": {
    "address1": "CardHolder House",
    "address2": "1 CardHolder Street",
    "town": "CardHolder Town",
```



Code	Description	Links


```
    "postCode": "AB1 2CD",
    "state": "FL",
    "countryCode": 826,
    "cardHolderName": "John Doe"
  },
  "clientIpAddress": "1.2.3.4",
  "clientUserAgent": "Mozilla/5.0",
  "companyName": "Test Merchant",
  "currency": "GBP",
  "expiryDate": "2026-02-05T16:28:32.8596+00:00",
  "judoId": "100100100",
```


400 Bad request (Api-Version header missing) *No links*

401 Unauthorized *No links*

Media type: application/json | Examples: Unauthorized

Example Value | Schema


```
{
  "requestId": "68ea7fab5879f147d51103c3eb265780",
  "message": "Sorry, we were unable to authorize this request. Please check your details and permissions before trying again.",
  "code": 7,
  "category": 1
}
```


403 Forbidden if token used does not have required permission *No links*

Media type: application/json | Examples: Forbidden

Example Value | Schema


```
{
  "requestId": "68ea7fab5879f147d51103c3eb265780",
  "message": "Sorry, it looks like permissions are not enabled to perform this request. Application permissions can be configured in the judo Dashboard.",
  "code": 84,
  "category": 1
}
```


404 Reference not found, or doesn't match authentication details. *No links*

GET /webpayments/payments/{reference}

Retrieve the details of a single payment session for a payment

Return the details of a single payment session associated with a payment with the given reference.

Parameters | Try it out

Name	Description
Api-Version * required `string`	The version of this API.



Name	Description
(header)	6.23
reference * required `string` *(path)*	Judo reference for a payment session, returned in response to creation of session. 5QcAAAMAAAAKAAAACwAAADFDclhHlx3-c`

Responses

Code	Description	Links
200	Successful response	*No links*

Media type: application/json

Controls `Accept` header.

Example Value | Schema


```
{
  "amount": 10.99,
  "cardAddress": {
    "address1": "CardHolder House",
    "address2": "1 CardHolder Street",
    "town": "CardHolder Town",
    "postCode": "AB1 2CD",
    "state": "FL",
    "countryCode": 826,
    "cardHolderName": "John Doe"
  },
  "clientIpAddress": "1.2.3.4",
  "clientUserAgent": "Mozilla/5.0",
  "companyName": "Test Merchant",
  "currency": "GBP",
  "expiryDate": "2026-02-05T16:28:32.8596+00:00",
  "judoId": "100100100",
  "paymentCancelUrl": "https://my.web.site/cancel",
```


400	Bad request (Api-Version header missing)	*No links*
401	Unauthorized	*No links*

Media type: application/json — Examples: Unauthorized

Example Value | Schema


```
{
  "requestId": "68ea7fab5879f147d51103c3eb265780",
  "message": "Sorry, we were unable to authorize this request. Please check your details and permissions before trying again.",
  "code": 7,
  "category": 1
}
```


403	Forbidden if token used does not have required permission	*No links*

Media type — Examples



Code	Description	Links
	application/json ⌄ Forbidden ⌄	

Example Value | Schema


```
{
  "requestId": "68ea7fab5879f147d51103c3eb265780",
  "message": "Sorry, it looks like permissions are not enabled to perform this request. Application permissions can be configured in the judo Dashboard.",
  "code": 84,
  "category": 1
}
```


404	Reference not found, or doesn't match authentication details, or not for a payment	*No links*

GET `/webpayments/preauths/{reference}` Retrieve the details of a single payment session for a preauth

Return the details of a single payment session associated with a pre-authorisation with the given reference.

Parameters Try it out

Name	Description
Api-Version * required `string` *(header)*	The version of this API. 6.23
reference * required `string` *(path)*	Judo reference for a payment session, returned in response to creation of session. 5QcAAAMAAAAKAAAACwAAADFDcIhHlx3-c`

Responses

Code	Description	Links
200	Successful response Media type application/json ⌄ Controls `Accept` header.	*No links*

Example Value | Schema


```
{
  "amount": 10.99,
  "cardAddress": {
    "address1": "CardHolder House",
    "address2": "1 CardHolder Street",
    "town": "CardHolder Town",
    "postCode": "AB1 2CD",
    "state": "FL",
    "countryCode": 826,
    "cardHolderName": "John Doe"
```



Code	Description	Links


```
  },
  "clientIpAddress": "1.2.3.4",
  "clientUserAgent": "Mozilla/5.0",
  "companyName": "Test Merchant",
  "currency": "GBP",
  "expiryDate": "2026-02-05T16:28:32.8596+00:00",
  "judoId": "100100100",
  "paymentCancelUrl": "https://my.web.site/cancel",
```


400 Bad request (Api-Version header missing) *No links*

401 Unauthorized *No links*

Media type: application/json

Examples: Unauthorized

Example Value | Schema


```
{
  "requestId": "68ea7fab5879f147d51103c3eb265780",
  "message": "Sorry, we were unable to authorize this request. Please check your details and permissions before trying again.",
  "code": 7,
  "category": 1
}
```


403 Forbidden if token used does not have required permission *No links*

Media type: application/json

Examples: Forbidden

Example Value | Schema


```
{
  "requestId": "68ea7fab5879f147d51103c3eb265780",
  "message": "Sorry, it looks like permissions are not enabled to perform this request. Application permissions can be configured in the judo Dashboard.",
  "code": 84,
  "category": 1
}
```


404 Reference not found, or doesn't match authentication details, or not for a preauth *No links*

POST /webpayments/checkcard Create a session for check card

Create a session for a check card operation (zero amount pre-authorisation). transactionType will be CHECKCARD, webPaymentOperation will be 1.

Parameters Try it out

Name	Description
Api-Version * required `string` *(header)*	The version of this API. 6.23



Request body required

application/json

Examples:

Session without 3DS

Example Value | Schema


```
{
  "judoId": "100100100",
  "yourConsumerReference": "2b45fd3f-cee5-4e7e-874f-28051db65408",
  "yourPaymentReference": "6482c678-cad3-4efd-b081-aeae7a89a134",
  "yourPaymentMetaData": {
    "internalLocationRef": "Example",
    "internalId": 99
  },
  "currency": "GBP",
  "amount": 10.99,
  "cardAddress": {
    "address1": "CardHolder House",
    "address2": "1 CardHolder Street",
    "town": "CardHolder Town",
    "postCode": "AB1 2CD",
    "countryCode": "826",
    "cardHolderName": "John Doe"
  },
```


Responses

Code	Description	Links
200	Successful response Media type: application/json Controls `Accept` header. Example Value | Schema	*No links*
400	Bad request (Api-Version header missing, or request body attributes invalid)	*No links*
401	Unauthorized Media type: application/json Examples: Unauthorized Example Value | Schema	*No links*


```
{
  "payByLinkUrl": "https://pay-sandbox.judopay.com/ABC123",
  "postUrl": "https://pay-sandbox.judopay.com/v2",
  "reference": "5QcAAAQAAAAPAAAACAAAABtGgvhBrF9BHTN7nqn1e0J4hVVmi-y27dGPjWBMtls3Gj_XDg"
}
```



```
{
  "requestId": "68ea7fab5879f147d51103c3eb265780",
  "message": "Sorry, we were unable to authorize this request. Please check your details and permissions before trying again.",
  "code": 7,
  "category": 1
}
```



Code	Description	Links
403	Forbidden if token used does not have required permission	*No links*

Media type: application/json

Examples: Forbidden

Example Value | Schema

```
{
  "requestId": "68ea7fab5879f147d51103c3eb265780",
  "message": "Sorry, it looks like permissions are not enabled to perform this request. Application permissions can be configured in the judo Dashboard.",
  "code": 84,
  "category": 1
}
```

POST /webpayments/payments Create a session for a payment

Create a session for a payment operation. transactionType will be Payment, webPaymentOperation will be 0.

Parameters | Try it out

Name	Description
Api-Version * required `string` *(header)*	The version of this API. 6.23

Request body required — application/json

Examples: Session without 3DS

Example Value | Schema

```
{
  "judoId": "100100100",
  "yourConsumerReference": "2b45fd3f-cee5-4e7e-874f-28051db65408",
  "yourPaymentReference": "6482c678-cad3-4efd-b081-aeae7a89a134",
  "yourPaymentMetaData": {
    "internalLocationRef": "Example",
    "internalId": 99
  },
  "currency": "GBP",
  "amount": 10.99,
  "cardAddress": {
    "address1": "CardHolder House",
    "address2": "1 CardHolder Street",
    "town": "CardHolder Town",
    "postCode": "AB1 2CD",
    "countryCode": "826",
    "cardHolderName": "John Doe"
  },
```



Responses

Code	Description	Links
200	Successful response Media type application/json Controls `Accept` header. **Example Value** | Schema `{ "payByLinkUrl": "https://pay-sandbox.judopay.com/ABC123", "postUrl": "https://pay-sandbox.judopay.com/v2", "reference": "5QcAAAQAAAAPAAAACAAAABtGgvhBrF9BHTN7nqn1e0J4hVVmi-y27dGPjWBMtls3Gj_XDg" }`	*No links*
400	Bad request (Api-Version header missing, or request body attributes invalid)	*No links*
401	Unauthorized Media type: application/json Examples: Unauthorized **Example Value** | Schema `{ "requestId": "68ea7fab5879f147d51103c3eb265780", "message": "Sorry, we were unable to authorize this request. Please check your details and permissions before trying again.", "code": 7, "category": 1 }`	*No links*
403	Forbidden if token used does not have required permission Media type: application/json Examples: Forbidden **Example Value** | Schema `{ "requestId": "68ea7fab5879f147d51103c3eb265780", "message": "Sorry, it looks like permissions are not enabled to perform this request. Application permissions can be configured in the judo Dashboard.", "code": 84, "category": 1 }`	*No links*

POST /webpayments/preauths Create a session for a preauth

Create a session for a pre-authorisation. transactionType will be PREAUTH, webPaymentOperation will be 0.

Parameters

Try it out



Name	Description
Api-Version * required `string` *(header)*	The version of this API. 6.23

Request body required — application/json

Examples: Session without 3DS

Example Value | Schema

```
{
  "judoId": "100100100",
  "yourConsumerReference": "2b45fd3f-cee5-4e7e-874f-28051db65408",
  "yourPaymentReference": "6482c678-cad3-4efd-b081-aeae7a89a134",
  "yourPaymentMetaData": {
    "internalLocationRef": "Example",
    "internalId": 99
  },
  "currency": "GBP",
  "amount": 10.99,
  "cardAddress": {
    "address1": "CardHolder House",
    "address2": "1 CardHolder Street",
    "town": "CardHolder Town",
    "postCode": "AB1 2CD",
    "countryCode": "826",
    "cardHolderName": "John Doe"
  },
```

Responses

Code	Description	Links
200	Successful response Media type: application/json Controls `Accept` header. Example Value | Schema	*No links*
400	Bad request (Api-Version header missing, or request body attributes invalid)	*No links*
401	Unauthorized Media type: application/json Examples: Unauthorized	*No links*

```
{
  "payByLinkUrl": "https://pay-sandbox.judopay.com/ABC123",
  "postUrl": "https://pay-sandbox.judopay.com/v2",
  "reference": "5QcAAAQAAAAPAAAACAAAABtGgvhBrF9BHTN7nqn1e0J4hVVmi-y27dGPjWBMtls3Gj_XDg"
}
```



Code	Description	Links

Example Value | Schema

```
{
  "requestId": "68ea7fab5879f147d51103c3eb265780",
  "message": "Sorry, we were unable to authorize this request. Please check your details and permissions before trying again.",
  "code": 7,
  "category": 1
}
```

403 Forbidden if token used does not have required permission — *No links*

Media type: application/json

Examples: Forbidden

Example Value | Schema

```
{
  "requestId": "68ea7fab5879f147d51103c3eb265780",
  "message": "Sorry, it looks like permissions are not enabled to perform this request. Application permissions can be configured in the judo Dashboard.",
  "code": 84,
  "category": 1
}
```

Other

GET `/info`

Obtain information about service, including latest supported Api-Version

Parameters — Try it out

Name	Description
Api-Version * required `string` *(header)*	The version of this API. 6.23

Responses

Code	Description	Links
200	Successful response	*No links*

Media type: application/json

Controls `Accept` header.

Example Value | Schema

```
{
  "version": "6.23.0.0"
```



Code	Description	Links
	<pre>}</pre>	
400	Bad request (Api-Version header missing)	*No links*
401	Unauthorized	*No links*

Media type: application/json

Examples: Unauthorized

Example Value | Schema


```
{
  "requestId": "68ea7fab5879f147d51103c3eb265780",
  "message": "Sorry, we were unable to authorize this request. Please check your details and permis
sions before trying again.",
  "code": 7,
  "category": 1
}
```


Schemas

allowIncrementRequest {

`allowIncrement` — boolean

example: true

Set to true to specify a customer initiated preauth can be incremented by the merchant before it is fully collected.

}

networkTokenRequest {

`disableNetworkTokenisation` — boolean

default: false

example: true

(Optional) Set to true to specify that network tokenisation should not be used for this transaction, even if network token registration has been enabled on the account.

}



applePayRequest {
pkPayment* {
token* {
paymentMethod {
displayName string
example: Visa 1234
Card description suitable for display, provided by t
network string
example: Visa
Name of the payment network backing the card, provid
type string
example: debit
Card's type of payment, provided by the ApplePay ses
}
paymentData* {
data* string
example: CXnUZAtPGWfkLGsHksn5WTzfFhEzNMqSJZXKTmbaWGYnyB8z3Zt4f
Encrypted payment data.
signature* string
example:
gbtqw7Zy9LcfsV53pSRxDJjwUKmrLAWkxVxL6NQzzChwGRMQ3dncrEb69HuyR3
Signature of the payment and header data. The signature includ
header {
description: Additional version-dependent informa
publicKeyHash string
example: EMq6ggcTqGNYn86U9fd6kC5gdN2
ephemeralPublicKey string
example: Y5q3gWSJL8byWFHkWtx8nzvMNkq
transactionId string
example: 7c3b36d16bb0f6eac87e15359be
}
version string
example: EC_V1
Version information about the payment token. The token uses EC
}
}
}
cardAddress cardAddressRequestAttributes {
description: Card holder address.
address1 string
pattern: ^[^<>\\]+$
example: CardHolder House
The first line of the cardholder's billing address. Mastercard recommends providin
address2 string
pattern: ^[^<>\\]+$
example: 1 CardHolder Street
Second line of the card holder's address. Max 50 characters for a 3DS2 enabled API
town string
pattern: ^[^<>\\]+$
example: CardHolder Town
Town of the card holder's address. Max 40 characters for a 3DS2 enabled API token.
postCode* string
pattern: ^[0-9a-zA-Z -]*$
example: AB1 2CD
Postcode of the card holder's address.
state string
pattern: ^[A-Z]{2}$
example: FL



This attribute can be used by merchants to provide state/province/territory ISO co

countryCode
string
pattern: ^\d{3}$
example: 826
ISO 3166-1 (3 digit) country code of the card holder's address.

}
}

cardAddressCommonAttributes {

description: Card holder address.

address1
string
pattern: ^[^<>\\]+$
example: CardHolder House
The first line of the cardholder's billing address. Mastercard recommends providing this field for Mastercard 3DS-authenticated transactions from 1 July 2026. Supplying the billing address may improve issuer authentication and authorisation decisions. Max 50 characters for a 3DS2 enabled API token.

address2
string
pattern: ^[^<>\\]+$
example: 1 CardHolder Street
Second line of the card holder's address. Max 50 characters for a 3DS2 enabled API token.

town
string
pattern: ^[^<>\\]+$
example: CardHolder Town
Town of the card holder's address. Max 40 characters for a 3DS2 enabled API token.

postCode*
string
pattern: ^[0-9a-zA-Z -]$*
example: AB1 2CD
Postcode of the card holder's address.

state
string
pattern: ^[A-Z]{2}$
example: FL
This attribute can be used by merchants to provide state/province/territory ISO codes (2-char alpha) - specifically for countries where this is required as part of the address (Eg: US, Canada, India, China).

}



cardAddressRequestAttributes ⌄ {

description:	Card holder address.
address1	⌄ string *pattern: ^[^<>\\]+$* *example: CardHolder House* The first line of the cardholder's billing address. Mastercard recommends providing this field for Mastercard 3DS-authenticated transactions from 1 July 2026. Supplying the billing address may improve issuer authentication and authorisation decisions. Max 50 characters for a 3DS2 enabled API token.
address2	⌄ string *pattern: ^[^<>\\]+$* *example: 1 CardHolder Street* Second line of the card holder's address. Max 50 characters for a 3DS2 enabled API token.
town	⌄ string *pattern: ^[^<>\\]+$* *example: CardHolder Town* Town of the card holder's address. Max 40 characters for a 3DS2 enabled API token.
postCode*	⌄ string *pattern: ^[0-9a-zA-Z -]*$* *example: AB1 2CD* Postcode of the card holder's address.
state	⌄ string *pattern: ^[A-Z]{2}$* *example: FL* This attribute can be used by merchants to provide state/province/territory ISO codes (2-char alpha) - specifically for countries where this is required as part of the address (Eg: US, Canada, India, China).
countryCode	⌄ string *pattern: ^\d{3}$* *example: 826* ISO 3166-1 (3 digit) country code of the card holder's address.

}



cardAddressRequest {

cardAddress **cardAddressRequestAttributes** {

description: Card holder address.

address1 string

pattern: ^[^<>\\]+$

example: CardHolder House

The first line of the cardholder's billing address. Mastercard recommends providing this field for Mastercard 3DS-authenticated transactions from 1 July 2026. Supplying the billing address may improve issuer authentication and authorisation decisions. Max 50 characters for a 3DS2 enabled API token.

address2 string

pattern: ^[^<>\\]+$

example: 1 CardHolder Street

Second line of the card holder's address. Max 50 characters for a 3DS2 enabled API token.

town string

pattern: ^[^<>\\]+$

example: CardHolder Town

Town of the card holder's address. Max 40 characters for a 3DS2 enabled API token.

postCode* string

pattern: ^[0-9a-zA-Z -]$*

example: AB1 2CD

Postcode of the card holder's address.

state string

pattern: ^[A-Z]{2}$

example: FL

This attribute can be used by merchants to provide state/province/territory ISO codes (2-char alpha) - specifically for countries where this is required as part of the address (Eg: US, Canada, India, China).

countryCode string

pattern: ^\d{3}$

example: 826

ISO 3166-1 (3 digit) country code of the card holder's address.

}

}



cardAddressResponseAttributes {

description: Card holder address.

address1

string

pattern: ^[^<>\\]+$

example: CardHolder House

The first line of the cardholder's billing address. Mastercard recommends providing this field for Mastercard 3DS-authenticated transactions from 1 July 2026. Supplying the billing address may improve issuer authentication and authorisation decisions. Max 50 characters for a 3DS2 enabled API token.

address2

string

pattern: ^[^<>\\]+$

example: 1 CardHolder Street

Second line of the card holder's address. Max 50 characters for a 3DS2 enabled API token.

town

string

pattern: ^[^<>\\]+$

example: CardHolder Town

Town of the card holder's address. Max 40 characters for a 3DS2 enabled API token.

postCode*

string

pattern: ^[0-9a-zA-Z -]$*

example: AB1 2CD

Postcode of the card holder's address.

state

string

pattern: ^[A-Z]{2}$

example: FL

This attribute can be used by merchants to provide state/province/territory ISO codes (2-char alpha) - specifically for countries where this is required as part of the address (Eg: US, Canada, India, China).

countryCode

integer

pattern: ^\d{2,3}$

example: 826

ISO 3166-1 (3 digit) country code of the card holder's address. If the first digit of the of the code is 0 this will not be shown

}



cardAddressWebPaymentRequest ⌄ {

description: Card holder address.

address1 ⌄ string
pattern: ^[^<>\\]+$
example: CardHolder House
The first line of the cardholder's billing address. Mastercard recommends providing this field for Mastercard 3DS-authenticated transactions from 1 July 2026. Supplying the billing address may improve issuer authentication and authorisation decisions. Max 50 characters for a 3DS2 enabled API token.

address2 ⌄ string
pattern: ^[^<>\\]+$
example: 1 CardHolder Street
Second line of the card holder's address. Max 50 characters for a 3DS2 enabled API token.

town ⌄ string
pattern: ^[^<>\\]+$
example: CardHolder Town
Town of the card holder's address. Max 40 characters for a 3DS2 enabled API token.

postCode* ⌄ string
pattern: ^[0-9a-zA-Z -]$*
example: AB1 2CD
Postcode of the card holder's address.

state ⌄ string
pattern: ^[A-Z]{2}$
example: FL
This attribute can be used by merchants to provide state/province/territory ISO codes (2-char alpha) - specifically for countries where this is required as part of the address (Eg: US, Canada, India, China).

countryCode ⌄ string
pattern: ^\d{3}$
example: 826
ISO 3166-1 (3 digit) country code of the card holder's address.

cardHolderName ⌄ string
example: John Doe
The cardholder's full name. Mastercard recommends providing this field for Mastercard 3DS-authenticated transactions from 1 July 2026. Supplying the cardholder name may improve issuer authentication and authorisation decisions.

}

cardDetailsCv2NotRequiredRequest ⌄ {

cardNumber* ⌄ string
example: 1234567890123456
REQUIRED. The unique number printed on a credit card (13 to 19 digits depending on card type). Submitted without whitespace or non-numeric characters.

cv2 ⌄ string
minLength: 3
maxLength: 4
pattern: ^\d{3,4}$
example: 123
The 3 or 4 digit number on the back of a credit card. Also known as the card verification value (CVV) or security code. If not specified, contact Judopay support to confirm gateway settings are configured to reflect this.

expiryDate* ⌄ string
pattern: ^\d{2}(\/)?\d{2}$
example: 01/28
REQUIRED. The expiry date of the card in format MM/YY

}



cardDetailsCv2RequiredRequest {

cardNumber* string
example: 1234567890123456
REQUIRED. The unique number printed on a credit card (13 to 19 digits depending on card type). Submitted without whitespace or non-numeric characters.

cv2* string
minLength: 3
maxLength: 4
pattern: ^\d{3,4}$
example: 123
REQUIRED. The 3 or 4 digit number on the back of a credit card. Also known as the card verification value (CVV) or security code.

expiryDate* string
pattern: ^\d{2}(\/)?\d{2}$
example: 01/28
REQUIRED. The expiry date of the card in format MM/YY

}

cardDetailsStartDateRequest {

startDate string
pattern: ^\d{2}(\/)?\d{2}$
example: 07/20
For Maestro cards only, start date in format MM/YY

}



cardDetailsResponse {

cardLastfour
string
example: 1234
Last four digits of card

endDate
string
example: 1225
Card expiry date in format MMYY

cardToken
string
example: Ck3AeNlBfjvs9d61MNZiG0gtCvijqvKr
Tokenized representation of card which can be used in future transactions instead of card number

cardType
integer
example: 2
Integer representing the type of card, see Reference Codes - Card Types on https://docs.judopay.com/Content/Developer%20Tools/Codes.htm#CardTypes

startDate
string
example: 0121
Card start date in format MMYY

cardScheme
string
example: Mastercard
Name of card scheme (determined from matching start of card number against issuer data)

cardFunding
string
example: Credit
Type of card (determined from matching start of card number against issuer data)

cardCategory
string
example: Acquirer Only
Category of card (determined from matching start of card number against issuer data)

cardCountry
string
example: GB
ISO 3166-1 alpha-2 code of country that issued the card (determined from matching start of card number against issuer data)

bank
string
example: Santander Uk Plc
Name of bank that issued the card (determined from matching start of card number against issuer data)

cardHolderName
string
example: John Doe
Name of card holder as supplied on request

ownerType
string
example: Personal
[Conditional] Indicator of whether a card is personal or commercial, if known

Enum:
[Personal, Commercial, null]

}



cardDetailsListResponse ⌄ {

cardLastfour ⌄ string
example: 1234
Last four digits of card

endDate ⌄ string
example: 1225
Card expiry date in format MMYY

cardType ⌄ integer
example: 2
Integer representing the type of card, see Reference Codes - Card Types on https://docs.judopay.com/Content/Developer%20Tools/Codes.htm#CardTypes

}



cardTokenPreAuthPaymentRequest {

cardToken* string
example: Cw8eZ7nLBM2rFXx12WiRI0YveZPinkPm
REQUIRED. A randomly generated string linked to a card saved securely within the Judopay Card Vault.

cv2 string
minLength: 3
maxLength: 4
pattern: ^\d{3,4}$
example: 123
The 3 or 4 digit number on the back of a credit card. Also known as the card verification value (CVV) or security code.

cardAddress **cardAddressRequestAttributes** {
description: Card holder address.

address1 string
pattern: ^[^<>\\]+$
example: CardHolder House
The first line of the cardholder's billing address. Mastercard recommends providing this field for Mastercard 3DS-authenticated transactions from 1 July 2026. Supplying the billing address may improve issuer authentication and authorisation decisions. Max 50 characters for a 3DS2 enabled API token.

address2 string
pattern: ^[^<>\\]+$
example: 1 CardHolder Street
Second line of the card holder's address. Max 50 characters for a 3DS2 enabled API token.

town string
pattern: ^[^<>\\]+$
example: CardHolder Town
Town of the card holder's address. Max 40 characters for a 3DS2 enabled API token.

postCode* string
pattern: ^[0-9a-zA-Z -]$*
example: AB1 2CD
Postcode of the card holder's address.

state string
pattern: ^[A-Z]{2}$
example: FL
This attribute can be used by merchants to provide state/province/territory ISO codes (2-char alpha) - specifically for countries where this is required as part of the address (Eg: US, Canada, India, China).

countryCode string
pattern: ^\d{3}$
example: 826
ISO 3166-1 (3 digit) country code of the card holder's address.

}
}

challengeRequestIndicatorRequest string
nullable: true
default: noPreference
example: challengePreferred

Indicates whether a challenge is requested for this transaction (this may be over-ruled by the issuer). Should not be specified in the same request as scaExemption

Enum:

[noPreference, noChallenge, challengePreferred, challengeAsMandate, challengeWithWhitelistPrompt, null]



checkCardRequestPan {

cardNumber*
string
example: 1234567890123456
REQUIRED. The unique number printed on a credit card (13 to 19 digits depending on card type). Submitted without whitespace or non-numeric characters.

cv2
string
minLength: 3
maxLength: 4
pattern: ^\d{3,4}$
example: 123
The 3 or 4 digit number on the back of a credit card. Also known as the card verification value (CVV) or security code. If not specified, contact Judopay support to confirm gateway settings are configured to reflect this.

expiryDate*
string
pattern: ^\d{2}(\/)?\d{2}$
example: 01/28
REQUIRED. The expiry date of the card in format MM/YY

startDate
string
pattern: ^\d{2}(\/)?\d{2}$
example: 07/20
For Maestro cards only, start date in format MM/YY

yourConsumerReference*
string
minLength: 1
maxLength: 50
example: 2b45fd3f-cee5-4e7e-874f-28051db65408
REQUIRED. Unique reference to anonymously identify your customer. Advisable to use GUIDs.

yourPaymentReference
string
minLength: 1
maxLength: 50
example: 6482c678-cad3-4efd-b081-aeae7a89a134
Your unique reference for this payment. This value should be unique in order to protect your customers against duplicate transactions. If not supplied, then a unique id will be generated internally

clientDetails
{
description: A set of fraud signals sent by the mobile SDK
key: string
value: string
}

judoId
string
maxLength: 9
pattern: ^\d{9}$
example: 100100100
Unique merchant and/or location ID supplied by Judopay. Do not include spaces or dashes. If not specified, then your default judoId will be used.

currency
string
example: GBP
The currency to use for the pre-authorisation, only required if different from the currency set during onboarding (default is GBP)

initialRecurringPayment
boolean
example: false
Indicates if this is the first transaction that forms part of a recurring payment schedule. This flag should be set to 'true' if this is a customer-initiated payment which will be followed by a series of scheduled recurring payments.

disableNetworkTokenisation
boolean
default: false
example: true
(Optional) Set to true to specify that network tokenisation should not be used for this transaction, even if network token registration has been enabled on the account.

cardHolderName
string
pattern: ^.{0,45}$
example: John Doe
The cardholder's full name. Mastercard recommends providing this field for Mastercard 3DS-authenticated transactions from 1 July 2026. Supplying the cardholder name may improve issuer authentication and authorisation decisions. Note: When testing in the sandbox



environment, it is the cardHolderName that is used to determine the test card for 3D Secure 2 authentication.

emailAddress

string
pattern: ^.{0,254}$
example: test.user@judopay.com
The cardholder's email address. Mastercard recommends providing at least one contact method for Mastercard 3DS-authenticated transactions from 1 July 2026. A contact method can be an email address or supported telephone number. Required for Visa Network Token transactions to allow successful token provisioning.

shippingAddress

{
description: (Optional) Shipping address

address1
string
example: CardHolder House
First line of the shipping address.

address2
string
example: 1 CardHolder Street
Second line of the shipping address.

town
string
example: CardHolder Town
Town of the shipping address.

postCode
string
example: AB1 2CD
Postcode of the shipping address.

countryCode
string
pattern: ^\d{3}$
example: 826
ISO 3166-1 (3 digit) country code of the shipping address.

isBillingAddress
boolean
example: true
Indicator of whether the billing address is the same as the shipping address.

}

mobileNumber

string
pattern: ^\d{0,15}$
example: 7999999999
The cardholder's mobile telephone number. Mastercard recommends providing at least one contact method for Mastercard 3DS-authenticated transactions from 1 July 2026. A contact method can be an email address or supported telephone number. Must be set if phoneCountryCode is set.

phoneCountryCode

string
default: 44
pattern: ^\d{0,3}$
example: 44
The international dialling code associated with the mobile telephone number. Should be set if mobileNumber is set, defaulted to 44 if not.

threeDSecure

{
authenticationSource
string
example: Browser
Type of device that will be used for challenge authentication.

Enum:
[Browser]

methodNotificationUrl
string
default: https://api.judopay.com/order/3ds/methodNotification
example: https://api.judopay.com/order/3ds/methodNotification
URL of the page that will receive a POST event after a device details call is completed. Example is a Judo-hosted page that triggers a call to POST /transactions/{receiptId}/resume3ds (which will be used if this attribute is not set)

challengeNotificationUrl
string
default: https://api.judopay.com/order/3ds/challengeNotification



example:
https://api.judopay.com/order/3ds/challengeNotification
URL of the page that will receive a POST event after a challenge is completed. Example is a Judo-hosted page that triggers a call to POST /transactions/{receiptId}/complete3ds (which will be used if this attribute is not set)

}

threeDSecureMpi {

description: Optional block to pass in 3DS2 authentication results performed outside of Judopay.

dsTransId* string
minLength: 36
maxLength: 64
pattern: ^[0-9a-zA-Z-]$*
example: 41aadf4c-e73f-4bd1-a0c4-acb2615a32af
REQUIRED. Unique identifier for the authentication assigned by the Directory Server (Card Scheme).

cavv* string
minLength: 28
maxLength: 28
pattern: ^[a-zA-Z0-9+/=]$*
example: AJkBCWUyZwAAAABugmKTdAAAAAA=
REQUIRED. Unique value for the authentication provided by the ACS. May be used to provide proof of authentication.

eci* string
minLength: 2
maxLength: 2
pattern: ^\d{2}$
example: 02
REQUIRED. Electronic Commerce Indicator (ECI) is the value returned by Directory Servers (namely Visa, MasterCard, JCB, and American Express) indicating the outcome of authentication attempted on transactions enforced by 3DS.

threeDSecureVersion* string
minLength: 5
example: 2.2.0
REQUIRED. Version of 3DSecure that the card is enrolled into.

}

cardAddress cardAddressRequestAttributes {

description: Card holder address.

address1 string
pattern: ^[^<>\\]+$
example: CardHolder House
The first line of the cardholder's billing address. Mastercard recommends providing this field for Mastercard 3DS-authenticated transactions from 1 July 2026. Supplying the billing address may improve issuer authentication and authorisation decisions. Max 50 characters for a 3DS2 enabled API token.

address2 string
pattern: ^[^<>\\]+$
example: 1 CardHolder Street
Second line of the card holder's address. Max 50 characters for a 3DS2 enabled API token.

town string
pattern: ^[^<>\\]+$
example: CardHolder Town
Town of the card holder's address. Max 40 characters for a 3DS2 enabled API token.

postCode* string
pattern: ^[0-9a-zA-Z -]$*
example: AB1 2CD
Postcode of the card holder's address.

state string
pattern: ^[A-Z]{2}$
example: FL



This attribute can be used by merchants to provide state/province/territory ISO codes (2-char alpha) - specifically for countries where this is required as part of the address (Eg: US, Canada, India, China).

countryCode
string
pattern: ^\d{3}$
example: 826
ISO 3166-1 (3 digit) country code of the card holder's address.

}

primaryAccountDetails {
description: Information about the primary account holder of the transaction. Used for MCC 6012, MCC 6051 and MCC 7299 accounts only. Does not need to be specified if these details were set on a payment session associated with this transaction.

name*
string
maxLength: 100
pattern: ^\D+$
example: Doe
Surname of the recipient as registered with the account.

accountNumber*
string
maxLength: 10
pattern: ^\d{0,10}$
example: 12345678
Up to the first 10 digits of the primary account number of the recipient.

dateOfBirth*
string
pattern: ^\d{4}-\d{2}-\d{2}$
example: 1980-01-31
Date of birth of the recipient in yyyy-mm-dd format. If the recipient is a business, then the value should be set to "0000-00-00"

postCode*
string
maxLength: 10
example: AB1 2CD
Postcode registered against the recipient's account.

}

}

collectionRequest {

receiptId*
string
example: 1001131610340495360
REQUIRED. Judopay's reference for the pre-authorisation transaction that is to be collected.

amount
number($float)
minimum: 0.01
example: 10.99
The amount to collect. Format for default currencies is to two decimal places. For currencies using a different structure please contact Judopay for support. If not set, the outstanding pre-authorised amount will be collected.

yourPaymentMetaData
{
description: Optional key-value map for additional metadata associated with this transaction. Will be stored but not processed or passed to gateways. Do not include sensitive information like card numbers.

}
example: { "internalLocationRef": "Example", "internalId": 99 }

yourPaymentReference*
string
example: aa648425-2230-4e4e-8a61-27f06df54542
REQUIRED. Your unique reference for this collection (not the yourPaymentReference of the original transaction)

}



complete3ds2Request {

cv2 string
minLength: 3
maxLength: 4
pattern: ^\d{3,4}$
example: 123
The 3 or 4 digit number on the back of a credit card. Also known as the card verification value (CVV) or security code. If not specified, contact Judopay support to confirm gateway settings are configured to reflect this.

version* string
example: 2.2.0
REQUIRED. The version of 3DS used for this transaction, returned in the response to the initial request.

primaryAccountDetails {
description: Information about the primary account holder of the transaction. Used for MCC 6012, MCC 6051 and MCC 7299 accounts only. Does not need to be specified if these details were set on a payment session associated with this transaction.

name* string
maxLength: 100
pattern: ^\D+$
example: Doe
Surname of the recipient as registered with the account.

accountNumber* string
maxLength: 10
pattern: ^\d{0,10}$
example: 12345678
Up to the first 10 digits of the primary account number of the recipient.

dateOfBirth* string
pattern: ^\d{4}-\d{2}-\d{2}$
example: 1980-01-31
Date of birth of the recipient in yyyy-mm-dd format. If the recipient is a business, then the value should be set to "0000-00-00"

postCode* string
maxLength: 10
example: AB1 2CD
Postcode registered against the recipient's account.

}
}

consumerResponse {

yourConsumerReference string
example: 2b45fd3f-cee5-4e7e-874f-28051db65408
Your unique reference to anonymously identify your customer.

}



createPaymentSessionResponse {

payByLinkUrl
string
example: https://pay-sandbox.judopay.com/ABC123
Full URL that directs consumer to the Judo hosted WebPayments UI for this session

postUrl
string
example: https://pay-sandbox.judopay.com/v2
Base URL that merchants can use to direct consumers to Judo hosted WebPayments UI

reference
string
example: 5QcAAAQAAAAPAAAACAAAABtGgvhBrF9BHTN7nqn1e0J4hVVmi-y27dGPjWBMtls3Gj_XDg
Judo internal reference for this payment session, used as a parameter on postUrl to direct to WebPayments UI

}

deviceResponse {

description:
[Conditional] If clientDetails was provided containing a kdeviceid

identifier
string
example: d73b4a7b58ce4e54a3bd73b7eda061e6
Judopay generated GUID associated with the kdeviceid passed in the clientDetails request attribute

}

errorResponse {

requestId
string
Identifier for this request, it is helpful to provide this to Judopay developer support when requesting help with issues.

message
string
Description of message, intended for issue diagnosis or display to users. Do not code against this string as it can change without notice.

code
integer
Numeric code associated with specific error, will not change over time so this can be coded against

category
integer
Numeric code associated with category type of error

}



googlePayRequest {
googlePayWallet* {
token* string
example: "{\"signature\":\"MEUCIQD6IX46z4mg8SCoS\/UnbrgeUk+azwYFOHOHltdsO8D\/VQIge3tia
{\\\"encryptedMessage\\\":\\\"Vp5ofRY6BkFnIdM1kNvHdwAfORynkLah8nBXwCPUt5Jt5RFAx+zvpjB4
JSON serialized payment token generated by Google Pay API
cardNetwork string
example: VISA
The payment card network returned by Google Pay API. Value should match one of the allo
cardDetails string
example: 1234
The details about the card returned by Google Pay API. This value is commonly the last
}
cardAddress cardAddressRequestAttributes {
description: Card holder address.
address1 string
pattern: ^[^<>\\]+$
example: CardHolder House
The first line of the cardholder's billing address. Mastercard recommends pro
address2 string
pattern: ^[^<>\\]+$
example: 1 CardHolder Street
Second line of the card holder's address. Max 50 characters for a 3DS2 enable
town string
pattern: ^[^<>\\]+$
example: CardHolder Town
Town of the card holder's address. Max 40 characters for a 3DS2 enabled API t
postCode* string
pattern: ^[0-9a-zA-Z -]*$
example: AB1 2CD
Postcode of the card holder's address.
state string
pattern: ^[A-Z]{2}$
example: FL
This attribute can be used by merchants to provide state/province/territory I
countryCode string
pattern: ^\d{3}$
example: 826
ISO 3166-1 (3 digit) country code of the card holder's address.
}
}



incrementAuthRequest {

receiptId* string

example: 1001131610340495360

REQUIRED. Judopay's reference for the pre-authorisation transaction that is to be incremented.

amount* number($float)

example: 0.99

REQUIRED. The amount to increment (in the same currency as the original pre-auth). Format for default currencies is to two decimal places. For currencies using a different structure please contact Judopay for support.

yourPaymentMetaData {

description: Optional key-value map for additional metadata associated with this transaction. Will be stored but not processed or passed to gateways. Do not include sensitive information like card numbers.

}

example: { "internalLocationRef": "Example", "internalId": 99 }

yourPaymentReference* string

example: aa648425-2230-4e4e-8a61-27f06df54542

REQUIRED. Your unique reference for this transaction (not the yourPaymentReference of the original transaction)

}

infoResponse {

version string

example: 6.23.0.0

Latest supported Api-Version to be used in headers for requests.

}



listPaymentSessionsResponse {

resultCount
number
example: 135
Total number of payment sessions that match the criteria

pageSize
integer
example: 10
Number of payment sessions to return in a page

offset
integer
example: 0
Zero-based index of first payment session in page

results
[
A page of payment session results

paymentSessionHistoricListResponse {

amount
number($float)
example: 10.99
The amount to process. Format for default currencies is to two decimal places. For currencies using a different structure please contact Judopay for support.

cardAddress
{
description: [Conditional] Card holder address, if provided in the session creation

address1
string
pattern: ^[^<>\\]+$
example: CardHolder House
The first line of the cardholder's billing address. Mastercard recommends providing this field for Mastercard 3DS-authenticated transactions from 1 July 2026. Supplying the billing address may improve issuer authentication and authorisation decisions. Max 50 characters for a 3DS2 enabled API token.

address2
string
pattern: ^[^<>\\]+$
example: 1 CardHolder Street
Second line of the card holder's address. Max 50 characters for a 3DS2 enabled API token.

town
string
pattern: ^[^<>\\]+$
example: CardHolder Town
Town of the card holder's address. Max 40 characters for a 3DS2 enabled API token.

postCode*
string
pattern: ^[0-9a-zA-Z -]$*
example: AB1 2CD
Postcode of the card holder's address.

state
string
pattern: ^[A-Z]{2}$
example: FL
This attribute can be used by merchants to provide state/province/territory ISO codes (2-char alpha) - specifically for countries where this is required as part of the address (Eg: US, Canada, India, China).

countryCode
integer
pattern: ^\d{2,3}$
example: 826
ISO 3166-1 (3 digit) country code of the card holder's address. If the first digit of the of the code is 0 this will not be shown

cardHolderName
string
example: John Doe
Name of card holder.



}

clientIpAddress
string
example: 1.2.3.4
[Conditional] IP address of caller, if provided in the session creation

clientUserAgent
string
example: Mozilla/5.0
[Conditional] User agent of caller, if provided in the session creation

companyName
string
example: Test Merchant
Trading name associated with JudoId specified for this session

currency
string
example: GBP
The ISO-4217 alphabetic code for the currency associated with the payment session

expiryDate
string($date-time)
example: 2026-02-05T16:28:32.8596+00:00
Date and time of expiry of payment session

judoId
string
example: 100100100
Unique merchant and/or location ID supplied by Judopay used for this session

paymentCancelUrl
string
example: https://my.web.site/cancel
[Conditional] Merchant page to POST to after unsuccessful hosted web payment operations using this payment session, if defined on the API application associated with the authentication details

paymentSuccessUrl
string
example: https://my.web.site/success
[Conditional] Merchant page to POST to after successful hosted web payment operations using this payment session, if defined on the API application associated with the authentication details

reference
string
example:
5QcAAAIAAAAMAAAADwAAAPwlgcQCLU6sLrNhi8kt1vEoVLwiFJc2EKTqT7JXb2xeQRqyYw
Unique Judo generated reference for this payment session

allowedCardTyes
[
minItems: 0
uniqueItems: true
example: [1, 2, 3, 10, 11, 12]
List of integer codes matching card types defined on https://docs.judopay.com/Content/Developer_Tools/Codes.htm?Highlight=codes#Card

integer]

status
string
example: Open
Current state of the payment session

Enum:
[Open, Success, Expired, Cancelled]

transactionType
string
example: PreAuth
[Conditional] Intended use of payment session, if specified on creation

Enum:
[Payment, PreAuth, CheckCard, null]

yourConsumerReference
string
example: 2b45fd3f-cee5-4e7e-874f-28051db65408
Your unique reference to anonymously identify your customer for this session

yourPaymentMetaData
{
description: [Conditional] Key-value map for additional metadata associated with this transaction, if specified on session creation



```
                                }
                                example: { "internalLocationRef": "Example", "internalId": 99 }
                        yourPaymentReference
                                string
                                example: 6482c678-cad3-4efd-b081-aeae7a89a134
                                Your unique reference for this payment session
                        webPaymentOperation
                                integer
                                example: 0
                                0 for Payment, 1 for Register
                        isPayByLink
                                boolean
                                example: true
                                [Conditional] Only shown if web payment session was created with
                                isPayBLink value set to true
                        isJudoAccept
                                boolean
                                example: false
                                Flag showing whether session was created with isJudoAccept flag (or
                                from POST /judoaccept/payments )
                        tradingName
                                string
                                deprecated: true
                                example: Test Merchant
                                Duplicates companyName (which should be used instead)
                }]
}
```



listTransactionsResponse {

resultCount number
example: 135
Total number of transactions that match the criteria. Note: This is currently capped at 10000 for performance reasons

pageSize integer
example: 10
Number of transactions to return in a page

offset integer
example: 0
Zero-based index of first transaction in page

results [
A page of transaction results

transactionReceiptListResponse {

receiptId string
example: 1001131610340495360
Judopay's reference for the transaction.

originalReceiptId string
example: 1001124307998347264
[Conditional] Judopay's reference for any linked transaction (for collections, refunds and voids)

yourPaymentReference string
example: 6482c678-cad3-4efd-b081-aeae7a89a134
Your unique reference for this payment.

type string
example: PreAuth
The type of the transaction.

Enum:
[Payment, Refund, PreAuth, VOID, Collection, Save, Register, CheckCard]

createdAt string($date-time)
example: 2025-02-05T16:28:32.8596+00:00
Date and time of creation.

result string
example: Success
Result of transaction.

message string
example: AuthCode: 123456
Message detailing the outcome.

judoId integer
example: 100100100
Unique merchant and/or location ID supplied by Judopay.

appearsOnStatementAs string
example: APL/TestMerchant*
Merchant description as it appears on the consumer's statement.

originalAmount string
example: 10.99
Amount of original transaction (not affected by refunds or collections).

amountCollected string
example: 1.99
[Conditional - PreAuth and Register only] Amount that has been collected on original pre-authorisation.

netAmount string
example: 1.00
For payments and refunds, this is the original sale amount mines the total of all refunds. For preauths, incremental auths and collections, this is the original preauth amount plus the total of all incremental auths minus the total of all collections. For voids, this is equal to the original preauth amount.



amount string
example: 1.00
The amount of this transaction, for refunds and collections the amount that has been refunded or collected in this transaction.

currency string
example: GBP
The ISO-217 alphabetic code of the currency of the transaction

postCodeCheckResult string
example: Passed
Show results of post code checks

walletType integer
example: 1
[Conditional] For digital wallet transactions, the type of wallet (1=ApplePay, 4=GooglePay, 5=ClickToPay)

allowIncrement boolean
example: true
[Conditional] true for customer initiated pre-auths that had the allowIncrement request attribute set to true

isIncrementalAuth boolean
example: true
[Conditional] true for calls to POST /transactions/incrementalAuth

cardDetails cardDetailsListResponse {
- **cardLastfour** string
  example: 1234
  Last four digits of card
- **endDate** string
  example: 1225
  Card expiry date in format MMYY
- **cardType** integer
  example: 2
  Integer representing the type of card, see Reference Codes - Card Types on https://docs.judopay.com/Content/Developer%20Tools/Codes.htm#CardTypes

}

consumer consumerResponse {
- **yourConsumerReference** string
  example: 2b45fd3f-cee5-4e7e-874f-28051db65408
  Your unique reference to anonymously identify your customer.

}

yourPaymentMetaData {
- **description:** Merchant metadata passed with transaction request.

}
example: { "internalLocationRef": "Example", "internalId": 99 }

threeDSecure threeDSecureCompletedResponse {
- **description:** [Conditional] Only returned for transactions using 3DS
- **attempted** boolean
  example: true
  Did the consumer attempt to authenticate through 3DSecure
- **result** string
  example: PASSED
  [Conditional] Outcome of the 3DSecure authentication, only present on initial receipt response, not historic receipts
- **eci** string
  example: 05
  Electronic Commerce Indicator returned by 3DS servers indicating outcome of authentication



challengeRequestIndicator string
nullable: true
example: ChallengeMandated
[Conditional] Challenge request indicator that was used for 3DS authentication

Enum:

[NoPreference, NoChallenge, ChallengePreferred, ChallengeAsMandate, ChallengeWithWhitelistPrompt, null]

scaExemption string
nullable: true
example:
[Conditional] SCA exemption that was used for 3DS authentication

Enum:

[TrustedBeneficiary, TransactionRiskAnalysis, DataShareOnly, ScaAlreadyPerformed, null]

}

disableNetworkTokenisation boolean
example: true
[Conditional] true if the request explicitly disabled network tokens for this transaction

networkTokenisationDetails networkTokenisationDetailsResponse {

networkTokenProvisioned boolean
example: true
Indicates if a new network token was created during the course of this transaction

networkTokenUsed boolean
example: true
Indicates if this transaction was completed using a network token

virtualPan {

lastFour string
example: 1234
Last four digits of the virtual PAN used

expiryDate string
example: 0129
Expiry date of the virtual PAN in format MMYY

}

}

}]

sort string
example: time-descending
Sort order of the transactions as specified in the request

}



Apple Pay {
pkPayment* {
token* {
paymentMethod {
displayName string
example: Visa 1234
Card description suitable for display,
network string
example: Visa
Name of the payment network backing the
type string
example: debit
Card's type of payment, provided by the
}
paymentData* {
data* string
example: CXnUZAtPGWfkLGsHksn5WTzfFhEzNMqSJZXKTmbaW
Encrypted payment data.
signature* string
example:
gbtqw7Zy9LcfsV53pSRxDJjwUKmrLAWkxVxL6NQzzChwGRMQ3d
Signature of the payment and header data. The signa
header {
description: Additional version-depen
publicKeyHash string
example: EMq6ggcTqGNYn86
ephemeralPublicKey string
example: Y5q3gWSJL8byWFH
transactionId string
example: 7c3b36d16bb0f6e
}
version string
example: EC_V1
Version information about the payment token. The t
}
}
}
cardAddress cardAddressRequestAttributes {
description: Card holder address.
address1 string
pattern: ^[^<>\\]+$
example: CardHolder House
The first line of the cardholder's billing address. Mastercard recomme
address2 string
pattern: ^[^<>\\]+$
example: 1 CardHolder Street
Second line of the card holder's address. Max 50 characters for a 3DS2
town string
pattern: ^[^<>\\]+$
example: CardHolder Town
Town of the card holder's address. Max 40 characters for a 3DS2 enable
postCode* string
pattern: ^[0-9a-zA-Z -]*$
example: AB1 2CD
Postcode of the card holder's address.
state string
pattern: ^[A-Z]{2}$
example: FL



This attribute can be used by merchants to provide state/province/terr

countryCode
string
pattern: ^\d{3}$
example: 826
ISO 3166-1 (3 digit) country code of the card holder's address.

}

judoId*
string
maxLength: 9
pattern: ^\d{9}$
example: 100100100
REQUIRED. Unique merchant and/or location ID supplied by Judopay. Do not include spaces or dashe

yourConsumerReference*
string
minLength: 1
maxLength: 50
example: 2b45fd3f-cee5-4e7e-874f-28051db65408
REQUIRED. Unique reference to anonymously identify your customer. Advisable to use GUIDs.

yourPaymentReference*
string
minLength: 1
maxLength: 50
example: 6482c678-cad3-4efd-b081-aeae7a89a134
REQUIRED. Your unique reference for this payment. This value should be unique in order to protec

yourPaymentMetaData
{
description: Optional key-value map for additional metadata associated with this tr
}
example: { "internalLocationRef": "Example", "internalId": 99 }

currency
string
example: GBP
The ISO-4217 alphabetic code for the currency to use for the transaction, only required if diffe

amount*
number($float)
minimum: 0.01
example: 10.99
REQUIRED. The amount to process. Format for default currencies is to two decimal places. For cur

webPaymentReference
string
example: 5QcAAAIAAAAMAAAADwAAAPwlgcQCLU6sLrNhi8kt1vEoVLwiFJc2EKTqT7JXb2xeQRqyYw
The unique reference of the Judopay payment session associated with this transaction.

clientDetails
{
description: A set of encrypted fraud signals sent by the mobile SDK
key
string
value
string
}

initialRecurringPayment
boolean
example: false
Indicates if this is the first transaction that forms part of a recurring payment schedule. This

primaryAccountDetails
{
description: Information about the primary account holder of the transaction. Used
name*
string
maxLength: 100
pattern: ^\D+$
example: Doe
Surname of the recipient as registered with the account.

accountNumber*
string
maxLength: 10
pattern: ^\d{0,10}$
example: 12345678
Up to the first 10 digits of the primary account number of the recipie

dateOfBirth*
string
pattern: ^\d{4}-\d{2}-\d{2}$
example: 1980-01-31
Date of birth of the recipient in yyyy-mm-dd format. If the recipient

postCode*
string



```
                                                    maxLength: 10
                                                    example: AB1 2CD
                                                    Postcode registered against the recipient's account.
                                }
}
```



Card token {

cardToken*
string
example: Cw8eZ7nLBM2rFXx12WiRI0YveZPinkPm
REQUIRED. A randomly generated string linked to a card saved securely within the Judopay Card Vault.

cv2
string
minLength: 3
maxLength: 4
pattern: ^\d{3,4}$
example: 123
The 3 or 4 digit number on the back of a credit card. Also known as the card verification value (CVV) or security code.

cardAddress
cardAddressRequestAttributes {
description: Card holder address.

address1
string
pattern: ^[^<>\\]+$
example: CardHolder House
The first line of the cardholder's billing address. Mastercard recommends providing this field for Mastercard 3DS-authenticated transactions from 1 July 2026. Supplying the billing address may improve issuer authentication and authorisation decisions. Max 50 characters for a 3DS2 enabled API token.

address2
string
pattern: ^[^<>\\]+$
example: 1 CardHolder Street
Second line of the card holder's address. Max 50 characters for a 3DS2 enabled API token.

town
string
pattern: ^[^<>\\]+$
example: CardHolder Town
Town of the card holder's address. Max 40 characters for a 3DS2 enabled API token.

postCode*
string
pattern: ^[0-9a-zA-Z -]$*
example: AB1 2CD
Postcode of the card holder's address.

state
string
pattern: ^[A-Z]{2}$
example: FL
This attribute can be used by merchants to provide state/province/territory ISO codes (2-char alpha) - specifically for countries where this is required as part of the address (Eg: US, Canada, India, China).

countryCode
string
pattern: ^\d{3}$
example: 826
ISO 3166-1 (3 digit) country code of the card holder's address.

}

judoId*
string
maxLength: 9
pattern: ^\d{9}$
example: 100100100
REQUIRED. Unique merchant and/or location ID supplied by Judopay. Do not include spaces or dashes. If not specified, then your default judoId will be used.

yourConsumerReference*
string
minLength: 1
maxLength: 50
example: 2b45fd3f-cee5-4e7e-874f-28051db65408
REQUIRED. Unique reference to anonymously identify your customer. Advisable to use GUIDs.

yourPaymentReference*
string
minLength: 1
maxLength: 50



example: 6482c678-cad3-4efd-b081-aeae7a89a134
REQUIRED. Your unique reference for this payment. This value should be unique in order to protect your customers against duplicate transactions. With a server side integration, if a payment reference is not supplied, the transaction will not be processed.

yourPaymentMetaData

{
description: Optional key-value map for additional metadata associated with this transaction. Will be stored but not processed or passed to gateways. Do not include sensitive information like card numbers.
}
example: { "internalLocationRef": "Example", "internalId": 99 }

currency

string
example: GBP
The ISO-4217 alphabetic code for the currency to use for the transaction, only required if different from the currency set during onboarding (default is GBP)

amount*

number($float)
minimum: 0.01
example: 10.99
REQUIRED. The amount to process. Format for default currencies is to two decimal places. For currencies using a different structure please contact Judopay for support.

webPaymentReference

string
example: 5QcAAAIAAAAMAAAADwAAAPwlgcQCLU6sLrNhi8kt1vEoVLwiFJc2EKTqT7JXb2xeQRqyYw
The unique reference of the Judopay payment session associated with this transaction.

clientDetails

{
description: A set of encrypted fraud signals sent by the mobile SDK
key **string**
value **string**
}

initialRecurringPayment

boolean
example: false
Indicates if this is the first transaction that forms part of a recurring payment schedule. This flag should be set to 'true' if this is a customer-initiated payment which will be followed by a series of scheduled recurring payments.

primaryAccountDetails

{
description: Information about the primary account holder of the transaction. Used for MCC 6012, MCC 6051 and MCC 7299 accounts only. Does not need to be specified if these details were set on a payment session associated with this transaction.

name* **string**
maxLength: 100
pattern: ^\D+$
example: Doe
Surname of the recipient as registered with the account.

accountNumber* **string**
maxLength: 10
pattern: ^\d{0,10}$
example: 12345678
Up to the first 10 digits of the primary account number of the recipient.

dateOfBirth* **string**
pattern: ^\d{4}-\d{2}-\d{2}$
example: 1980-01-31
Date of birth of the recipient in yyyy-mm-dd format. If the recipient is a business, then the value should be set to "0000-00-00"

postCode* **string**
maxLength: 10
example: AB1 2CD
Postcode registered against the recipient's account.

}

acceptHeaders

string
default: text/html
example: text/html
Passed to ACS for use in rendering challenge screens.



userAgent

string

example: Mozilla/5.0 (iPhone; CPU iPhone OS 14_4 like Mac OS X) AppleWebKit/605.1.15 (KHTML, like Gecko) Mobile/15E148

Passed to ACS for use in rendering challenge screens.

cardHolderName

string

pattern: ^.{0,45}$

example: John Doe

The cardholder's full name. Mastercard recommends providing this field for Mastercard 3DS-authenticated transactions from 1 July 2026. Supplying the cardholder name may improve issuer authentication and authorisation decisions. Note: When testing in the sandbox environment, it is the cardHolderName that is used to determine the test card for 3D Secure 2 authentication.

emailAddress

string

pattern: ^.{0,254}$

example: test.user@judopay.com

The cardholder's email address. Mastercard recommends providing at least one contact method for Mastercard 3DS-authenticated transactions from 1 July 2026. A contact method can be an email address or supported telephone number. Required for Visa Network Token transactions to allow successful token provisioning.

shippingAddress

{

description: (Optional) Shipping address

address1

string

example: CardHolder House

First line of the shipping address.

address2

string

example: 1 CardHolder Street

Second line of the shipping address.

town

string

example: CardHolder Town

Town of the shipping address.

postCode

string

example: AB1 2CD

Postcode of the shipping address.

countryCode

string

pattern: ^\d{3}$

example: 826

ISO 3166-1 (3 digit) country code of the shipping address.

isBillingAddress

boolean

example: true

Indicator of whether the billing address is the same as the shipping address.

}

mobileNumber

string

pattern: ^\d{0,15}$

example: 7999999999

The cardholder's mobile telephone number. Mastercard recommends providing at least one contact method for Mastercard 3DS-authenticated transactions from 1 July 2026. A contact method can be an email address or supported telephone number. Must be set if phoneCountryCode is set.

phoneCountryCode

string

default: 44

pattern: ^\d{0,3}$

example: 44

The international dialling code associated with the mobile telephone number. Should be set if mobileNumber is set, defaulted to 44 if not.

threeDSecure

{

authenticationSource

string

example: Browser

Type of device that will be used for challenge authentication.

Enum:

[Browser]

methodNotificationUrl

string



default:
https://api.judopay.com/order/3ds/methodNotification
example:
https://api.judopay.com/order/3ds/methodNotification
URL of the page that will receive a POST event after a device details call is completed. Example is a Judo-hosted page that triggers a call to POST /transactions/{receiptId}/resume3ds (which will be used if this attribute is not set)

challengeNotificationUrl ⌄ **string**
default:
https://api.judopay.com/order/3ds/challengeNotificatio
example:
https://api.judopay.com/order/3ds/challengeNotificatio
URL of the page that will receive a POST event after a challenge is completed. Example is a Judo-hosted page that triggers a call to POST /transactions/{receiptId}/complete3ds (which will be used if this attribute is not set)

challengeRequestIndicator **challengeRequestIndicatorRequest** ⌄
challengeRequestIndicatorRequest **string**
nullable: true
default: noPreference
example: challengePreferred
Indicates whether a challenge is requested for this transaction (this may be over-ruled by the issuer). Should not be specified in the same request as scaExemption

Enum:

⌄ **[noPreference, noChallenge, challengePreferred, challengeAsMandate, challengeWithWhitelistPrompt, null]**

scaExemption **scaExemptionRequest** ⌄ **scaExemptionRequest** **string**
nullable: true
example: transactionRiskAnalysis
Indicates reason why challenge may not be necessary (this may be over-ruled by the issuer). Should not be specified in the same request as challengeRequestIndicator

Enum:

⌄ **[trustedBeneficiary, transactionRiskAnalysis, dataShareOnly, scaAlreadyPerformed, null]**

}

threeDSecureMpi ⌄ **{**

description: Optional block to pass in 3DS2 authentication results performed outside of Judopay.

dsTransId* ⌄ **string**
minLength: 36
maxLength: 64
pattern: ^[0-9a-zA-Z-]$*
example: 41aadf4c-e73f-4bd1-a0c4-acb2615a32af
REQUIRED. Unique identifier for the authentication assigned by the Directory Server (Card Scheme).

cavv* ⌄ **string**
minLength: 28
maxLength: 28
pattern: ^[a-zA-Z0-9+/=]$*
example: AJkBCWUyZwAAAABugmKTdAAAAAA=
REQUIRED. Unique value for the authentication provided by the ACS. May be used to provide proof of authentication.

eci* ⌄ **string**
minLength: 2
maxLength: 2
pattern: ^\d{2}$
example: 02
REQUIRED. Electronic Commerce Indicator (ECI) is the value returned by Directory Servers (namely Visa, MasterCard, JCB, and American Express) indicating the outcome of authentication attempted on transactions enforced by 3DS.

threeDSecureVersion* ⌄ **string**



minLength: 5

example: 2.2.0

REQUIRED. Version of 3DSecure that the card is enrolled into.

}

recurringPayment

boolean

example: true

Indicates this transaction is not being initiated by the customer. This should be set to 'true' if the merchant is trying to initiate a scheduled (recurring) or unscheduled (MIT) transaction in the absence of the customer.

recurringPaymentType

string

nullable: true

example: MIT

Indicates the type of recurring payment that the merchant is attempting to process. This can be set to 'RECURRING' (for scheduled payments) or 'MIT' (for unscheduled payments)

Enum:

[RECURRING, MIT, null]

relatedReceiptId

string

example: 1001131610340495360

Receipt identifier of the initial payment/pre-auth transaction that was customer initiated. If recurringPayment is true and recurringPaymentType is set, setting this to the Judo receipt identifier of the initial transaction in the sequence will check that original transaction was successful and on the same account, and pass the gateway transaction identifier of the original transaction on to the gateway with this transaction

relatedPaymentNetworkTransactionId

string

maxLength: 25

example: MNKWQVNQS0915

Card scheme identifier of the initial payment/pre-auth transaction that was customer initiated. If recurringPayment is true and recurringPaymentType is set, setting this to the card scheme identifier of the initial transaction in the sequence will pass this on on to the gateway with this transaction (no checks are made on this identifier). Should not be specified if relatedReceiptId is specified. Only supported by certain processors. Please check with Developer Support if you intend to use this field

disableNetworkTokenisation

boolean

default: false

example: true

(Optional) Set to true to specify that network tokenisation should not be used for this transaction, even if network token registration has been enabled on the account.

}



Google Pay {

googlePayWallet* {

token* string
example: "{\"signature\":\"MEUCIQD6IX46z4mg8SCoS\/UnbrgeUk+azwYFOHOHltdsO8D\/VQ
{\\\"encryptedMessage\\\":\\\"Vp5ofRY6BkFnIdM1kNvHdwAfORynkLah8nBXwCPUt5Jt5RFAx
JSON serialized payment token generated by Google Pay API

cardNetwork string
example: VISA
The payment card network returned by Google Pay API. Value should match one of t

cardDetails string
example: 1234
The details about the card returned by Google Pay API. This value is commonly th

}

cardAddress **cardAddressRequestAttributes** {

description: Card holder address.

address1 string
pattern: ^[^<>\\]+$
example: CardHolder House
The first line of the cardholder's billing address. Mastercard recomme

address2 string
pattern: ^[^<>\\]+$
example: 1 CardHolder Street
Second line of the card holder's address. Max 50 characters for a 3DS2

town string
pattern: ^[^<>\\]+$
example: CardHolder Town
Town of the card holder's address. Max 40 characters for a 3DS2 enable

postCode* string
pattern: ^[0-9a-zA-Z -]$*
example: AB1 2CD
Postcode of the card holder's address.

state string
pattern: ^[A-Z]{2}$
example: FL
This attribute can be used by merchants to provide state/province/terr

countryCode string
pattern: ^\d{3}$
example: 826
ISO 3166-1 (3 digit) country code of the card holder's address.

}

judoId* string
maxLength: 9
pattern: ^\d{9}$
example: 100100100
REQUIRED. Unique merchant and/or location ID supplied by Judopay. Do not include spaces or dashe

yourConsumerReference* string
minLength: 1
maxLength: 50
example: 2b45fd3f-cee5-4e7e-874f-28051db65408
REQUIRED. Unique reference to anonymously identify your customer. Advisable to use GUIDs.

yourPaymentReference* string
minLength: 1
maxLength: 50
example: 6482c678-cad3-4efd-b081-aeae7a89a134
REQUIRED. Your unique reference for this payment. This value should be unique in order to protec

yourPaymentMetaData {

description: Optional key-value map for additional metadata associated with this tr

}
example: { "internalLocationRef": "Example", "internalId": 99 }



currency
string
example: GBP
The ISO-4217 alphabetic code for the currency to use for the transaction, only required if diffe

amount*
number($float)
minimum: 0.01
example: 10.99
REQUIRED. The amount to process. Format for default currencies is to two decimal places. For cur

webPaymentReference
string
example: 5QcAAAIAAAAMAAAADwAAAPwlgcQCLU6sLrNhi8kt1vEoVLwiFJc2EKTqT7JXb2xeQRqyYw
The unique reference of the Judopay payment session associated with this transaction.

clientDetails
{
description: A set of encrypted fraud signals sent by the mobile SDK
key: string
value: string
}

initialRecurringPayment
boolean
example: false
Indicates if this is the first transaction that forms part of a recurring payment schedule. This

primaryAccountDetails
{
description: Information about the primary account holder of the transaction. Used
name*: string
maxLength: 100
pattern: ^\D+$
example: Doe
Surname of the recipient as registered with the account.
accountNumber*: string
maxLength: 10
pattern: ^\d{0,10}$
example: 12345678
Up to the first 10 digits of the primary account number of the recipie
dateOfBirth*: string
pattern: ^\d{4}-\d{2}-\d{2}$
example: 1980-01-31
Date of birth of the recipient in yyyy-mm-dd format. If the recipient
postCode*: string
maxLength: 10
example: AB1 2CD
Postcode registered against the recipient's account.
}

acceptHeaders
string
default: text/html
example: text/html
Passed to ACS for use in rendering challenge screens.

userAgent
string
example: Mozilla/5.0 (iPhone; CPU iPhone OS 14_4 like Mac OS X) AppleWebKit/605.1.15 (KHTML, lik
Passed to ACS for use in rendering challenge screens.

cardHolderName
string
pattern: ^.{0,45}$
example: John Doe
The cardholder's full name. Mastercard recommends providing this field for Mastercard 3DS-authen

emailAddress
string
pattern: ^.{0,254}$
example: test.user@judopay.com
The cardholder's email address. Mastercard recommends providing at least one contact method for

shippingAddress
{
description: (Optional) Shipping address
address1: string
example: CardHolder House
First line of the shipping address.



address2 — string
example: 1 CardHolder Street
Second line of the shipping address.

town — string
example: CardHolder Town
Town of the shipping address.

postCode — string
example: AB1 2CD
Postcode of the shipping address.

countryCode — string
pattern: ^\d{3}$
example: 826
ISO 3166-1 (3 digit) country code of the shipping address.

isBillingAddress — boolean
example: true
Indicator of whether the billing address is the same as the shipping a

}

mobileNumber — string
pattern: ^\d{0,15}$
example: 7999999999
The cardholder's mobile telephone number. Mastercard recommends providing at least one contact m

phoneCountryCode — string
default: 44
pattern: ^\d{0,3}$
example: 44
The international dialling code associated with the mobile telephone number. Should be set if mo

threeDSecure — {

authenticationSource — string
example: Browser
Type of device that will be used for challenge authentication.

Enum:
[Browser]

methodNotificationUrl — string
default: https://api.judopay.com/order/3ds/methodNotification
example: https://api.judopay.com/order/3ds/methodNotification
URL of the page that will receive a POST event after a device deta

challengeNotificationUrl — string
default: https://api.judopay.com/order/3ds/challengeNotification
example: https://api.judopay.com/order/3ds/challengeNotification
URL of the page that will receive a POST event after a challenge i

challengeRequestIndicator — challengeRequestIndicatorRequest challengeRequestIndicator
nullable: true
default: noPreference
example: challengePreferred
Indicates whether a challenge is requested for this transaction (t

Enum:
[noPreference, noChallenge, challengePreferred, challengeAsMa

scaExemption — scaExemptionRequest scaExemptionRequest string
nullable: true
example: transactionRiskAnalysis
Indicates reason why challenge may not be necessary (this may be c

Enum:
[trustedBeneficiary, transactionRiskAnalysis, dataShareOnly,

}

threeDSecureMpi — {

description: Optional block to pass in 3DS2 authentication results performed outsic

dsTransId* — string
minLength: 36
maxLength: 64
pattern: ^[0-9a-zA-Z-]$*



```
                                    example: 41aadf4c-e73f-4bd1-a0c4-acb2615a32af
                                    REQUIRED. Unique identifier for the authentication assigned by the Dir
                cavv*               string
                                    minLength: 28
                                    maxLength: 28
                                    pattern: ^[a-zA-Z0-9+/=]*$
                                    example: AJkBCWUyZwAAAABugmKTdAAAAAA=
                                    REQUIRED. Unique value for the authentication provided by the ACS. May
                eci*                string
                                    minLength: 2
                                    maxLength: 2
                                    pattern: ^\d{2}$
                                    example: 02
                                    REQUIRED. Electronic Commerce Indicator (ECI) is the value returned by
                threeDSecureVersion* string
                                    minLength: 5
                                    example: 2.2.0
                                    REQUIRED. Version of 3DSecure that the card is enrolled into.
            }
}
```



Card PAN {

cardNumber*
string
example: 1234567890123456
REQUIRED. The unique number printed on a credit card (13 to 19 digits depending on card type). Submitted without whitespace or non-numeric characters.

cv2
string
minLength: 3
maxLength: 4
pattern: ^\d{3,4}$
example: 123
The 3 or 4 digit number on the back of a credit card. Also known as the card verification value (CVV) or security code. If not specified, contact Judopay support to confirm gateway settings are configured to reflect this.

expiryDate*
string
pattern: ^\d{2}(\/)?\d{2}$
example: 01/28
REQUIRED. The expiry date of the card in format MM/YY

startDate
string
pattern: ^\d{2}(\/)?\d{2}$
example: 07/20
For Maestro cards only, start date in format MM/YY

cardAddress
cardAddressRequestAttributes {
description: Card holder address.

address1
string
pattern: ^[^<>\\]+$
example: CardHolder House
The first line of the cardholder's billing address. Mastercard recommends providing this field for Mastercard 3DS-authenticated transactions from 1 July 2026. Supplying the billing address may improve issuer authentication and authorisation decisions. Max 50 characters for a 3DS2 enabled API token.

address2
string
pattern: ^[^<>\\]+$
example: 1 CardHolder Street
Second line of the card holder's address. Max 50 characters for a 3DS2 enabled API token.

town
string
pattern: ^[^<>\\]+$
example: CardHolder Town
Town of the card holder's address. Max 40 characters for a 3DS2 enabled API token.

postCode*
string
pattern: ^[0-9a-zA-Z -]$*
example: AB1 2CD
Postcode of the card holder's address.

state
string
pattern: ^[A-Z]{2}$
example: FL
This attribute can be used by merchants to provide state/province/territory ISO codes (2-char alpha) - specifically for countries where this is required as part of the address (Eg: US, Canada, India, China).

countryCode
string
pattern: ^\d{3}$
example: 826
ISO 3166-1 (3 digit) country code of the card holder's address.

}

judoId*
string
maxLength: 9
pattern: ^\d{9}$
example: 100100100
REQUIRED. Unique merchant and/or location ID supplied by Judopay. Do not include spaces or dashes. If not specified, then your default judoId will be used.



yourConsumerReference*

string
minLength: 1
maxLength: 50
example: 2b45fd3f-cee5-4e7e-874f-28051db65408
REQUIRED. Unique reference to anonymously identify your customer. Advisable to use GUIDs.

yourPaymentReference*

string
minLength: 1
maxLength: 50
example: 6482c678-cad3-4efd-b081-aeae7a89a134
REQUIRED. Your unique reference for this payment. This value should be unique in order to protect your customers against duplicate transactions. With a server side integration, if a payment reference is not supplied, the transaction will not be processed.

yourPaymentMetaData

{
description: Optional key-value map for additional metadata associated with this transaction. Will be stored but not processed or passed to gateways. Do not include sensitive information like card numbers.
}
example: { "internalLocationRef": "Example", "internalId": 99 }

currency

string
example: GBP
The ISO-4217 alphabetic code for the currency to use for the transaction, only required if different from the currency set during onboarding (default is GBP)

amount*

number($float)
minimum: 0.01
example: 10.99
REQUIRED. The amount to process. Format for default currencies is to two decimal places. For currencies using a different structure please contact Judopay for support.

webPaymentReference

string
example: 5QcAAAIAAAAMAAAADwAAAPwlgcQCLU6sLrNhi8kt1vEoVLwiFJc2EKTqT7JXb2xeQRqyYw
The unique reference of the Judopay payment session associated with this transaction.

clientDetails

{
description: A set of encrypted fraud signals sent by the mobile SDK
key: string
value: string
}

initialRecurringPayment

boolean
example: false
Indicates if this is the first transaction that forms part of a recurring payment schedule. This flag should be set to 'true' if this is a customer-initiated payment which will be followed by a series of scheduled recurring payments.

primaryAccountDetails

{
description: Information about the primary account holder of the transaction. Used for MCC 6012, MCC 6051 and MCC 7299 accounts only. Does not need to be specified if these details were set on a payment session associated with this transaction.

name*

string
maxLength: 100
pattern: ^\D+$
example: Doe
Surname of the recipient as registered with the account.

accountNumber*

string
maxLength: 10
pattern: ^\d{0,10}$
example: 12345678
Up to the first 10 digits of the primary account number of the recipient.

dateOfBirth*

string
pattern: ^\d{4}-\d{2}-\d{2}$
example: 1980-01-31
Date of birth of the recipient in yyyy-mm-dd format. If the recipient is a business, then the value should be set to "0000-00-00"



postCode* string
maxLength: 10
example: AB1 2CD
Postcode registered against the recipient's account.

}

acceptHeaders string
default: text/html
example: text/html
Passed to ACS for use in rendering challenge screens.

userAgent string
example: Mozilla/5.0 (iPhone; CPU iPhone OS 14_4 like Mac OS X) AppleWebKit/605.1.15 (KHTML, like Gecko) Mobile/15E148
Passed to ACS for use in rendering challenge screens.

cardHolderName string
pattern: ^.{0,45}$
example: John Doe
The cardholder's full name. Mastercard recommends providing this field for Mastercard 3DS-authenticated transactions from 1 July 2026. Supplying the cardholder name may improve issuer authentication and authorisation decisions. Note: When testing in the sandbox environment, it is the cardHolderName that is used to determine the test card for 3D Secure 2 authentication.

emailAddress string
pattern: ^.{0,254}$
example: test.user@judopay.com
The cardholder's email address. Mastercard recommends providing at least one contact method for Mastercard 3DS-authenticated transactions from 1 July 2026. A contact method can be an email address or supported telephone number. Required for Visa Network Token transactions to allow successful token provisioning.

shippingAddress {
description: (Optional) Shipping address

address1 string
example: CardHolder House
First line of the shipping address.

address2 string
example: 1 CardHolder Street
Second line of the shipping address.

town string
example: CardHolder Town
Town of the shipping address.

postCode string
example: AB1 2CD
Postcode of the shipping address.

countryCode string
pattern: ^\d{3}$
example: 826
ISO 3166-1 (3 digit) country code of the shipping address.

isBillingAddress boolean
example: true
Indicator of whether the billing address is the same as the shipping address.

}

mobileNumber string
pattern: ^\d{0,15}$
example: 7999999999
The cardholder's mobile telephone number. Mastercard recommends providing at least one contact method for Mastercard 3DS-authenticated transactions from 1 July 2026. A contact method can be an email address or supported telephone number. Must be set if phoneCountryCode is set.

phoneCountryCode string
default: 44
pattern: ^\d{0,3}$
example: 44
The international dialling code associated with the mobile telephone number. Should be set if mobileNumber is set, defaulted to 44 if not.



threeDSecure { }

- **authenticationSource** — string
  example: Browser
  Type of device that will be used for challenge authentication.

  Enum: [Browser]
- **methodNotificationUrl** — string
  default: https://api.judopay.com/order/3ds/methodNotification
  example: https://api.judopay.com/order/3ds/methodNotification
  URL of the page that will receive a POST event after a device details call is completed. Example is a Judo-hosted page that triggers a call to POST /transactions/{receiptId}/resume3ds (which will be used if this attribute is not set)
- **challengeNotificationUrl** — string
  default: https://api.judopay.com/order/3ds/challengeNotificatio
  example: https://api.judopay.com/order/3ds/challengeNotificatio
  URL of the page that will receive a POST event after a challenge is completed. Example is a Judo-hosted page that triggers a call to POST /transactions/{receiptId}/complete3ds (which will be used if this attribute is not set)
- **challengeRequestIndicator** — challengeRequestIndicatorRequest
  challengeRequestIndicatorRequest string
  nullable: true
  default: noPreference
  example: challengePreferred
  Indicates whether a challenge is requested for this transaction (this may be over-ruled by the issuer). Should not be specified in the same request as scaExemption

  Enum: [noPreference, noChallenge, challengePreferred, challengeAsMandate, challengeWithWhitelistPrompt, null]
- **scaExemption** — scaExemptionRequest scaExemptionRequest string
  nullable: true
  example: transactionRiskAnalysis
  Indicates reason why challenge may not be necessary (this may be over-ruled by the issuer). Should not be specified in the same request as challengeRequestIndicator

  Enum: [trustedBeneficiary, transactionRiskAnalysis, dataShareOnly, scaAlreadyPerformed, null]

}

threeDSecureMpi { }

- **description:** Optional block to pass in 3DS2 authentication results performed outside of Judopay.
- **dsTransId*** — string
  minLength: 36
  maxLength: 64
  pattern: ^[0-9a-zA-Z-]$*
  example: 41aadf4c-e73f-4bd1-a0c4-acb2615a32af
  REQUIRED. Unique identifier for the authentication assigned by the Directory Server (Card Scheme).
- **cavv*** — string
  minLength: 28
  maxLength: 28
  pattern: ^[a-zA-Z0-9+/=]$*
  example: AJkBCWUyZwAAAABugmKTdAAAAAA=
  REQUIRED. Unique value for the authentication provided by the ACS. May be used to provide proof of authentication.



eci* string
minLength: 2
maxLength: 2
pattern: ^\d{2}$
example: 02
REQUIRED. Electronic Commerce Indicator (ECI) is the value returned by Directory Servers (namely Visa, MasterCard, JCB, and American Express) indicating the outcome of authentication attempted on transactions enforced by 3DS.

threeDSecureVersion* string
minLength: 5
example: 2.2.0
REQUIRED. Version of 3DSecure that the card is enrolled into.

}

recurringPayment boolean
example: true
Indicates this transaction is not being initiated by the customer. This should be set to 'true' if the merchant is trying to initiate a scheduled (recurring) or unscheduled (MIT) transaction in the absence of the customer.

recurringPaymentType string
nullable: true
example: MIT
Indicates the type of recurring payment that the merchant is attempting to process. This can be set to 'RECURRING' (for scheduled payments) or 'MIT' (for unscheduled payments)

Enum:
[RECURRING, MIT, null]

relatedReceiptId string
example: 1001131610340495360
Receipt identifier of the initial payment/pre-auth transaction that was customer initiated. If recurringPayment is true and recurringPaymentType is set, setting this to the Judo receipt identifier of the initial transaction in the sequence will check that original transaction was successful and on the same account, and pass the gateway transaction identifier of the original transaction on to the gateway with this transaction

relatedPaymentNetworkTransactionId string
maxLength: 25
example: MNKWQVNQS0915
Card scheme identifier of the initial payment/pre-auth transaction that was customer initiated. If recurringPayment is true and recurringPaymentType is set, setting this to the card scheme identifier of the initial transaction in the sequence will pass this on on to the gateway with this transaction (no checks are made on this identifier). Should not be specified if relatedReceiptId is specified. Only supported by certain processors. Please check with Developer Support if you intend to use this field

disableNetworkTokenisation boolean
default: false
example: true
(Optional) Set to true to specify that network tokenisation should not be used for this transaction, even if network token registration has been enabled on the account.

}



paymentSessionHistoricResponse {

amount number($float)
example: 10.99
The amount to process. Format for default currencies is to two decimal places. For currencies usin
different structure please contact Judopay for support.

cardAddress {
description: [Conditional] Card holder address, if provided in the session creation

address1 string
pattern: ^[^<>\\]+$
example: CardHolder House
The first line of the cardholder's billing address. Mastercard recommend
providing this field for Mastercard 3DS-authenticated transactions from
2026. Supplying the billing address may improve issuer authentication an
authorisation decisions. Max 50 characters for a 3DS2 enabled API token.

address2 string
pattern: ^[^<>\\]+$
example: 1 CardHolder Street
Second line of the card holder's address. Max 50 characters for a 3DS2 e
API token.

town string
pattern: ^[^<>\\]+$
example: CardHolder Town
Town of the card holder's address. Max 40 characters for a 3DS2 enabled
token.

postCode* string
pattern: ^[0-9a-zA-Z -]$*
example: AB1 2CD
Postcode of the card holder's address.

state string
pattern: ^[A-Z]{2}$
example: FL
This attribute can be used by merchants to provide state/province/territ
ISO codes (2-char alpha) - specifically for countries where this is requ
as part of the address (Eg: US, Canada, India, China).

countryCode integer
pattern: ^\d{2,3}$
example: 826
ISO 3166-1 (3 digit) country code of the card holder's address. If the f
digit of the of the code is 0 this will not be shown

cardHolderName string
example: John Doe
Name of card holder.

}

clientIpAddress string
example: 1.2.3.4
[Conditional] IP address of caller, if provided in the session creation

clientUserAgent string
example: Mozilla/5.0
[Conditional] User agent of caller, if provided in the session creation

companyName string
example: Test Merchant
Trading name associated with JudoId specified for this session

currency string
example: GBP
The ISO-4217 alphabetic code for the currency associated with the payment session

expiryDate string($date-time)
example: 2026-02-05T16:28:32.8596+00:00
Date and time of expiry of payment session

judoId string
example: 100100100
Unique merchant and/or location ID supplied by Judopay used for this session



paymentCancelUrl
string
example: https://my.web.site/cancel
[Conditional] Merchant page to POST to after unsuccessful hosted web payment operations using this payment session, if defined on the API application associated with the authentication details

paymentSuccessUrl
string
example: https://my.web.site/success
[Conditional] Merchant page to POST to after successful hosted web payment operations using this p session, if defined on the API application associated with the authentication details

reference
string
example: 5QcAAAIAAAAMAAAADwAAAPwlgcQCLU6sLrNhi8kt1vEoVLwiFJc2EKTqT7JXb2xeQRqyYw
Unique Judo generated reference for this payment session

allowedCardTyes
[
minItems: 0
uniqueItems: true
example: [1, 2, 3, 10, 11, 12]
List of integer codes matching card types defined on https://docs.judopay.com/Content/Developer_Tools/Codes.htm?Highlight=codes#Card

integer]

response
{
description: Details provided when payment session was first created.

postUrl
string
example: https://pay-sandbox.judopay.com/v2
Prefix of URL directing to Judo hosted Web Payments user interface

reference
string
example: 5QcAAAIAAAAMAAAADwAAAPwlgcQCLU6sLrNhi8kt1vEoVLwiFJc2EKTqT7JXb2xeQRqyYw
Suffix to above URL to identify web payment

}

status
string
example: Success
Current state of the payment session

Enum:
[Open, Success, Expired, Cancelled]

transactionType
string
example: PreAuth
[Conditional] Intended use of payment session, if specified on creation

Enum:
[Payment, PreAuth, CheckCard, null]

yourConsumerReference
string
example: 2b45fd3f-cee5-4e7e-874f-28051db65408
Your unique reference to anonymously identify your customer for this session

yourPaymentMetaData
{
description: [Conditional] Key-value map for additional metadata associated with this transaction, if specified on session creation

}
example: { "internalLocationRef": "Example", "internalId": 99 }

yourPaymentReference
string
example: 6482c678-cad3-4efd-b081-aeae7a89a134
Your unique reference for this payment session

receipt
transactionReceiptHistoricResponse {
receiptId
string
example: 1001131610340495360
Judopay's reference for the transaction.

originalReceiptId
string
example: 1001124307998347264
[Conditional] Judopay's reference for any linked transaction (for collections, refunds and voids)

yourPaymentReference
string
example: 6482c678-cad3-4efd-b081-aeae7a89a134
Your unique reference for this payment.



type	string *example: PreAuth* The type of the transaction. Enum: [Payment, Refund, PreAuth, VOID, Collection, Save, Register, CheckCard]
createdAt	string($date-time) *example: 2025-02-05T16:28:32.8596+00:00* Date and time of creation.
result	string *example: Success* Result of transaction.
message	string *example: AuthCode: 123456* Message detailing the outcome.
judoId	integer *example: 100100100* Unique merchant and/or location ID supplied by Judopay.
merchantName	string *example: Test Merchant* Merchant's trading name.
appearsOnStatementAs	string *example: APL*/TestMerchant* Merchant description as it appears on the consumer's statement.
originalAmount	string *example: 10.99* Amount of original transaction (not affected by refunds or collect
amountCollected	string *example: 1.99* [Conditional - PreAuth and Register only] Amount that has been col on original pre-authorisation.
netAmount	string *example: 1.00* For payments and refunds, this is the original sale amount mines t total of all refunds. For preauths, incremental auths and collecti this is the original preauth amount plus the total of all incremer auths minus the total of all collections. For voids, this is equal the original preauth amount.
amount	string *example: 1.00* The amount of this transaction, for refunds and collections the am that has been refunded or collected in this transaction.
currency	string *example: GBP* The ISO-217 alphabetic code of the currency of the transaction
recurringPaymentType	string *example: MIT* [Conditional] Type of recurring payment (RECURRING for scheduled, for non-scheduled)
acquirerTransactionId	string *example: 33666277256892153705* The unique ID of the transaction set by the acquirer.
externalBankResponseCode	string *example: A* Response code set by the acquirer.
authCode	string *example: 123456* Authorisation code set by acquirer
postCodeCheckResult	string *example: Passed* Show results of post code checks



walletType
integer
example: 1
[Conditional] For digital wallet transactions, the type of wallet (1=ApplePay, 4=GooglePay, 5=ClickToPay)

acquirer
string
example: MyBank
Name of acquirer

webPaymentReference
string
example: 5QcAAAMAAAASAAAADAAAAHaKm0p0p-Ew6VrBhPxjxvMLinJOMZEJWG_K7kJ53UdvRGQ3qA
[Conditional] Judo identifier for associated web payment reference the transaction was associated with a payment session)

noOfAuthAttempts
integer
example: 1
[Conditional] This indicates the number of authentication attempts for transactions using the associated payment session reference.

paymentNetworkTransactionId
string
example: 123456789012345
[Conditional] Unique identifier of the transaction returned by the payment network/ card scheme.

allowIncrement
boolean
example: true
[Conditional] true for customer initiated pre-auths that had the allowIncrement request attribute set to true

isIncrementalAuth
boolean
example: true
[Conditional] true for calls to POST /transactions/incrementalAuth

disableNetworkTokenisation
boolean
example: true
[Conditional] true if the request explicitly disabled network toke this transaction

cardDetails
cardDetailsResponse {

cardLastfour
string
example: 1234
Last four digits of card

endDate
string
example: 1225
Card expiry date in format MMYY

cardToken
string
example: Ck3AeNlBfjvs9d61MNZiG0gtCvijqv
Tokenized representation of card which used in future transactions instead of number

cardType
integer
example: 2
Integer representing the type of card, Reference Codes - Card Types on https://docs.judopay.com/Content/Develo Tools/Codes.htm#CardTypes

startDate
string
example: 0121
Card start date in format MMYY

cardScheme
string
example: Mastercard
Name of card scheme (determined from ma start of card number against issuer dat

cardFunding
string
example: Credit
Type of card (determined from matching of card number against issuer data)

cardCategory
string
example: Acquirer Only



Category of card (determined from matchi
start of card number against issuer data

cardCountry
string
example: GB
ISO 3166-1 alpha-2 code of country that
issued the card (determined from matchi
start of card number against issuer data

bank
string
example: Santander Uk Plc
Name of bank that issued the card (dete
from matching start of card number again
issuer data)

cardHolderName
string
example: John Doe
Name of card holder as supplied on requ

ownerType
string
example: Personal
[Conditional] Indicator of whether a ca
personal or commercial, if known

Enum:
[Personal, Commercial, null]

}

billingAddress
cardAddressResponseAttributes {
description: Card holder address.

address1
string
pattern: ^[^<>\\]+$
example: CardHolder House
The first line of the cardholder's bill
address. Mastercard recommends providin
field for Mastercard 3DS-authenticated
transactions from 1 July 2026. Supplyin
billing address may improve issuer
authentication and authorisation decisi
Max 50 characters for a 3DS2 enabled AP
token.

address2
string
pattern: ^[^<>\\]+$
example: 1 CardHolder Street
Second line of the card holder's addres
50 characters for a 3DS2 enabled API to

town
string
pattern: ^[^<>\\]+$
example: CardHolder Town
Town of the card holder's address. Max
characters for a 3DS2 enabled API token

postCode*
string
pattern: ^[0-9a-zA-Z -]*$
example: AB1 2CD
Postcode of the card holder's address.

state
string
pattern: ^[A-Z]{2}$
example: FL
This attribute can be used by merchants
provide state/province/territory ISO co
(2-char alpha) - specifically for count
where this is required as part of the a
(Eg: US, Canada, India, China).

countryCode
integer
pattern: ^\d{2,3}$
example: 826
ISO 3166-1 (3 digit) country code of th
holder's address. If the first digit of
of the code is 0 this will not be shown

}

consumer
consumerResponse {
yourConsumerReference
string
example: 2b45fd3f-cee5-4e7e-874f-28051d



Your unique reference to anonymously id
your customer.

}

device

deviceResponse {

description: [Conditional] If clientDetails was provi
containing a kdeviceid

identifier

string

example: d73b4a7b58ce4e54a3bd73b7eda061
Judopay generated GUID associated with
kdeviceid passed in the clientDetails r
attribute

}

yourPaymentMetaData

{

description: Merchant metadata passed with transacti
request.

}

example: { "internalLocationRef": "Example", "internalId": 99 }

threeDSecure

threeDSecureCompletedResponse {

description: [Conditional] Only returned for transactions using 3DS

attempted

boolean

example: true
Did the consumer attempt to authent
through 3DSecure

result

string

example: PASSED
[Conditional] Outcome of the 3DSecu
authentication, only present on ini
receipt response, not historic rece

eci

string

example: 05
Electronic Commerce Indicator retur
3DS servers indicating outcome of
authentication

challengeRequestIndicator

string

nullable: true
example: ChallengeMandated
[Conditional] Challenge request ind
that was used for 3DS authenticatio

Enum:

[NoPreference, NoChallenge,
ChallengePreferred, ChallengeAsMand
ChallengeWithWhitelistPrompt, null

scaExemption

string

nullable: true
example:
[Conditional] SCA exemption that wa
for 3DS authentication

Enum:

[TrustedBeneficiary,
TransactionRiskAnalysis, DataShareO
ScaAlreadyPerformed, null]

}

risks

riskParametersResponse {

postCodeCheck

string

example: PASSED
Indication of whether supplied postCode
verified

Enum:

[PASSED, FAILED, UNKNOWN]

cv2Check

string

example: PASSED
Indication of whether supplied cv2 was
verified

Enum:



```
                                                                    [ PASSED, FAILED, NOT_PROCESSED,
                                                                    NOT_SUBMITTED, NOT_SUPPORTED, NOT_CHECK
                                                                    SUSPICIOUS, UNKNOWN ]
                                                merchantSuggestion  string
                                                                    example: Allow
                                                                    Suggestion action for the merchant from
                                                                    risk engine
                                            }
                    networkTokenisationDetails  networkTokenisationDetailsResponse {
                                                networkTokenProvisioned  boolean
                                                                         example: true
                                                                         Indicates if a new network token was
                                                                         created during the course of this
                                                                         transaction
                                                networkTokenUsed         boolean
                                                                         example: true
                                                                         Indicates if this transaction was com
                                                                         using a network token
                                                virtualPan               {
                                                                           lastFour    string
                                                                                       example: 1234
                                                                                       Last four digits of th
                                                                                       virtual PAN used
                                                                           expiryDate  string
                                                                                       example: 0129
                                                                                       Expiry date of the vir
                                                                                       PAN in format MMYY
                                                                         }
                                            }
                    }
portalUserRecId     integer
                    example: 1509318
                    [Conditional] Internal identifier of user who created payment session (if created through Portal o
                    Accept web app)
webPaymentOperation integer
                    example: 0
                    0 for Payment, 1 for Register
isPayByLink         boolean
                    example: true
                    [Conditional] Only shown if web payment session was created with isPayBLink value set to true
isJudoAccept        boolean
                    example: false
                    Flag showing whether session was created with isJudoAccept flag (or from POST /judoaccept/payments
isThreeDSecureTwo   boolean
                    example: false
                    [Conditional] Only shown when retrieving a single transaction. Indication of whether session will
                    require additional input for 3DS2 authentication.
mobileNumber        string
                    example: 07999999999
                    [Conditional] Only shown when retrieving a single transaction. Consumer mobile number specified wh
                    session was created.
phoneCountryCode    string
                    example: 44
                    [Conditional] Only shown when retrieving a single transaction. Consumer phone country code specifi
                    when session was created.
emailAddress        string
                    example: test.user@judopay.com
                    [Conditional] Only shown when retrieving a single transaction. Consumer email address specified wh
                    session was created.
noOfAuthAttempts    integer
                    example: 1
                    This indicates the number of authentication attempts made for transactions using this payment sess
chartReference
```



shortReference string
example: p5dqOA
Judopay generated short reference returned in payByLinkUrl when session is first created

}



paymentSessionHistoricListResponse {

amount
number($float)
example: 10.99
The amount to process. Format for default currencies is to two decimal places. For currencies using a different structure please contact Judopay for support.

cardAddress
{
description: [Conditional] Card holder address, if provided in the session creation

address1
string
pattern: ^[^<>\\]+$
example: CardHolder House
The first line of the cardholder's billing address. Mastercard recommends providing this field for Mastercard 3DS-authenticated transactions from 1 July 2026. Supplying the billing address may improve issuer authentication and authorisation decisions. Max 50 characters for a 3DS2 enabled API token.

address2
string
pattern: ^[^<>\\]+$
example: 1 CardHolder Street
Second line of the card holder's address. Max 50 characters for a 3DS2 enabled API token.

town
string
pattern: ^[^<>\\]+$
example: CardHolder Town
Town of the card holder's address. Max 40 characters for a 3DS2 enabled API token.

postCode*
string
pattern: ^[0-9a-zA-Z -]$*
example: AB1 2CD
Postcode of the card holder's address.

state
string
pattern: ^[A-Z]{2}$
example: FL
This attribute can be used by merchants to provide state/province/territory ISO codes (2-char alpha) - specifically for countries where this is required as part of the address (Eg: US, Canada, India, China).

countryCode
integer
pattern: ^\d{2,3}$
example: 826
ISO 3166-1 (3 digit) country code of the card holder's address. If the first digit of the of the code is 0 this will not be shown

cardHolderName
string
example: John Doe
Name of card holder.

}

clientIpAddress
string
example: 1.2.3.4
[Conditional] IP address of caller, if provided in the session creation

clientUserAgent
string
example: Mozilla/5.0
[Conditional] User agent of caller, if provided in the session creation

companyName
string
example: Test Merchant
Trading name associated with JudoId specified for this session

currency
string
example: GBP
The ISO-4217 alphabetic code for the currency associated with the payment session

expiryDate
string($date-time)
example: 2026-02-05T16:28:32.8596+00:00
Date and time of expiry of payment session

judoId
string



example: 100100100
Unique merchant and/or location ID supplied by Judopay used for this session

paymentCancelUrl
string
example: https://my.web.site/cancel
[Conditional] Merchant page to POST to after unsuccessful hosted web payment operations using this payment session, if defined on the API application associated with the authentication details

paymentSuccessUrl
string
example: https://my.web.site/success
[Conditional] Merchant page to POST to after successful hosted web payment operations using this payment session, if defined on the API application associated with the authentication details

reference
string
example: 5QcAAAIAAAAMAAAADwAAAPwlgcQCLU6sLrNhi8kt1vEoVLwiFJc2EKTqT7JXb2xeQRqyYw
Unique Judo generated reference for this payment session

allowedCardTyes
[
minItems: 0
uniqueItems: true
example: [1, 2, 3, 10, 11, 12]
List of integer codes matching card types defined on https://docs.judopay.com/Content/Developer_Tools/Codes.htm?Highlight=codes#Card

integer]

status
string
example: Open
Current state of the payment session

Enum:
[Open, Success, Expired, Cancelled]

transactionType
string
example: PreAuth
[Conditional] Intended use of payment session, if specified on creation

Enum:
[Payment, PreAuth, CheckCard, null]

yourConsumerReference
string
example: 2b45fd3f-cee5-4e7e-874f-28051db65408
Your unique reference to anonymously identify your customer for this session

yourPaymentMetaData
{
description: [Conditional] Key-value map for additional metadata associated with this transaction, if specified on session creation
}
example: { "internalLocationRef": "Example", "internalId": 99 }

yourPaymentReference
string
example: 6482c678-cad3-4efd-b081-aeae7a89a134
Your unique reference for this payment session

webPaymentOperation
integer
example: 0
0 for Payment, 1 for Register

isPayByLink
boolean
example: true
[Conditional] Only shown if web payment session was created with isPayBLink value set to true

isJudoAccept
boolean
example: false
Flag showing whether session was created with isJudoAccept flag (or from POST /judoaccept/payments)

~~**tradingName**~~
string
deprecated: true
example: Test Merchant
Duplicates companyName (which should be used instead)

}



paymentSessionRequest ⌄ {

judoId*
⌄ string
maxLength: 9
pattern: ^\d{9}$
example: 100100100
REQUIRED. Unique merchant and/or location ID supplied by Judopay. Do not include spaces or dashes. If not specified, then your default judoId will be used.

yourConsumerReference*
⌄ string
minLength: 1
maxLength: 50
example: 2b45fd3f-cee5-4e7e-874f-28051db65408
REQUIRED. Unique reference to anonymously identify your customer. Advisable to use GUIDs.

yourPaymentReference*
⌄ string
minLength: 1
maxLength: 50
example: 6482c678-cad3-4efd-b081-aeae7a89a134
REQUIRED. Your unique reference for this payment. This value should be unique in order to protect your customers against duplicate transactions. With a server side integration, if a payment reference is not supplied, the transaction will not be processed.

yourPaymentMetaData
⌄ {
description: Optional key-value map for additional metadata associated with this transaction. Will be stored but not processed or passed to gateways. Do not include sensitive information like card numbers.
}
example: { "internalLocationRef": "Example", "internalId": 99 }

currency
⌄ string
example: GBP
The ISO-4217 alphabetic code for the currency to use for the transaction, only required if different from the currency set during onboarding (default is GBP)

amount
⌄ number($float)
example: 10.99
The amount to process. Format for default currencies is to two decimal places. For currencies using a different structure please contact Judopay for support.

cardAddress
cardAddressWebPaymentRequest ⌄ {
description: Card holder address.

address1
⌄ string
pattern: ^[^<>\\]+$
example: CardHolder House
The first line of the cardholder's billing address. Mastercard recommends providing this field for Mastercard 3DS-authenticated transactions from 1 July 2026. Supplying the billing address may improve issuer authentication and authorisation decisions. Max 50 characters for a 3DS2 enabled API token.

address2
⌄ string
pattern: ^[^<>\\]+$
example: 1 CardHolder Street
Second line of the card holder's address. Max 50 characters for a 3DS2 enabled API token.

town
⌄ string
pattern: ^[^<>\\]+$
example: CardHolder Town
Town of the card holder's address. Max 40 characters for a 3DS2 enabled API token.

postCode*
⌄ string
pattern: ^[0-9a-zA-Z -]$*
example: AB1 2CD
Postcode of the card holder's address.

state
⌄ string
pattern: ^[A-Z]{2}$
example: FL
This attribute can be used by merchants to provide state/province/territory ISO codes (2-char alpha) - specifically for countries where this is required as part of the address (Eg: US, Canada, India, China).



countryCode
string
pattern: ^\d{3}$
example: 826
ISO 3166-1 (3 digit) country code of the card holder's address.

cardHolderName
string
example: John Doe
The cardholder's full name. Mastercard recommends providing this field for Mastercard 3DS-authenticated transactions from 1 July 2026. Supplying the cardholder name may improve issuer authentication and authorisation decisions.

}

expiryDate
string($date-time)
example: 2026-02-05T16:28:32.8596+00:00
Date and time of expiry (default is 30 minutes from creation)

isPayByLink
boolean
default: false
Flag indicating whether this session should be shown in the Pay By Link section in Portal.

isJudoAccept
boolean
default: false
Flag indicating whether this session should be shown in the Judo Accept section in Portal.

successUrl
string
maxLength: 2048
example: https://my.site.com/success
For web payments, this is the URL to which the customer is redirected if their transaction is successful. If not set, then the default success url specified on your account is used.

cancelUrl
string
maxLength: 2048
example: https://my.site.com/cancel
For web payments, this is the URL to which the customer is redirected if they cancel the transaction or if the transaction fails. If not set, then the default cancel url specified on your account is used.

emailAddress
string
pattern: ^.{0,254}$
example: test.user@judopay.com
The cardholder's email address. Mastercard recommends providing at least one contact method for Mastercard 3DS-authenticated transactions from 1 July 2026. A contact method can be an email address or supported telephone number. Required for Visa Network Token transactions to allow successful token provisioning.

mobileNumber
string
pattern: ^\d{0,15}$
example: 7999999999
The cardholder's mobile telephone number. Mastercard recommends providing at least one contact method for Mastercard 3DS-authenticated transactions from 1 July 2026. A contact method can be an email address or supported telephone number. Must be set if phoneCountryCode is set.

phoneCountryCode
string
default: 44
pattern: ^\d{0,3}$
example: 44
The international dialling code associated with the mobile telephone number. Should be set if mobileNumber is set, defaulted to 44 if not.

threeDSecure
{

challengeRequestIndicator
challengeRequestIndicatorRequest
challengeRequestIndicatorRequest string
nullable: true
default: noPreference
example: challengePreferred
Indicates whether a challenge is requested for this transaction (this may be over-ruled by the issuer). Should not be specified in the same request as scaExemption

Enum:
[noPreference, noChallenge, challengePreferred, challengeAsMandate, challengeWithWhitelistPrompt, null]

scaExemption
scaExemptionRequest
scaExemptionRequest string
nullable: true
example: transactionRiskAnalysis



example: transactionRiskAnalysis
Indicates reason why challenge may not be necessary (this may be over-ruled by the issuer). Should not be specified in the same request as challengeRequestIndicator

Enum:

[trustedBeneficiary, transactionRiskAnalysis, dataShareOnly, scaAlreadyPerformed, null]

}

hideBillingInfo

boolean

default: false if the API token used has 3DS enabled, true if not

example: true

(Optional) This flag can be used to determine whether the 'Billing information' page is shown on the Webpayments UI

hideReviewInfo

boolean

default: true

example: true

(Optional) This flag can be used to determine whether the 'Review and Confirm' page is shown on the Webpayments UI

disableNetworkTokenisation

boolean

default: false

example: true

(Optional) Set to true to specify that network tokenisation should not be used for the associated transaction, even if network token registration has been enabled on the account.

primaryAccountDetails

{

description: Information about the primary account holder of the transaction. Used for MCC 6012, MCC 6051 and MCC 7299 accounts only. Does not need to be specified if these details were set on a payment session associated with this transaction.

name*

string

maxLength: 100

pattern: ^\D+$

example: Doe

Surname of the recipient as registered with the account.

accountNumber*

string

maxLength: 10

pattern: ^\d{0,10}$

example: 12345678

Up to the first 10 digits of the primary account number of the recipient.

dateOfBirth*

string

pattern: ^\d{4}-\d{2}-\d{2}$

example: 1980-01-31

Date of birth of the recipient in yyyy-mm-dd format. If the recipient is a business, then the value should be set to "0000-00-00"

postCode*

string

maxLength: 10

example: AB1 2CD

Postcode registered against the recipient's account.

}

}



paymentSessionRequestForPreAuths {

judoId*
string
maxLength: 9
pattern: ^\d{9}$
example: 100100100
REQUIRED. Unique merchant and/or location ID supplied by Judopay. Do not include spaces or dashes. If not specified, then your default judoId will be used.

yourConsumerReference*
string
minLength: 1
maxLength: 50
example: 2b45fd3f-cee5-4e7e-874f-28051db65408
REQUIRED. Unique reference to anonymously identify your customer. Advisable to use GUIDs.

yourPaymentReference*
string
minLength: 1
maxLength: 50
example: 6482c678-cad3-4efd-b081-aeae7a89a134
REQUIRED. Your unique reference for this payment. This value should be unique in order to protect your customers against duplicate transactions. With a server side integration, if a payment reference is not supplied, the transaction will not be processed.

yourPaymentMetaData
{
description: Optional key-value map for additional metadata associated with this transaction. Will be stored but not processed or passed to gateways. Do not include sensitive information like card numbers.

}
example: { "internalLocationRef": "Example", "internalId": 99 }

currency
string
example: GBP
The ISO-4217 alphabetic code for the currency to use for the transaction, only required if different from the currency set during onboarding (default is GBP)

amount
number($float)
example: 10.99
The amount to process. Format for default currencies is to two decimal places. For currencies using a different structure please contact Judopay for support.

cardAddress
cardAddressWebPaymentRequest {
description: Card holder address.

address1
string
pattern: ^[^<>\\]+$
example: CardHolder House
The first line of the cardholder's billing address. Mastercard recommends providing this field for Mastercard 3DS-authenticated transactions from 1 July 2026. Supplying the billing address may improve issuer authentication and authorisation decisions. Max 50 characters for a 3DS2 enabled API token.

address2
string
pattern: ^[^<>\\]+$
example: 1 CardHolder Street
Second line of the card holder's address. Max 50 characters for a 3DS2 enabled API token.

town
string
pattern: ^[^<>\\]+$
example: CardHolder Town
Town of the card holder's address. Max 40 characters for a 3DS2 enabled API token.

postCode*
string
pattern: ^[0-9a-zA-Z -]$*
example: AB1 2CD
Postcode of the card holder's address.

state
string
pattern: ^[A-Z]{2}$
example: FL
This attribute can be used by merchants to provide state/province/territory ISO codes (2-char alpha) - specifically for countries where this is required as part of the address (Eg: US, Canada, India, China).



countryCode
string
pattern: ^\d{3}$
example: 826
ISO 3166-1 (3 digit) country code of the card holder's address.

cardHolderName
string
example: John Doe
The cardholder's full name. Mastercard recommends providing this field for Mastercard 3DS-authenticated transactions from 1 July 2026. Supplying the cardholder name may improve issuer authentication and authorisation decisions.

}

expiryDate
string($date-time)
example: 2026-02-05T16:28:32.8596+00:00
Date and time of expiry (default is 30 minutes from creation)

isPayByLink
boolean
default: false
Flag indicating whether this session should be shown in the Pay By Link section in Portal.

isJudoAccept
boolean
default: false
Flag indicating whether this session should be shown in the Judo Accept section in Portal.

successUrl
string
maxLength: 2048
example: https://my.site.com/success
For web payments, this is the URL to which the customer is redirected if their transaction is successful. If not set, then the default success url specified on your account is used.

cancelUrl
string
maxLength: 2048
example: https://my.site.com/cancel
For web payments, this is the URL to which the customer is redirected if they cancel the transaction or if the transaction fails. If not set, then the default cancel url specified on your account is used.

emailAddress
string
pattern: ^.{0,254}$
example: test.user@judopay.com
The cardholder's email address. Mastercard recommends providing at least one contact method for Mastercard 3DS-authenticated transactions from 1 July 2026. A contact method can be an email address or supported telephone number. Required for Visa Network Token transactions to allow successful token provisioning.

mobileNumber
string
pattern: ^\d{0,15}$
example: 7999999999
The cardholder's mobile telephone number. Mastercard recommends providing at least one contact method for Mastercard 3DS-authenticated transactions from 1 July 2026. A contact method can be an email address or supported telephone number. Must be set if phoneCountryCode is set.

phoneCountryCode
string
default: 44
pattern: ^\d{0,3}$
example: 44
The international dialling code associated with the mobile telephone number. Should be set if mobileNumber is set, defaulted to 44 if not.

threeDSecure
{

challengeRequestIndicator
challengeRequestIndicatorRequest
challengeRequestIndicatorRequest string
nullable: true
default: noPreference
example: challengePreferred
Indicates whether a challenge is requested for this transaction (this may be over-ruled by the issuer). Should not be specified in the same request as scaExemption

Enum:
[noPreference, noChallenge, challengePreferred, challengeAsMandate, challengeWithWhitelistPrompt, null]

scaExemption
scaExemptionRequest
scaExemptionRequest string
nullable: true
example: transactionRiskAnalysis



example: transactionRiskAnalysis
Indicates reason why challenge may not be necessary (this may be over-ruled by the issuer). Should not be specified in the same request as challengeRequestIndicator

Enum:

[trustedBeneficiary, transactionRiskAnalysis, dataShareOnly, scaAlreadyPerformed, null]

}

hideBillingInfo

boolean

default: false if the API token used has 3DS enabled, true if not

example: true

(Optional) This flag can be used to determine whether the 'Billing information' page is shown on the Webpayments UI

hideReviewInfo

boolean

default: true

example: true

(Optional) This flag can be used to determine whether the 'Review and Confirm' page is shown on the Webpayments UI

disableNetworkTokenisation

boolean

default: false

example: true

(Optional) Set to true to specify that network tokenisation should not be used for the associated transaction, even if network token registration has been enabled on the account.

primaryAccountDetails

{

description: Information about the primary account holder of the transaction. Used for MCC 6012, MCC 6051 and MCC 7299 accounts only. Does not need to be specified if these details were set on a payment session associated with this transaction.

name*

string

maxLength: 100

pattern: ^\D+$

example: Doe

Surname of the recipient as registered with the account.

accountNumber*

string

maxLength: 10

pattern: ^\d{0,10}$

example: 12345678

Up to the first 10 digits of the primary account number of the recipient.

dateOfBirth*

string

pattern: ^\d{4}-\d{2}-\d{2}$

example: 1980-01-31

Date of birth of the recipient in yyyy-mm-dd format. If the recipient is a business, then the value should be set to "0000-00-00"

postCode*

string

maxLength: 10

example: AB1 2CD

Postcode registered against the recipient's account.

}

allowIncrement

boolean

example: true

Set to true to create a session for customer initiated pre-auths that allow the merchant to increment the amount before it is fully collected.

}



paymentSessionRequestCommon {

judoId*
string
maxLength: 9
pattern: ^\d{9}$
example: 100100100
REQUIRED. Unique merchant and/or location ID supplied by Judopay. Do not include spaces or dashes. If not specified, then your default judoId will be used.

yourConsumerReference*
string
minLength: 1
maxLength: 50
example: 2b45fd3f-cee5-4e7e-874f-28051db65408
REQUIRED. Unique reference to anonymously identify your customer. Advisable to use GUIDs.

yourPaymentReference*
string
minLength: 1
maxLength: 50
example: 6482c678-cad3-4efd-b081-aeae7a89a134
REQUIRED. Your unique reference for this payment. This value should be unique in order to protect your customers against duplicate transactions. With a server side integration, if a payment reference is not supplied, the transaction will not be processed.

yourPaymentMetaData
{
description: Optional key-value map for additional metadata associated with this transaction. Will be stored but not processed or passed to gateways. Do not include sensitive information like card numbers.
}
example: { "internalLocationRef": "Example", "internalId": 99 }

currency
string
example: GBP
The ISO-4217 alphabetic code for the currency to use for the transaction, only required if different from the currency set during onboarding (default is GBP)

}

paymentSessionResponse {

expiryDate
string($date-time)
example: 2026-02-05T16:28:32.8596+00:00
Date and time of expiry of payment session

reference
string
example: 5QcAAAQAAAAPAAAACAAAABtGgvhBrF9BHTN7nqn1e0J4hVVmi-y27dGPjWBMtls3Gj_XDg
Judo internal reference for this payment session.

}

paymentSessionCancelResponse {

reference
string
example: 5QcAAAQAAAAPAAAACAAAABtGgvhBrF9BHTN7nqn1e0J4hVVmi-y27dGPjWBMtls3Gj_XDg
Judo internal reference for this payment session.

status
string
example: Cancelled
New status of payment session.

}



preAuthPaymentRequestCommon {

judoId*
string
maxLength: 9
pattern: ^\d{9}$
example: 100100100
REQUIRED. Unique merchant and/or location ID supplied by Judopay. Do not include spaces or dashes. If not specified, then your default judoId will be used.

yourConsumerReference*
string
minLength: 1
maxLength: 50
example: 2b45fd3f-cee5-4e7e-874f-28051db65408
REQUIRED. Unique reference to anonymously identify your customer. Advisable to use GUIDs.

yourPaymentReference*
string
minLength: 1
maxLength: 50
example: 6482c678-cad3-4efd-b081-aeae7a89a134
REQUIRED. Your unique reference for this payment. This value should be unique in order to protect your customers against duplicate transactions. With a server side integration, if a payment reference is not supplied, the transaction will not be processed.

yourPaymentMetaData
{
description: Optional key-value map for additional metadata associated with this transaction. Will be stored but not processed or passed to gateways. Do not include sensitive information like card numbers.
}
example: { "internalLocationRef": "Example", "internalId": 99 }

currency
string
example: GBP
The ISO-4217 alphabetic code for the currency to use for the transaction, only required if different from the currency set during onboarding (default is GBP)

amount*
number($float)
minimum: 0.01
example: 10.99
REQUIRED. The amount to process. Format for default currencies is to two decimal places. For currencies using a different structure please contact Judopay for support.

webPaymentReference
string
example: 5QcAAAIAAAAMAAAADwAAAPwlgcQCLU6sLrNhi8kt1vEoVLwiFJc2EKTqT7JXb2xeQRqyYw
The unique reference of the Judopay payment session associated with this transaction.

clientDetails
{
description: A set of encrypted fraud signals sent by the mobile SDK
key: string
value: string
}

initialRecurringPayment
boolean
example: false
Indicates if this is the first transaction that forms part of a recurring payment schedule. This flag should be set to 'true' if this is a customer-initiated payment which will be followed by a series of scheduled recurring payments.

primaryAccountDetails
{
description: Information about the primary account holder of the transaction. Used for MCC 6012, MCC 6051 and MCC 7299 accounts only. Does not need to be specified if these details were set on a payment session associated with this transaction.

name*
string
maxLength: 100
pattern: ^\D+$
example: Doe
Surname of the recipient as registered with the account.

accountNumber*
string
maxLength: 10
pattern: ^\d{0,10}$
example: 12345678
Up to the first 10 digits of the primary account number of the recipient.

dateOfBirth*




```
                                dateOfBirth                    string
                                                               pattern: ^\d{4}-\d{2}-\d{2}$
                                                               example: 1980-01-31
                                                               Date of birth of the recipient in yyyy-mm-dd format. If the
                                                               recipient is a business, then the value should be set to "0000-00-
                                                               00"
                                postCode*                      string
                                                               maxLength: 10
                                                               example: AB1 2CD
                                                               Postcode registered against the recipient's account.
                            }
}
```



Apple Pay {
pkPayment* {
token* {
paymentMethod {
displayName string
example: Visa 1234
Card description suitable for display,
network string
example: Visa
Name of the payment network backing the
type string
example: debit
Card's type of payment, provided by the
}
paymentData* {
data* string
example: CXnUZAtPGWfkLGsHksn5WTzfFhEzNMqSJZXKTmbaW
Encrypted payment data.
signature* string
example:
gbtqw7Zy9LcfsV53pSRxDJjwUKmrLAWkxVxL6NQzzChwGRMQ3d
Signature of the payment and header data. The signa
header {
description: Additional version-depen
publicKeyHash string
example: EMq6ggcTqGNYn86
ephemeralPublicKey string
example: Y5q3gWSJL8byWFH
transactionId string
example: 7c3b36d16bb0f6e
}
version string
example: EC_V1
Version information about the payment token. The t
}
}
}
cardAddress cardAddressRequestAttributes {
description: Card holder address.
address1 string
pattern: ^[^<>\\]+$
example: CardHolder House
The first line of the cardholder's billing address. Mastercard recomme
address2 string
pattern: ^[^<>\\]+$
example: 1 CardHolder Street
Second line of the card holder's address. Max 50 characters for a 3DS2
town string
pattern: ^[^<>\\]+$
example: CardHolder Town
Town of the card holder's address. Max 40 characters for a 3DS2 enable
postCode* string
pattern: ^[0-9a-zA-Z -]*$
example: AB1 2CD
Postcode of the card holder's address.
state string
pattern: ^[A-Z]{2}$
example: FL



This attribute can be used by merchants to provide state/province/terr

countryCode
string
pattern: ^\d{3}$
example: 826
ISO 3166-1 (3 digit) country code of the card holder's address.

}

judoId*
string
maxLength: 9
pattern: ^\d{9}$
example: 100100100
REQUIRED. Unique merchant and/or location ID supplied by Judopay. Do not include spaces or dashe

yourConsumerReference*
string
minLength: 1
maxLength: 50
example: 2b45fd3f-cee5-4e7e-874f-28051db65408
REQUIRED. Unique reference to anonymously identify your customer. Advisable to use GUIDs.

yourPaymentReference*
string
minLength: 1
maxLength: 50
example: 6482c678-cad3-4efd-b081-aeae7a89a134
REQUIRED. Your unique reference for this payment. This value should be unique in order to protec

yourPaymentMetaData
{
description: Optional key-value map for additional metadata associated with this tr
}
example: { "internalLocationRef": "Example", "internalId": 99 }

currency
string
example: GBP
The ISO-4217 alphabetic code for the currency to use for the transaction, only required if diffe

amount*
number($float)
minimum: 0.01
example: 10.99
REQUIRED. The amount to process. Format for default currencies is to two decimal places. For cur

webPaymentReference
string
example: 5QcAAAIAAAAMAAAADwAAAPwlgcQCLU6sLrNhi8kt1vEoVLwiFJc2EKTqT7JXb2xeQRqyYw
The unique reference of the Judopay payment session associated with this transaction.

clientDetails
{
description: A set of encrypted fraud signals sent by the mobile SDK
key
string
value
string
}

initialRecurringPayment
boolean
example: false
Indicates if this is the first transaction that forms part of a recurring payment schedule. This

primaryAccountDetails
{
description: Information about the primary account holder of the transaction. Used
name*
string
maxLength: 100
pattern: ^\D+$
example: Doe
Surname of the recipient as registered with the account.

accountNumber*
string
maxLength: 10
pattern: ^\d{0,10}$
example: 12345678
Up to the first 10 digits of the primary account number of the recipie

dateOfBirth*
string
pattern: ^\d{4}-\d{2}-\d{2}$
example: 1980-01-31
Date of birth of the recipient in yyyy-mm-dd format. If the recipient

postCode*
string



```
                                                    maxLength: 10
                                                    example: AB1 2CD
                                                    Postcode registered against the recipient's account.
                        }
    allowIncrement      boolean
                        example: true
                        Set to true to specify a customer initiated preauth can be incremented by the merchant before it
}
```



Card token {

cardToken*
string
example: Cw8eZ7nLBM2rFXx12WiRI0YveZPinkPm
REQUIRED. A randomly generated string linked to a card saved securely within the Judopay Card Vault.

cv2
string
minLength: 3
maxLength: 4
pattern: ^\d{3,4}$
example: 123
The 3 or 4 digit number on the back of a credit card. Also known as the card verification value (CVV) or security code.

cardAddress
cardAddressRequestAttributes {
description: Card holder address.

address1
string
pattern: ^[^<>\\]+$
example: CardHolder House
The first line of the cardholder's billing address. Mastercard recommends providing this field for Mastercard 3DS-authenticated transactions from 1 July 2026. Supplying the billing address may improve issuer authentication and authorisation decisions. Max 50 characters for a 3DS2 enabled API token.

address2
string
pattern: ^[^<>\\]+$
example: 1 CardHolder Street
Second line of the card holder's address. Max 50 characters for a 3DS2 enabled API token.

town
string
pattern: ^[^<>\\]+$
example: CardHolder Town
Town of the card holder's address. Max 40 characters for a 3DS2 enabled API token.

postCode*
string
pattern: ^[0-9a-zA-Z -]$*
example: AB1 2CD
Postcode of the card holder's address.

state
string
pattern: ^[A-Z]{2}$
example: FL
This attribute can be used by merchants to provide state/province/territory ISO codes (2-char alpha) - specifically for countries where this is required as part of the address (Eg: US, Canada, India, China).

countryCode
string
pattern: ^\d{3}$
example: 826
ISO 3166-1 (3 digit) country code of the card holder's address.

}

judoId*
string
maxLength: 9
pattern: ^\d{9}$
example: 100100100
REQUIRED. Unique merchant and/or location ID supplied by Judopay. Do not include spaces or dashes. If not specified, then your default judoId will be used.

yourConsumerReference*
string
minLength: 1
maxLength: 50
example: 2b45fd3f-cee5-4e7e-874f-28051db65408
REQUIRED. Unique reference to anonymously identify your customer. Advisable to use GUIDs.

yourPaymentReference*
string
minLength: 1
maxLength: 50



example: 6482c678-cad3-4efd-b081-aeae7a89a134
REQUIRED. Your unique reference for this payment. This value should be unique in order to protect your customers against duplicate transactions. With a server side integration, if a payment reference is not supplied, the transaction will not be processed.

yourPaymentMetaData

{
description: Optional key-value map for additional metadata associated with this transaction. Will be stored but not processed or passed to gateways. Do not include sensitive information like card numbers.

}
example: { "internalLocationRef": "Example", "internalId": 99 }

currency

string
example: GBP
The ISO-4217 alphabetic code for the currency to use for the transaction, only required if different from the currency set during onboarding (default is GBP)

amount*

number($float)
minimum: 0.01
example: 10.99
REQUIRED. The amount to process. Format for default currencies is to two decimal places. For currencies using a different structure please contact Judopay for support.

webPaymentReference

string
example: 5QcAAAIAAAAMAAAADwAAAPwlgcQCLU6sLrNhi8kt1vEoVLwiFJc2EKTqT7JXb2xeQRqyYw
The unique reference of the Judopay payment session associated with this transaction.

clientDetails

{
description: A set of encrypted fraud signals sent by the mobile SDK

key: string

value: string

}

initialRecurringPayment

boolean
example: false
Indicates if this is the first transaction that forms part of a recurring payment schedule. This flag should be set to 'true' if this is a customer-initiated payment which will be followed by a series of scheduled recurring payments.

primaryAccountDetails

{
description: Information about the primary account holder of the transaction. Used for MCC 6012, MCC 6051 and MCC 7299 accounts only. Does not need to be specified if these details were set on a payment session associated with this transaction.

name*

string
maxLength: 100
pattern: ^\D+$
example: Doe
Surname of the recipient as registered with the account.

accountNumber*

string
maxLength: 10
pattern: ^\d{0,10}$
example: 12345678
Up to the first 10 digits of the primary account number of the recipient.

dateOfBirth*

string
pattern: ^\d{4}-\d{2}-\d{2}$
example: 1980-01-31
Date of birth of the recipient in yyyy-mm-dd format. If the recipient is a business, then the value should be set to "0000-00-00"

postCode*

string
maxLength: 10
example: AB1 2CD
Postcode registered against the recipient's account.

}

acceptHeaders

string
default: text/html
example: text/html
Passed to ACS for use in rendering challenge screens.



userAgent

string

example: Mozilla/5.0 (iPhone; CPU iPhone OS 14_4 like Mac OS X) AppleWebKit/605.1.15 (KHTML, like Gecko) Mobile/15E148

Passed to ACS for use in rendering challenge screens.

cardHolderName

string

pattern: ^.{0,45}$

example: John Doe

The cardholder's full name. Mastercard recommends providing this field for Mastercard 3DS-authenticated transactions from 1 July 2026. Supplying the cardholder name may improve issuer authentication and authorisation decisions. Note: When testing in the sandbox environment, it is the cardHolderName that is used to determine the test card for 3D Secure 2 authentication.

emailAddress

string

pattern: ^.{0,254}$

example: test.user@judopay.com

The cardholder's email address. Mastercard recommends providing at least one contact method for Mastercard 3DS-authenticated transactions from 1 July 2026. A contact method can be an email address or supported telephone number. Required for Visa Network Token transactions to allow successful token provisioning.

shippingAddress

{

description: (Optional) Shipping address

address1

string

example: CardHolder House

First line of the shipping address.

address2

string

example: 1 CardHolder Street

Second line of the shipping address.

town

string

example: CardHolder Town

Town of the shipping address.

postCode

string

example: AB1 2CD

Postcode of the shipping address.

countryCode

string

pattern: ^\d{3}$

example: 826

ISO 3166-1 (3 digit) country code of the shipping address.

isBillingAddress

boolean

example: true

Indicator of whether the billing address is the same as the shipping address.

}

mobileNumber

string

pattern: ^\d{0,15}$

example: 7999999999

The cardholder's mobile telephone number. Mastercard recommends providing at least one contact method for Mastercard 3DS-authenticated transactions from 1 July 2026. A contact method can be an email address or supported telephone number. Must be set if phoneCountryCode is set.

phoneCountryCode

string

default: 44

pattern: ^\d{0,3}$

example: 44

The international dialling code associated with the mobile telephone number. Should be set if mobileNumber is set, defaulted to 44 if not.

threeDSecure

{

authenticationSource

string

example: Browser

Type of device that will be used for challenge authentication.

Enum:

[Browser]

methodNotificationUrl

string



default:
https://api.judopay.com/order/3ds/methodNotification
example:
https://api.judopay.com/order/3ds/methodNotification
URL of the page that will receive a POST event after a device details call is completed. Example is a Judo-hosted page that triggers a call to POST /transactions/{receiptId}/resume3ds (which will be used if this attribute is not set)

challengeNotificationUrl ⌄ **string**
default:
https://api.judopay.com/order/3ds/challengeNotificatio
example:
https://api.judopay.com/order/3ds/challengeNotificatio
URL of the page that will receive a POST event after a challenge is completed. Example is a Judo-hosted page that triggers a call to POST /transactions/{receiptId}/complete3ds (which will be used if this attribute is not set)

challengeRequestIndicator **challengeRequestIndicatorRequest** ⌄
challengeRequestIndicatorRequest **string**
nullable: true
default: noPreference
example: challengePreferred
Indicates whether a challenge is requested for this transaction (this may be over-ruled by the issuer). Should not be specified in the same request as scaExemption

Enum:

⌄ **[noPreference, noChallenge, challengePreferred, challengeAsMandate, challengeWithWhitelistPrompt, null]**

scaExemption **scaExemptionRequest** ⌄ **scaExemptionRequest** **string**
nullable: true
example: transactionRiskAnalysis
Indicates reason why challenge may not be necessary (this may be over-ruled by the issuer). Should not be specified in the same request as challengeRequestIndicator

Enum:

⌄ **[trustedBeneficiary, transactionRiskAnalysis, dataShareOnly, scaAlreadyPerformed, null]**

}

threeDSecureMpi ⌄ **{**

description: Optional block to pass in 3DS2 authentication results performed outside of Judopay.

dsTransId* ⌄ **string**
minLength: 36
maxLength: 64
pattern: ^[0-9a-zA-Z-]$*
example: 41aadf4c-e73f-4bd1-a0c4-acb2615a32af
REQUIRED. Unique identifier for the authentication assigned by the Directory Server (Card Scheme).

cavv* ⌄ **string**
minLength: 28
maxLength: 28
pattern: ^[a-zA-Z0-9+/=]$*
example: AJkBCWUyZwAAAABugmKTdAAAAAA=
REQUIRED. Unique value for the authentication provided by the ACS. May be used to provide proof of authentication.

eci* ⌄ **string**
minLength: 2
maxLength: 2
pattern: ^\d{2}$
example: 02
REQUIRED. Electronic Commerce Indicator (ECI) is the value returned by Directory Servers (namely Visa, MasterCard, JCB, and American Express) indicating the outcome of authentication attempted on transactions enforced by 3DS.

threeDSecureVersion* ⌄ **string**



minLength: 5

example: 2.2.0

REQUIRED. Version of 3DSecure that the card is enrolled into.

}

recurringPayment

boolean

example: true

Indicates this transaction is not being initiated by the customer. This should be set to 'true' if the merchant is trying to initiate a scheduled (recurring) or unscheduled (MIT) transaction in the absence of the customer.

recurringPaymentType

string

nullable: true

example: MIT

Indicates the type of recurring payment that the merchant is attempting to process. This can be set to 'RECURRING' (for scheduled payments) or 'MIT' (for unscheduled payments)

Enum:

[RECURRING, MIT, null]

relatedReceiptId

string

example: 1001131610340495360

Receipt identifier of the initial payment/pre-auth transaction that was customer initiated. If recurringPayment is true and recurringPaymentType is set, setting this to the Judo receipt identifier of the initial transaction in the sequence will check that original transaction was successful and on the same account, and pass the gateway transaction identifier of the original transaction on to the gateway with this transaction

relatedPaymentNetworkTransactionId

string

maxLength: 25

example: MNKWQVNQS0915

Card scheme identifier of the initial payment/pre-auth transaction that was customer initiated. If recurringPayment is true and recurringPaymentType is set, setting this to the card scheme identifier of the initial transaction in the sequence will pass this on on to the gateway with this transaction (no checks are made on this identifier). Should not be specified if relatedReceiptId is specified. Only supported by certain processors. Please check with Developer Support if you intend to use this field

allowIncrement

boolean

example: true

Set to true to specify a customer initiated preauth can be incremented by the merchant before it is fully collected.

disableNetworkTokenisation

boolean

default: false

example: true

(Optional) Set to true to specify that network tokenisation should not be used for this transaction, even if network token registration has been enabled on the account.

}



Google Pay {

googlePayWallet* {

token* string
example: "{\"signature\":\"MEUCIQD6IX46z4mg8SCoS\/UnbrgeUk+azwYFOHOHltdsO8D\/VQ
{\\\"encryptedMessage\\\":\\\"Vp5ofRY6BkFnIdM1kNvHdwAfORynkLah8nBXwCPUt5Jt5RFAx
JSON serialized payment token generated by Google Pay API

cardNetwork string
example: VISA
The payment card network returned by Google Pay API. Value should match one of t

cardDetails string
example: 1234
The details about the card returned by Google Pay API. This value is commonly th

}

cardAddress **cardAddressRequestAttributes** {

description: Card holder address.

address1 string
pattern: ^[^<>\\]+$
example: CardHolder House
The first line of the cardholder's billing address. Mastercard recomme

address2 string
pattern: ^[^<>\\]+$
example: 1 CardHolder Street
Second line of the card holder's address. Max 50 characters for a 3DS2

town string
pattern: ^[^<>\\]+$
example: CardHolder Town
Town of the card holder's address. Max 40 characters for a 3DS2 enable

postCode* string
pattern: ^[0-9a-zA-Z -]$*
example: AB1 2CD
Postcode of the card holder's address.

state string
pattern: ^[A-Z]{2}$
example: FL
This attribute can be used by merchants to provide state/province/terr

countryCode string
pattern: ^\d{3}$
example: 826
ISO 3166-1 (3 digit) country code of the card holder's address.

}

judoId* string
maxLength: 9
pattern: ^\d{9}$
example: 100100100
REQUIRED. Unique merchant and/or location ID supplied by Judopay. Do not include spaces or dashe

yourConsumerReference* string
minLength: 1
maxLength: 50
example: 2b45fd3f-cee5-4e7e-874f-28051db65408
REQUIRED. Unique reference to anonymously identify your customer. Advisable to use GUIDs.

yourPaymentReference* string
minLength: 1
maxLength: 50
example: 6482c678-cad3-4efd-b081-aeae7a89a134
REQUIRED. Your unique reference for this payment. This value should be unique in order to protec

yourPaymentMetaData {

description: Optional key-value map for additional metadata associated with this tr

}
example: { "internalLocationRef": "Example", "internalId": 99 }



currency
string
example: GBP
The ISO-4217 alphabetic code for the currency to use for the transaction, only required if diffe

amount*
number($float)
minimum: 0.01
example: 10.99
REQUIRED. The amount to process. Format for default currencies is to two decimal places. For cur

webPaymentReference
string
example: 5QcAAAIAAAAMAAAADwAAAPwlgcQCLU6sLrNhi8kt1vEoVLwiFJc2EKTqT7JXb2xeQRqyYw
The unique reference of the Judopay payment session associated with this transaction.

clientDetails
{
description: A set of encrypted fraud signals sent by the mobile SDK
key: string
value: string
}

initialRecurringPayment
boolean
example: false
Indicates if this is the first transaction that forms part of a recurring payment schedule. This

primaryAccountDetails
{
description: Information about the primary account holder of the transaction. Used
name*: string
maxLength: 100
pattern: ^\D+$
example: Doe
Surname of the recipient as registered with the account.
accountNumber*: string
maxLength: 10
pattern: ^\d{0,10}$
example: 12345678
Up to the first 10 digits of the primary account number of the recipie
dateOfBirth*: string
pattern: ^\d{4}-\d{2}-\d{2}$
example: 1980-01-31
Date of birth of the recipient in yyyy-mm-dd format. If the recipient
postCode*: string
maxLength: 10
example: AB1 2CD
Postcode registered against the recipient's account.
}

acceptHeaders
string
default: text/html
example: text/html
Passed to ACS for use in rendering challenge screens.

userAgent
string
example: Mozilla/5.0 (iPhone; CPU iPhone OS 14_4 like Mac OS X) AppleWebKit/605.1.15 (KHTML, lik
Passed to ACS for use in rendering challenge screens.

cardHolderName
string
pattern: ^.{0,45}$
example: John Doe
The cardholder's full name. Mastercard recommends providing this field for Mastercard 3DS-authen

emailAddress
string
pattern: ^.{0,254}$
example: test.user@judopay.com
The cardholder's email address. Mastercard recommends providing at least one contact method for

shippingAddress
{
description: (Optional) Shipping address
address1: string
example: CardHolder House
First line of the shipping address.



address2
string
example: 1 CardHolder Street
Second line of the shipping address.

town
string
example: CardHolder Town
Town of the shipping address.

postCode
string
example: AB1 2CD
Postcode of the shipping address.

countryCode
string
pattern: ^\d{3}$
example: 826
ISO 3166-1 (3 digit) country code of the shipping address.

isBillingAddress
boolean
example: true
Indicator of whether the billing address is the same as the shipping a

}

mobileNumber
string
pattern: ^\d{0,15}$
example: 7999999999
The cardholder's mobile telephone number. Mastercard recommends providing at least one contact m

phoneCountryCode
string
default: 44
pattern: ^\d{0,3}$
example: 44
The international dialling code associated with the mobile telephone number. Should be set if mo

threeDSecure
{

authenticationSource
string
example: Browser
Type of device that will be used for challenge authentication.

Enum:
[Browser]

methodNotificationUrl
string
default: https://api.judopay.com/order/3ds/methodNotification
example: https://api.judopay.com/order/3ds/methodNotification
URL of the page that will receive a POST event after a device deta

challengeNotificationUrl
string
default: https://api.judopay.com/order/3ds/challengeNotification
example: https://api.judopay.com/order/3ds/challengeNotification
URL of the page that will receive a POST event after a challenge i

challengeRequestIndicator
challengeRequestIndicatorRequest challengeRequestIndicator
nullable: true
default: noPreference
example: challengePreferred
Indicates whether a challenge is requested for this transaction (t

Enum:
[noPreference, noChallenge, challengePreferred, challengeAsMa

scaExemption
scaExemptionRequest scaExemptionRequest string
nullable: true
example: transactionRiskAnalysis
Indicates reason why challenge may not be necessary (this may be c

Enum:
[trustedBeneficiary, transactionRiskAnalysis, dataShareOnly,

}

threeDSecureMpi
{

description:
Optional block to pass in 3DS2 authentication results performed outsic

dsTransId*
string
minLength: 36
maxLength: 64
pattern: ^[0-9a-zA-Z-]$*



example: 41aadf4c-e73f-4bd1-a0c4-acb2615a32af
REQUIRED. Unique identifier for the authentication assigned by the Dir

cavv*
string
minLength: 28
maxLength: 28
pattern: ^[a-zA-Z0-9+/=]$*
example: AJkBCWUyZwAAAABugmKTdAAAAAA=
REQUIRED. Unique value for the authentication provided by the ACS. May

eci*
string
minLength: 2
maxLength: 2
pattern: ^\d{2}$
example: 02
REQUIRED. Electronic Commerce Indicator (ECI) is the value returned by

threeDSecureVersion*
string
minLength: 5
example: 2.2.0
REQUIRED. Version of 3DSecure that the card is enrolled into.

}

allowIncrement
boolean
example: true
Set to true to specify a customer initiated preauth can be incremented by the merchant before it

}



Card PAN {

cardNumber*
string
example: 1234567890123456
REQUIRED. The unique number printed on a credit card (13 to 19 digits depending on card type). Submitted without whitespace or non-numeric characters.

cv2
string
minLength: 3
maxLength: 4
pattern: ^\d{3,4}$
example: 123
The 3 or 4 digit number on the back of a credit card. Also known as the card verification value (CVV) or security code. If not specified, contact Judopay support to confirm gateway settings are configured to reflect this.

expiryDate*
string
pattern: ^\d{2}(\/)?\d{2}$
example: 01/28
REQUIRED. The expiry date of the card in format MM/YY

startDate
string
pattern: ^\d{2}(\/)?\d{2}$
example: 07/20
For Maestro cards only, start date in format MM/YY

cardAddress
cardAddressRequestAttributes {
description: Card holder address.

address1
string
pattern: ^[^<>\\]+$
example: CardHolder House
The first line of the cardholder's billing address. Mastercard recommends providing this field for Mastercard 3DS-authenticated transactions from 1 July 2026. Supplying the billing address may improve issuer authentication and authorisation decisions. Max 50 characters for a 3DS2 enabled API token.

address2
string
pattern: ^[^<>\\]+$
example: 1 CardHolder Street
Second line of the card holder's address. Max 50 characters for a 3DS2 enabled API token.

town
string
pattern: ^[^<>\\]+$
example: CardHolder Town
Town of the card holder's address. Max 40 characters for a 3DS2 enabled API token.

postCode*
string
pattern: ^[0-9a-zA-Z -]$*
example: AB1 2CD
Postcode of the card holder's address.

state
string
pattern: ^[A-Z]{2}$
example: FL
This attribute can be used by merchants to provide state/province/territory ISO codes (2-char alpha) - specifically for countries where this is required as part of the address (Eg: US, Canada, India, China).

countryCode
string
pattern: ^\d{3}$
example: 826
ISO 3166-1 (3 digit) country code of the card holder's address.

}

judoId*
string
maxLength: 9
pattern: ^\d{9}$
example: 100100100
REQUIRED. Unique merchant and/or location ID supplied by Judopay. Do not include spaces or dashes. If not specified, then your default judoId will be used.



yourConsumerReference*

string
minLength: 1
maxLength: 50
example: 2b45fd3f-cee5-4e7e-874f-28051db65408
REQUIRED. Unique reference to anonymously identify your customer. Advisable to use GUIDs.

yourPaymentReference*

string
minLength: 1
maxLength: 50
example: 6482c678-cad3-4efd-b081-aeae7a89a134
REQUIRED. Your unique reference for this payment. This value should be unique in order to protect your customers against duplicate transactions. With a server side integration, if a payment reference is not supplied, the transaction will not be processed.

yourPaymentMetaData

{
description: Optional key-value map for additional metadata associated with this transaction. Will be stored but not processed or passed to gateways. Do not include sensitive information like card numbers.
}
example: { "internalLocationRef": "Example", "internalId": 99 }

currency

string
example: GBP
The ISO-4217 alphabetic code for the currency to use for the transaction, only required if different from the currency set during onboarding (default is GBP)

amount*

number($float)
minimum: 0.01
example: 10.99
REQUIRED. The amount to process. Format for default currencies is to two decimal places. For currencies using a different structure please contact Judopay for support.

webPaymentReference

string
example: 5QcAAAIAAAAMAAAADwAAAPwlgcQCLU6sLrNhi8kt1vEoVLwiFJc2EKTqT7JXb2xeQRqyYw
The unique reference of the Judopay payment session associated with this transaction.

clientDetails

{
description: A set of encrypted fraud signals sent by the mobile SDK
key: string
value: string
}

initialRecurringPayment

boolean
example: false
Indicates if this is the first transaction that forms part of a recurring payment schedule. This flag should be set to 'true' if this is a customer-initiated payment which will be followed by a series of scheduled recurring payments.

primaryAccountDetails

{
description: Information about the primary account holder of the transaction. Used for MCC 6012, MCC 6051 and MCC 7299 accounts only. Does not need to be specified if these details were set on a payment session associated with this transaction.

name*

string
maxLength: 100
pattern: ^\D+$
example: Doe
Surname of the recipient as registered with the account.

accountNumber*

string
maxLength: 10
pattern: ^\d{0,10}$
example: 12345678
Up to the first 10 digits of the primary account number of the recipient.

dateOfBirth*

string
pattern: ^\d{4}-\d{2}-\d{2}$
example: 1980-01-31
Date of birth of the recipient in yyyy-mm-dd format. If the recipient is a business, then the value should be set to "0000-00-00"



postCode*
string
maxLength: 10
example: AB1 2CD
Postcode registered against the recipient's account.

}

acceptHeaders
string
default: text/html
example: text/html
Passed to ACS for use in rendering challenge screens.

userAgent
string
example: Mozilla/5.0 (iPhone; CPU iPhone OS 14_4 like Mac OS X) AppleWebKit/605.1.15 (KHTML, like Gecko) Mobile/15E148
Passed to ACS for use in rendering challenge screens.

cardHolderName
string
pattern: ^.{0,45}$
example: John Doe
The cardholder's full name. Mastercard recommends providing this field for Mastercard 3DS-authenticated transactions from 1 July 2026. Supplying the cardholder name may improve issuer authentication and authorisation decisions. Note: When testing in the sandbox environment, it is the cardHolderName that is used to determine the test card for 3D Secure 2 authentication.

emailAddress
string
pattern: ^.{0,254}$
example: test.user@judopay.com
The cardholder's email address. Mastercard recommends providing at least one contact method for Mastercard 3DS-authenticated transactions from 1 July 2026. A contact method can be an email address or supported telephone number. Required for Visa Network Token transactions to allow successful token provisioning.

shippingAddress
{
description: (Optional) Shipping address

address1
string
example: CardHolder House
First line of the shipping address.

address2
string
example: 1 CardHolder Street
Second line of the shipping address.

town
string
example: CardHolder Town
Town of the shipping address.

postCode
string
example: AB1 2CD
Postcode of the shipping address.

countryCode
string
pattern: ^\d{3}$
example: 826
ISO 3166-1 (3 digit) country code of the shipping address.

isBillingAddress
boolean
example: true
Indicator of whether the billing address is the same as the shipping address.

}

mobileNumber
string
pattern: ^\d{0,15}$
example: 7999999999
The cardholder's mobile telephone number. Mastercard recommends providing at least one contact method for Mastercard 3DS-authenticated transactions from 1 July 2026. A contact method can be an email address or supported telephone number. Must be set if phoneCountryCode is set.

phoneCountryCode
string
default: 44
pattern: ^\d{0,3}$
example: 44
The international dialling code associated with the mobile telephone number. Should be set if mobileNumber is set, defaulted to 44 if not.



threeDSecure {

authenticationSource string

example: Browser

Type of device that will be used for challenge authentication.

Enum:

[Browser]

methodNotificationUrl string

default: https://api.judopay.com/order/3ds/methodNotification

example: https://api.judopay.com/order/3ds/methodNotification

URL of the page that will receive a POST event after a device details call is completed. Example is a Judo-hosted page that triggers a call to POST /transactions/{receiptId}/resume3ds (which will be used if this attribute is not set)

challengeNotificationUrl string

default: https://api.judopay.com/order/3ds/challengeNotificatio

example: https://api.judopay.com/order/3ds/challengeNotificatio

URL of the page that will receive a POST event after a challenge is completed. Example is a Judo-hosted page that triggers a call to POST /transactions/{receiptId}/complete3ds (which will be used if this attribute is not set)

challengeRequestIndicator challengeRequestIndicatorRequest

challengeRequestIndicatorRequest string

nullable: true

default: noPreference

example: challengePreferred

Indicates whether a challenge is requested for this transaction (this may be over-ruled by the issuer). Should not be specified in the same request as scaExemption

Enum:

[noPreference, noChallenge, challengePreferred, challengeAsMandate, challengeWithWhitelistPrompt, null]

scaExemption scaExemptionRequest scaExemptionRequest string

nullable: true

example: transactionRiskAnalysis

Indicates reason why challenge may not be necessary (this may be over-ruled by the issuer). Should not be specified in the same request as challengeRequestIndicator

Enum:

[trustedBeneficiary, transactionRiskAnalysis, dataShareOnly, scaAlreadyPerformed, null]

}

threeDSecureMpi {

description: Optional block to pass in 3DS2 authentication results performed outside of Judopay.

dsTransId* string

minLength: 36

maxLength: 64

pattern: ^[0-9a-zA-Z-]$*

example: 41aadf4c-e73f-4bd1-a0c4-acb2615a32af

REQUIRED. Unique identifier for the authentication assigned by the Directory Server (Card Scheme).

cavv* string

minLength: 28

maxLength: 28

pattern: ^[a-zA-Z0-9+/=]$*

example: AJkBCWUyZwAAAABugmKTdAAAAAA=

REQUIRED. Unique value for the authentication provided by the ACS. May be used to provide proof of authentication.



eci* string
minLength: 2
maxLength: 2
pattern: ^\d{2}$
example: 02
REQUIRED. Electronic Commerce Indicator (ECI) is the value returned by Directory Servers (namely Visa, MasterCard, JCB, and American Express) indicating the outcome of authentication attempted on transactions enforced by 3DS.

threeDSecureVersion* string
minLength: 5
example: 2.2.0
REQUIRED. Version of 3DSecure that the card is enrolled into.

}

recurringPayment boolean
example: true
Indicates this transaction is not being initiated by the customer. This should be set to 'true' if the merchant is trying to initiate a scheduled (recurring) or unscheduled (MIT) transaction in the absence of the customer.

recurringPaymentType string
nullable: true
example: MIT
Indicates the type of recurring payment that the merchant is attempting to process. This can be set to 'RECURRING' (for scheduled payments) or 'MIT' (for unscheduled payments)

Enum:
[RECURRING, MIT, null]

relatedReceiptId string
example: 1001131610340495360
Receipt identifier of the initial payment/pre-auth transaction that was customer initiated. If recurringPayment is true and recurringPaymentType is set, setting this to the Judo receipt identifier of the initial transaction in the sequence will check that original transaction was successful and on the same account, and pass the gateway transaction identifier of the original transaction on to the gateway with this transaction

relatedPaymentNetworkTransactionId string
maxLength: 25
example: MNKWQVNQS0915
Card scheme identifier of the initial payment/pre-auth transaction that was customer initiated. If recurringPayment is true and recurringPaymentType is set, setting this to the card scheme identifier of the initial transaction in the sequence will pass this on on to the gateway with this transaction (no checks are made on this identifier). Should not be specified if relatedReceiptId is specified. Only supported by certain processors. Please check with Developer Support if you intend to use this field

allowIncrement boolean
example: true
Set to true to specify a customer initiated preauth can be incremented by the merchant before it is fully collected.

disableNetworkTokenisation boolean
default: false
example: true
(Optional) Set to true to specify that network tokenisation should not be used for this transaction, even if network token registration has been enabled on the account.

}



primaryAccountDetailsRequest {

primaryAccountDetails {

description: Information about the primary account holder of the transaction. Used for MCC 6012, MCC 6051 and MCC 7299 accounts only. Does not need to be specified if these details were set on a payment session associated with this transaction.

name* string
maxLength: 100
pattern: ^\D+$
example: Doe
Surname of the recipient as registered with the account.

accountNumber* string
maxLength: 10
pattern: ^\d{0,10}$
example: 12345678
Up to the first 10 digits of the primary account number of the recipient.

dateOfBirth* string
pattern: ^\d{4}-\d{2}-\d{2}$
example: 1980-01-31
Date of birth of the recipient in yyyy-mm-dd format. If the recipient is a business, then the value should be set to "0000-00-00"

postCode* string
maxLength: 10
example: AB1 2CD
Postcode registered against the recipient's account.

}

}

recurringPaymentRequest {

recurringPayment boolean
example: true
Indicates this transaction is not being initiated by the customer. This should be set to 'true' if the merchant is trying to initiate a scheduled (recurring) or unscheduled (MIT) transaction in the absence of the customer.

recurringPaymentType string
nullable: true
example: MIT
Indicates the type of recurring payment that the merchant is attempting to process. This can be set to 'RECURRING' (for scheduled payments) or 'MIT' (for unscheduled payments)

Enum:
[RECURRING, MIT, null]

relatedReceiptId string
example: 1001131610340495360
Receipt identifier of the initial payment/pre-auth transaction that was customer initiated. If recurringPayment is true and recurringPaymentType is set, setting this to the Judo receipt identifier of the initial transaction in the sequence will check that original transaction was successful and on the same account, and pass the gateway transaction identifier of the original transaction on to the gateway with this transaction

relatedPaymentNetworkTransactionId string
maxLength: 25
example: MNKWQVNQS0915
Card scheme identifier of the initial payment/pre-auth transaction that was customer initiated. If recurringPayment is true and recurringPaymentType is set, setting this to the card scheme identifier of the initial transaction in the sequence will pass this on on to the gateway with this transaction (no checks are made on this identifier). Should not be specified if relatedReceiptId is specified. Only supported by certain processors. Please check with Developer Support if you intend to use this field

}



refundRequest {

receiptId* — string
example: 1001131610340495360
REQUIRED. Judopay's reference for the transaction that is to be refunded.

amount — number($float)
minimum: 0.01
example: 10.99
The amount to refund. Format for default currencies is to two decimal places. For currencies using a different structure please contact Judopay for support. If not specified, the remaining amount will be refunded.

yourPaymentReference* — string
minLength: 1
maxLength: 50
example: 6482c678-cad3-4efd-b081-aeae7a89a134
REQUIRED. Your unique reference for this refund (not the yourPaymentReference of the original transaction)

}



registerCardRequest {

yourConsumerReference* string
minLength: 1
maxLength: 50
example: 2b45fd3f-cee5-4e7e-874f-28051db65408
REQUIRED. Unique reference to anonymously identify your customer. Advisable to use GUIDs.

yourPaymentReference string
minLength: 1
maxLength: 50
example: 6482c678-cad3-4efd-b081-aeae7a89a134
Your unique reference for this payment. This value should be unique in order to protect your customers against duplicate transactions. If not supplied, then a unique id will be generated internally

clientDetails {
description: A set of fraud signals sent by the mobile SDK
key string
value string
}

judoId string
maxLength: 9
pattern: ^\d{9}$
example: 100100100
Unique merchant and/or location ID supplied by Judopay. Do not include spaces or dashes. If not specified, then your default judoId will be used.

currency string
example: GBP
The currency to use for the pre-authorisation, only required if different from the currency set during onboarding (default is GBP)

initialRecurringPayment boolean
example: false
Indicates if this is the first transaction that forms part of a recurring payment schedule. This flag should be set to 'true' if this is a customer-initiated payment which will be followed by a series of scheduled recurring payments.

cardHolderName string
pattern: ^.{0,45}$
example: John Doe
The cardholder's full name. Mastercard recommends providing this field for Mastercard 3DS-authenticated transactions from 1 July 2026. Supplying the cardholder name may improve issuer authentication and authorisation decisions. Note: When testing in the sandbox environment, it is the cardHolderName that is used to determine the test card for 3D Secure 2 authentication.

emailAddress string
pattern: ^.{0,254}$
example: test.user@judopay.com
The cardholder's email address. Mastercard recommends providing at least one contact method for Mastercard 3DS-authenticated transactions from 1 July 2026. A contact method can be an email address or supported telephone number. Required for Visa Network Token transactions to allow successful token provisioning.

shippingAddress {
description: (Optional) Shipping address
address1 string
example: CardHolder House
First line of the shipping address.
address2 string
example: 1 CardHolder Street
Second line of the shipping address.
town string
example: CardHolder Town
Town of the shipping address.
postCode string
example: AB1 2CD
Postcode of the shipping address.
countryCode string



pattern: ^\d{3}$
example: 826
ISO 3166-1 (3 digit) country code of the shipping address.

isBillingAddress ⌄ **boolean**
example: true
Indicator of whether the billing address is the same as the shipping address.

}

mobileNumber ⌄ **string**
pattern: ^\d{0,15}$
example: 7999999999
The cardholder's mobile telephone number. Mastercard recommends providing at least one contact method for Mastercard 3DS-authenticated transactions from 1 July 2026. A contact method can be an email address or supported telephone number. Must be set if phoneCountryCode is set.

phoneCountryCode ⌄ **string**
default: 44
pattern: ^\d{0,3}$
example: 44
The international dialling code associated with the mobile telephone number. Should be set if mobileNumber is set, defaulted to 44 if not.

threeDSecure ⌄ {

authenticationSource ⌄ **string**
example: Browser
Type of device that will be used for challenge authentication.

Enum:

⌄ **[Browser]**

methodNotificationUrl ⌄ **string**
default: https://api.judopay.com/order/3ds/methodNotification
example: https://api.judopay.com/order/3ds/methodNotification
URL of the page that will receive a POST event after a device details call is completed. Example is a Judo-hosted page that triggers a call to POST /transactions/{receiptId}/resume3ds (which will be used if this attribute is not set)

challengeNotificationUrl ⌄ **string**
default: https://api.judopay.com/order/3ds/challengeNotification
example: https://api.judopay.com/order/3ds/challengeNotification
URL of the page that will receive a POST event after a challenge is completed. Example is a Judo-hosted page that triggers a call to POST /transactions/{receiptId}/complete3ds (which will be used if this attribute is not set)

challengeRequestIndicator challengeRequestIndicatorRequest ⌄
challengeRequestIndicatorRequest **string**
nullable: true
default: noPreference
example: challengePreferred
Indicates whether a challenge is requested for this transaction (this may be over-ruled by the issuer). Should not be specified in the same request as scaExemption

Enum:

⌄ **[noPreference, noChallenge, challengePreferred, challengeAsMandate, challengeWithWhitelistPrompt, null]**

scaExemption scaExemptionRequest ⌄ **scaExemptionRequest** **string**
nullable: true
example: transactionRiskAnalysis
Indicates reason why challenge may not be necessary (this may be over-ruled by the issuer). Should not be specified in the same request as challengeRequestIndicator

Enum:

⌄ **[trustedBeneficiary, transactionRiskAnalysis, dataShareOnly, scaAlreadyPerformed, null]**

}

threeDSecureMpi ⌄ {

description: Optional block to pass in 3DS2 authentication results performed outside of Judopay.



dsTransId* string
minLength: 36
maxLength: 64
pattern: ^[0-9a-zA-Z-]*$
example: 41aadf4c-e73f-4bd1-a0c4-acb2615a32af
REQUIRED. Unique identifier for the authentication assigned by the Directory Server (Card Scheme).

cavv* string
minLength: 28
maxLength: 28
pattern: ^[a-zA-Z0-9+/=]*$
example: AJkBCWUyZwAAAABugmKTdAAAAAA=
REQUIRED. Unique value for the authentication provided by the ACS. May be used to provide proof of authentication.

eci* string
minLength: 2
maxLength: 2
pattern: ^\d{2}$
example: 02
REQUIRED. Electronic Commerce Indicator (ECI) is the value returned by Directory Servers (namely Visa, MasterCard, JCB, and American Express) indicating the outcome of authentication attempted on transactions enforced by 3DS.

threeDSecureVersion* string
minLength: 5
example: 2.2.0
REQUIRED. Version of 3DSecure that the card is enrolled into.

}

cardAddress

cardAddressRequestAttributes {
description: Card holder address.

address1 string
pattern: ^[^<>\\]+$
example: CardHolder House
The first line of the cardholder's billing address. Mastercard recommends providing this field for Mastercard 3DS-authenticated transactions from 1 July 2026. Supplying the billing address may improve issuer authentication and authorisation decisions. Max 50 characters for a 3DS2 enabled API token.

address2 string
pattern: ^[^<>\\]+$
example: 1 CardHolder Street
Second line of the card holder's address. Max 50 characters for a 3DS2 enabled API token.

town string
pattern: ^[^<>\\]+$
example: CardHolder Town
Town of the card holder's address. Max 40 characters for a 3DS2 enabled API token.

postCode* string
pattern: ^[0-9a-zA-Z -]*$
example: AB1 2CD
Postcode of the card holder's address.

state string
pattern: ^[A-Z]{2}$
example: FL
This attribute can be used by merchants to provide state/province/territory ISO codes (2-char alpha) - specifically for countries where this is required as part of the address (Eg: US, Canada, India, China).

countryCode string
pattern: ^\d{3}$
example: 826
ISO 3166-1 (3 digit) country code of the card holder's address.

}

primaryAccountDetails

{
description: Information about the primary account holder of the transaction. Used for MCC 6012, MCC 6051 and MCC 7299 accounts only. Does not



need to be specified if these details were set on a payment session associated with this transaction.

name* — string
maxLength: 100
pattern: ^\D+$
example: Doe
Surname of the recipient as registered with the account.

accountNumber* — string
maxLength: 10
pattern: ^\d{0,10}$
example: 12345678
Up to the first 10 digits of the primary account number of the recipient.

dateOfBirth* — string
pattern: ^\d{4}-\d{2}-\d{2}$
example: 1980-01-31
Date of birth of the recipient in yyyy-mm-dd format. If the recipient is a business, then the value should be set to "0000-00-00"

postCode* — string
maxLength: 10
example: AB1 2CD
Postcode registered against the recipient's account.

}
}



checkCardRequest {

yourConsumerReference*
string
minLength: 1
maxLength: 50
example: 2b45fd3f-cee5-4e7e-874f-28051db65408
REQUIRED. Unique reference to anonymously identify your customer. Advisable to use GUIDs.

yourPaymentReference
string
minLength: 1
maxLength: 50
example: 6482c678-cad3-4efd-b081-aeae7a89a134
Your unique reference for this payment. This value should be unique in order to protect your customers against duplicate transactions. If not supplied, then a unique id will be generated internally

clientDetails
{
description: A set of fraud signals sent by the mobile SDK
key: string
value: string
}

judoId
string
maxLength: 9
pattern: ^\d{9}$
example: 100100100
Unique merchant and/or location ID supplied by Judopay. Do not include spaces or dashes. If not specified, then your default judoId will be used.

currency
string
example: GBP
The currency to use for the pre-authorisation, only required if different from the currency set during onboarding (default is GBP)

initialRecurringPayment
boolean
example: false
Indicates if this is the first transaction that forms part of a recurring payment schedule. This flag should be set to 'true' if this is a customer-initiated payment which will be followed by a series of scheduled recurring payments.

disableNetworkTokenisation
boolean
default: false
example: true
(Optional) Set to true to specify that network tokenisation should not be used for this transaction, even if network token registration has been enabled on the account.

cardHolderName
string
pattern: ^.{0,45}$
example: John Doe
The cardholder's full name. Mastercard recommends providing this field for Mastercard 3DS-authenticated transactions from 1 July 2026. Supplying the cardholder name may improve issuer authentication and authorisation decisions. Note: When testing in the sandbox environment, it is the cardHolderName that is used to determine the test card for 3D Secure 2 authentication.

emailAddress
string
pattern: ^.{0,254}$
example: test.user@judopay.com
The cardholder's email address. Mastercard recommends providing at least one contact method for Mastercard 3DS-authenticated transactions from 1 July 2026. A contact method can be an email address or supported telephone number. Required for Visa Network Token transactions to allow successful token provisioning.

shippingAddress
{
description: (Optional) Shipping address

address1
string
example: CardHolder House
First line of the shipping address.

address2
string
example: 1 CardHolder Street
Second line of the shipping address.

town
string
example: CardHolder Town
Town of the shipping address



postCode
string
example: AB1 2CD
Postcode of the shipping address.

countryCode
string
pattern: ^\d{3}$
example: 826
ISO 3166-1 (3 digit) country code of the shipping address.

isBillingAddress
boolean
example: true
Indicator of whether the billing address is the same as the shipping address.

}

mobileNumber
string
pattern: ^\d{0,15}$
example: 7999999999
The cardholder's mobile telephone number. Mastercard recommends providing at least one contact method for Mastercard 3DS-authenticated transactions from 1 July 2026. A contact method can be an email address or supported telephone number. Must be set if phoneCountryCode is set.

phoneCountryCode
string
default: 44
pattern: ^\d{0,3}$
example: 44
The international dialling code associated with the mobile telephone number. Should be set if mobileNumber is set, defaulted to 44 if not.

threeDSecure
{
authenticationSource
string
example: Browser
Type of device that will be used for challenge authentication.

Enum:
[Browser]

methodNotificationUrl
string
default: https://api.judopay.com/order/3ds/methodNotification
example: https://api.judopay.com/order/3ds/methodNotification
URL of the page that will receive a POST event after a device details call is completed. Example is a Judo-hosted page that triggers a call to POST /transactions/{receiptId}/resume3ds (which will be used if this attribute is not set)

challengeNotificationUrl
string
default: https://api.judopay.com/order/3ds/challengeNotification
example: https://api.judopay.com/order/3ds/challengeNotification
URL of the page that will receive a POST event after a challenge is completed. Example is a Judo-hosted page that triggers a call to POST /transactions/{receiptId}/complete3ds (which will be used if this attribute is not set)

}

threeDSecureMpi
{
description: Optional block to pass in 3DS2 authentication results performed outside of Judopay.

dsTransId*
string
minLength: 36
maxLength: 64
pattern: ^[0-9a-zA-Z-]$*
example: 41aadf4c-e73f-4bd1-a0c4-acb2615a32af
REQUIRED. Unique identifier for the authentication assigned by the Directory Server (Card Scheme).

cavv*
string
minLength: 28
maxLength: 28



pattern: ^[a-zA-Z0-9+/=]$*
example: AJkBCWUyZwAAAABugmKTdAAAAAA=
REQUIRED. Unique value for the authentication provided by the ACS. May be used to provide proof of authentication.

eci* string
minLength: 2
maxLength: 2
pattern: ^\d{2}$
example: 02
REQUIRED. Electronic Commerce Indicator (ECI) is the value returned by Directory Servers (namely Visa, MasterCard, JCB, and American Express) indicating the outcome of authentication attempted on transactions enforced by 3DS.

threeDSecureVersion* string
minLength: 5
example: 2.2.0
REQUIRED. Version of 3DSecure that the card is enrolled into.

}

cardAddress cardAddressRequestAttributes {

description: Card holder address.

address1 string
pattern: ^[^<>\\]+$
example: CardHolder House
The first line of the cardholder's billing address. Mastercard recommends providing this field for Mastercard 3DS-authenticated transactions from 1 July 2026. Supplying the billing address may improve issuer authentication and authorisation decisions. Max 50 characters for a 3DS2 enabled API token.

address2 string
pattern: ^[^<>\\]+$
example: 1 CardHolder Street
Second line of the card holder's address. Max 50 characters for a 3DS2 enabled API token.

town string
pattern: ^[^<>\\]+$
example: CardHolder Town
Town of the card holder's address. Max 40 characters for a 3DS2 enabled API token.

postCode* string
pattern: ^[0-9a-zA-Z -]$*
example: AB1 2CD
Postcode of the card holder's address.

state string
pattern: ^[A-Z]{2}$
example: FL
This attribute can be used by merchants to provide state/province/territory ISO codes (2-char alpha) - specifically for countries where this is required as part of the address (Eg: US, Canada, India, China).

countryCode string
pattern: ^\d{3}$
example: 826
ISO 3166-1 (3 digit) country code of the card holder's address.

}

primaryAccountDetails {

description: Information about the primary account holder of the transaction. Used for MCC 6012, MCC 6051 and MCC 7299 accounts only. Does not need to be specified if these details were set on a payment session associated with this transaction.

name* string
maxLength: 100
pattern: ^\D+$
example: Doe
Surname of the recipient as registered with the account.

accountNumber* string
maxLength: 10



pattern: ^\d{0,10}$
example: 12345678
Up to the first 10 digits of the primary account number of the recipient.

dateOfBirth* string
pattern: ^\d{4}-\d{2}-\d{2}$
example: 1980-01-31
Date of birth of the recipient in yyyy-mm-dd format. If the recipient is a business, then the value should be set to "0000-00-00"

postCode* string
maxLength: 10
example: AB1 2CD
Postcode registered against the recipient's account.

}
}



registerCardRequestPan {

cardNumber* string
example: 1234567890123456
REQUIRED. The unique number printed on a credit card (13 to 19 digits depending on card type). Submitted without whitespace or non-numeric characters.

cv2 string
minLength: 3
maxLength: 4
pattern: ^\d{3,4}$
example: 123
The 3 or 4 digit number on the back of a credit card. Also known as the card verification value (CVV) or security code. If not specified, contact Judopay support to confirm gateway settings are configured to reflect this.

expiryDate* string
pattern: ^\d{2}(\/)?\d{2}$
example: 01/28
REQUIRED. The expiry date of the card in format MM/YY

startDate string
pattern: ^\d{2}(\/)?\d{2}$
example: 07/20
For Maestro cards only, start date in format MM/YY

yourConsumerReference* string
minLength: 1
maxLength: 50
example: 2b45fd3f-cee5-4e7e-874f-28051db65408
REQUIRED. Unique reference to anonymously identify your customer. Advisable to use GUIDs.

yourPaymentReference string
minLength: 1
maxLength: 50
example: 6482c678-cad3-4efd-b081-aeae7a89a134
Your unique reference for this payment. This value should be unique in order to protect your customers against duplicate transactions. If not supplied, then a unique id will be generated internally

clientDetails {
description: A set of fraud signals sent by the mobile SDK
key string
value string
}

judoId string
maxLength: 9
pattern: ^\d{9}$
example: 100100100
Unique merchant and/or location ID supplied by Judopay. Do not include spaces or dashes. If not specified, then your default judoId will be used.

currency string
example: GBP
The currency to use for the pre-authorisation, only required if different from the currency set during onboarding (default is GBP)

initialRecurringPayment boolean
example: false
Indicates if this is the first transaction that forms part of a recurring payment schedule. This flag should be set to 'true' if this is a customer-initiated payment which will be followed by a series of scheduled recurring payments.

cardHolderName string
pattern: ^.{0,45}$
example: John Doe
The cardholder's full name. Mastercard recommends providing this field for Mastercard 3DS-authenticated transactions from 1 July 2026. Supplying the cardholder name may improve issuer authentication and authorisation decisions. Note: When testing in the sandbox environment, it is the cardHolderName that is used to determine the test card for 3D Secure 2 authentication.

emailAddress string
pattern: ^.{0,254}$
example: test.user@judopay.com
The cardholder's email address. Mastercard recommends providing at least one contact method



for Mastercard 3DS-authenticated transactions from 1 July 2026. A contact method can be an email address or supported telephone number. Required for Visa Network Token transactions to allow successful token provisioning.

shippingAddress

{
- description: (Optional) Shipping address
- address1: string
  - *example: CardHolder House*
  - First line of the shipping address.
- address2: string
  - *example: 1 CardHolder Street*
  - Second line of the shipping address.
- town: string
  - *example: CardHolder Town*
  - Town of the shipping address.
- postCode: string
  - *example: AB1 2CD*
  - Postcode of the shipping address.
- countryCode: string
  - *pattern: ^\d{3}$*
  - *example: 826*
  - ISO 3166-1 (3 digit) country code of the shipping address.
- isBillingAddress: boolean
  - *example: true*
  - Indicator of whether the billing address is the same as the shipping address.

}

mobileNumber

string
pattern: ^\d{0,15}$
example: 7999999999
The cardholder's mobile telephone number. Mastercard recommends providing at least one contact method for Mastercard 3DS-authenticated transactions from 1 July 2026. A contact method can be an email address or supported telephone number. Must be set if phoneCountryCode is set.

phoneCountryCode

string
default: 44
pattern: ^\d{0,3}$
example: 44
The international dialling code associated with the mobile telephone number. Should be set if mobileNumber is set, defaulted to 44 if not.

threeDSecure

{
- authenticationSource: string
  - *example: Browser*
  - Type of device that will be used for challenge authentication.
  - Enum:
  - [Browser]
- methodNotificationUrl: string
  - *default: https://api.judopay.com/order/3ds/methodNotification*
  - *example: https://api.judopay.com/order/3ds/methodNotification*
  - URL of the page that will receive a POST event after a device details call is completed. Example is a Judo-hosted page that triggers a call to POST /transactions/{receiptId}/resume3ds (which will be used if this attribute is not set)
- challengeNotificationUrl: string
  - *default: https://api.judopay.com/order/3ds/challengeNotification*
  - *example: https://api.judopay.com/order/3ds/challengeNotification*
  - URL of the page that will receive a POST event after a challenge is completed. Example is a Judo-hosted page that triggers a call to POST /transactions/{receiptId}/complete3ds (which will be used if this attribute is not set)
- challengeRequestIndicator: challengeRequestIndicatorRequest
  - challengeRequestIndicatorRequest string
  - *nullable: true*



default: noPreference
example: challengePreferred
Indicates whether a challenge is requested for this transaction (this may be over-ruled by the issuer). Should not be specified in the same request as scaExemption

Enum:

[noPreference, noChallenge, challengePreferred, challengeAsMandate, challengeWithWhitelistPrompt, null]

scaExemption

scaExemptionRequest scaExemptionRequest string
nullable: true
example: transactionRiskAnalysis
Indicates reason why challenge may not be necessary (this may be over-ruled by the issuer). Should not be specified in the same request as challengeRequestIndicator

Enum:

[trustedBeneficiary, transactionRiskAnalysis, dataShareOnly, scaAlreadyPerformed, null]

}

threeDSecureMpi

{

description: Optional block to pass in 3DS2 authentication results performed outside of Judopay.

dsTransId*

string
minLength: 36
maxLength: 64
pattern: ^[0-9a-zA-Z-]*$
example: 41aadf4c-e73f-4bd1-a0c4-acb2615a32af
REQUIRED. Unique identifier for the authentication assigned by the Directory Server (Card Scheme).

cavv*

string
minLength: 28
maxLength: 28
pattern: ^[a-zA-Z0-9+/=]*$
example: AJkBCWUyZwAAAABugmKTdAAAAAA=
REQUIRED. Unique value for the authentication provided by the ACS. May be used to provide proof of authentication.

eci*

string
minLength: 2
maxLength: 2
pattern: ^\d{2}$
example: 02
REQUIRED. Electronic Commerce Indicator (ECI) is the value returned by Directory Servers (namely Visa, MasterCard, JCB, and American Express) indicating the outcome of authentication attempted on transactions enforced by 3DS.

threeDSecureVersion*

string
minLength: 5
example: 2.2.0
REQUIRED. Version of 3DSecure that the card is enrolled into.

}

cardAddress

cardAddressRequestAttributes {

description: Card holder address.

address1

string
pattern: ^[^<>\\]+$
example: CardHolder House
The first line of the cardholder's billing address. Mastercard recommends providing this field for Mastercard 3DS-authenticated transactions from 1 July 2026. Supplying the billing address may improve issuer authentication and authorisation decisions. Max 50 characters for a 3DS2 enabled API token.

address2

string
pattern: ^[^<>\\]+$
example: 1 CardHolder Street
Second line of the card holder's address. Max 50 characters for a 3DS2 enabled API token.

town

string
pattern: ^[^<>\\]+$
example: CardHolder Town
Town of the card holder's address. Max 40 characters for a 3DS2



enabled API token.

postCode* string
pattern: ^[0-9a-zA-Z -]*$
example: AB1 2CD
Postcode of the card holder's address.

state string
pattern: ^[A-Z]{2}$
example: FL
This attribute can be used by merchants to provide state/province/territory ISO codes (2-char alpha) - specifically for countries where this is required as part of the address (Eg: US, Canada, India, China).

countryCode string
pattern: ^\d{3}$
example: 826
ISO 3166-1 (3 digit) country code of the card holder's address.

}

primaryAccountDetails {
description: Information about the primary account holder of the transaction. Used for MCC 6012, MCC 6051 and MCC 7299 accounts only. Does not need to be specified if these details were set on a payment session associated with this transaction.

name* string
maxLength: 100
pattern: ^\D+$
example: Doe
Surname of the recipient as registered with the account.

accountNumber* string
maxLength: 10
pattern: ^\d{0,10}$
example: 12345678
Up to the first 10 digits of the primary account number of the recipient.

dateOfBirth* string
pattern: ^\d{4}-\d{2}-\d{2}$
example: 1980-01-31
Date of birth of the recipient in yyyy-mm-dd format. If the recipient is a business, then the value should be set to "0000-00-00"

postCode* string
maxLength: 10
example: AB1 2CD
Postcode registered against the recipient's account.

}

}



resume3dsRequest {

cv2

string

minLength: 3

maxLength: 4

pattern: ^\d{3,4}$

example: 123

The 3 or 4 digit number on the back of a credit card. Also known as the card verification value (CVV) or security code. If not specified, contact Judopay support to confirm gateway settings are configured to reflect this.

threeDSecure*

{

methodCompletion*

string

example: Yes

REQUIRED. Indicates whether the 3D Secure method (device details) check was completed successfully.

Enum:

[Yes, No, Unavailable]

}

primaryAccountDetails

{

description:

Information about the primary account holder of the transaction. Used for MCC 6012, MCC 6051 and MCC 7299 accounts only. Does not need to be specified if these details were set on a payment session associated with this transaction.

name*

string

maxLength: 100

pattern: ^\D+$

example: Doe

Surname of the recipient as registered with the account.

accountNumber*

string

maxLength: 10

pattern: ^\d{0,10}$

example: 12345678

Up to the first 10 digits of the primary account number of the recipient.

dateOfBirth*

string

pattern: ^\d{4}-\d{2}-\d{2}$

example: 1980-01-31

Date of birth of the recipient in yyyy-mm-dd format. If the recipient is a business, then the value should be set to "0000-00-00"

postCode*

string

maxLength: 10

example: AB1 2CD

Postcode registered against the recipient's account.

}

}



riskParametersResponse {

postCodeCheck string
example: PASSED
Indication of whether supplied postCode was verified

Enum:

[PASSED, FAILED, UNKNOWN]

cv2Check string
example: PASSED
Indication of whether supplied cv2 was verified

Enum:

[PASSED, FAILED, NOT_PROCESSED, NOT_SUBMITTED, NOT_SUPPORTED, NOT_CHECKED, SUSPICIOUS, UNKNOWN]

merchantSuggestion string
example: Allow
Suggestion action for the merchant from the risk engine

}

networkTokenisationDetailsResponse {

networkTokenProvisioned boolean
example: true
Indicates if a new network token was created during the course of this transaction

networkTokenUsed boolean
example: true
Indicates if this transaction was completed using a network token

virtualPan {

lastFour string
example: 1234
Last four digits of the virtual PAN used

expiryDate string
example: 0129
Expiry date of the virtual PAN in format MMYY

}

}



saveCardRequestCommon {

yourConsumerReference*
string
minLength: 1
maxLength: 50
example: 2b45fd3f-cee5-4e7e-874f-28051db65408
REQUIRED. Unique reference to anonymously identify your customer. Advisable to use GUIDs.

judoId
string
maxLength: 9
pattern: ^\d{9}$
example: 100100100
Unique merchant and/or location ID supplied by Judopay. Do not include spaces or dashes. If not specified, then your default judoId will be used.

currency
string
example: GBP
The currency to use for the pre-authorisation, only required if different from the currency set during onboarding (default is GBP)

cardHolderName
string
pattern: ^.{0,45}$
example: John Doe
The full name of the card holder, used in 3DS2 authentication. Not required if the cardHolderName will be supplied on the 3DS2 transaction using the cardToken generated from this request. Note: When testing in the sandbox environment, it is the cardHolderName that is used to determine the test card for 3D Secure 2 authentication.

disableNetworkTokenisation
boolean
default: false
example: true
(Optional) Set to true to specify that network tokenisation should not be used for this transaction, even if network token registration has been enabled on the account.

cardAddress
cardAddressRequestAttributes {
description: Card holder address.

address1
string
pattern: ^[^<>\\]+$
example: CardHolder House
The first line of the cardholder's billing address. Mastercard recommends providing this field for Mastercard 3DS-authenticated transactions from 1 July 2026. Supplying the billing address may improve issuer authentication and authorisation decisions. Max 50 characters for a 3DS2 enabled API token.

address2
string
pattern: ^[^<>\\]+$
example: 1 CardHolder Street
Second line of the card holder's address. Max 50 characters for a 3DS2 enabled API token.

town
string
pattern: ^[^<>\\]+$
example: CardHolder Town
Town of the card holder's address. Max 40 characters for a 3DS2 enabled API token.

postCode*
string
pattern: ^[0-9a-zA-Z -]$*
example: AB1 2CD
Postcode of the card holder's address.

state
string
pattern: ^[A-Z]{2}$
example: FL
This attribute can be used by merchants to provide state/province/territory ISO codes (2-char alpha) - specifically for countries where this is required as part of the address (Eg: US, Canada, India, China).

countryCode
string
pattern: ^\d{3}$
example: 826
ISO 3166-1 (3 digit) country code of the card holder's address.

}



```
}
```



saveCardRequestPan {

cardNumber*
string
example: 1234567890123456
REQUIRED. The unique number printed on a credit card (13 to 19 digits depending on card type). Submitted without whitespace or non-numeric characters.

expiryDate*
string
pattern: ^\d{2}(\/)?\d{2}$
example: 01/28
REQUIRED. The expiry date of the card in format MM/YY

startDate
string
pattern: ^\d{2}(\/)?\d{2}$
example: 07/20
For Maestro cards only, start date in format MM/YY

yourConsumerReference*
string
minLength: 1
maxLength: 50
example: 2b45fd3f-cee5-4e7e-874f-28051db65408
REQUIRED. Unique reference to anonymously identify your customer. Advisable to use GUIDs.

judoId
string
maxLength: 9
pattern: ^\d{9}$
example: 100100100
Unique merchant and/or location ID supplied by Judopay. Do not include spaces or dashes. If not specified, then your default judoId will be used.

currency
string
example: GBP
The currency to use for the pre-authorisation, only required if different from the currency set during onboarding (default is GBP)

cardHolderName
string
pattern: ^.{0,45}$
example: John Doe
The full name of the card holder, used in 3DS2 authentication. Not required if the cardHolderName will be supplied on the 3DS2 transaction using the cardToken generated from this request. Note: When testing in the sandbox environment, it is the cardHolderName that is used to determine the test card for 3D Secure 2 authentication.

disableNetworkTokenisation
boolean
default: false
example: true
(Optional) Set to true to specify that network tokenisation should not be used for this transaction, even if network token registration has been enabled on the account.

cardAddress
cardAddressRequestAttributes {

description: Card holder address.

address1
string
pattern: ^[^<>\\]+$
example: CardHolder House
The first line of the cardholder's billing address. Mastercard recommends providing this field for Mastercard 3DS-authenticated transactions from 1 July 2026. Supplying the billing address may improve issuer authentication and authorisation decisions. Max 50 characters for a 3DS2 enabled API token.

address2
string
pattern: ^[^<>\\]+$
example: 1 CardHolder Street
Second line of the card holder's address. Max 50 characters for a 3DS2 enabled API token.

town
string
pattern: ^[^<>\\]+$
example: CardHolder Town
Town of the card holder's address. Max 40 characters for a 3DS2 enabled API token.

postCode*
string
pattern: ^[0-9a-zA-Z -]$*
example: AB1 2CD
Postcode of the card holder's address



Postcode of the card holder's address.

state
string
pattern: ^[A-Z]{2}$
example: FL
This attribute can be used by merchants to provide state/province/territory ISO codes (2-char alpha) - specifically for countries where this is required as part of the address (Eg: US, Canada, India, China).

countryCode
string
pattern: ^\d{3}$
example: 826
ISO 3166-1 (3 digit) country code of the card holder's address.

}
}

scaExemptionRequest string

nullable: true
example: transactionRiskAnalysis

Indicates reason why challenge may not be necessary (this may be over-ruled by the issuer). Should not be specified in the same request as challengeRequestIndicator

Enum:

[trustedBeneficiary, transactionRiskAnalysis, dataShareOnly, scaAlreadyPerformed, null]

threeDSecureChallengeRequest {

acceptHeaders
string
default: text/html
example: text/html
Passed to ACS for use in rendering challenge screens.

userAgent
string
example: Mozilla/5.0 (iPhone; CPU iPhone OS 14_4 like Mac OS X) AppleWebKit/605.1.15 (KHTML, like Gecko) Mobile/15E148
Passed to ACS for use in rendering challenge screens.

}



threeDSecureCompletedResponse {

description: [Conditional] Only returned for transactions using 3DS

attempted boolean
example: true
Did the consumer attempt to authenticate through 3DSecure

result string
example: PASSED
[Conditional] Outcome of the 3DSecure authentication, only present on initial receipt response, not historic receipts

eci string
example: 05
Electronic Commerce Indicator returned by 3DS servers indicating outcome of authentication

challengeRequestIndicator string
nullable: true
example: ChallengeMandated
[Conditional] Challenge request indicator that was used for 3DS authentication

Enum:
[NoPreference, NoChallenge, ChallengePreferred, ChallengeAsMandate, ChallengeWithWhitelistPrompt, null]

scaExemption string
nullable: true
example:
[Conditional] SCA exemption that was used for 3DS authentication

Enum:
[TrustedBeneficiary, TransactionRiskAnalysis, DataShareOnly, ScaAlreadyPerformed, null]

}

3DS Challenge Required {

challengeUrl string
example: https://acs2.test.gpe.cz/tds/challenge/brw/ea510e00-90ab-41eb-8a54-482e20fc2a36
URL to direct consumer to to complete challenge

cReq string
example:
ewogICJtZXNzYWdlVHlwZSIgOiAiQ1JlcSIsCiAgIm1lc3NhZ2VWZXJzaW9uIiA6ICIyLjEuMCIsCiAgInRocmVlRFNTZXJ2ZXJUcmFuc0l
Encoded value that should be POSTed to the challengeUrl

version string
example: 2.2.0
ThreeDSecure version

receiptId string
example: 1001131610340495360
Judopay's reference for the transaction.

result string
example: Challenge completion is needed for 3D Secure 2
Result of transaction .

message string
example: Issuer ACS has responded with a Challenge URL
Message detailing the outcome.

md string
example: ewogICJ0aHJlZURTU2VydmVyVHJhbnNJRCIgOiAiOGU4ZDY2ZTctM2Y1Yy00MTQyLTk5ZTItY2Y4YWE4ODBhMTMxIiwKICAidG
Encoded string representing ThreeDSecure transaction details.

}



3DS Device Details Required {

methodUrl string
example: https://test.portal.gpwebpay.com/pay-sim/sim/acs
URL to direct consumer to to complete challenge

version string
example: 2.2.0
ThreeDSecure version

receiptId string
example: 1001131610340495360
Judopay's reference for the transaction.

result string
example: Additional device data is needed for 3D Secure 2
Result of transaction.

message string
example: Issuer ACS has requested additional device data gathering
Message detailing the outcome.

md string
example:
ewogICJ0aHJlZURTU2VydmVyVHJhbnNJRCIgOiAiOGU4ZDY2ZTctM2Y1Yy00MTQyLTk5ZTItY2Y4YWE4ODBhMTMxIiwKICAidGhyZWVEU01ldG
Encoded string representing ThreeDSecure transaction details.

}



threeDSecureTwoCheckCardRequest {

cardHolderName
string
pattern: ^.{0,45}$
example: John Doe
The cardholder's full name. Mastercard recommends providing this field for Mastercard 3DS-authenticated transactions from 1 July 2026. Supplying the cardholder name may improve issuer authentication and authorisation decisions. Note: When testing in the sandbox environment, it is the cardHolderName that is used to determine the test card for 3D Secure 2 authentication.

emailAddress
string
pattern: ^.{0,254}$
example: test.user@judopay.com
The cardholder's email address. Mastercard recommends providing at least one contact method for Mastercard 3DS-authenticated transactions from 1 July 2026. A contact method can be an email address or supported telephone number. Required for Visa Network Token transactions to allow successful token provisioning.

shippingAddress
{
description: (Optional) Shipping address

address1
string
example: CardHolder House
First line of the shipping address.

address2
string
example: 1 CardHolder Street
Second line of the shipping address.

town
string
example: CardHolder Town
Town of the shipping address.

postCode
string
example: AB1 2CD
Postcode of the shipping address.

countryCode
string
pattern: ^\d{3}$
example: 826
ISO 3166-1 (3 digit) country code of the shipping address.

isBillingAddress
boolean
example: true
Indicator of whether the billing address is the same as the shipping address.

}

mobileNumber
string
pattern: ^\d{0,15}$
example: 7999999999
The cardholder's mobile telephone number. Mastercard recommends providing at least one contact method for Mastercard 3DS-authenticated transactions from 1 July 2026. A contact method can be an email address or supported telephone number. Must be set if phoneCountryCode is set.

phoneCountryCode
string
default: 44
pattern: ^\d{0,3}$
example: 44
The international dialling code associated with the mobile telephone number. Should be set if mobileNumber is set, defaulted to 44 if not.

threeDSecure
{
authenticationSource
string
example: Browser
Type of device that will be used for challenge authentication.

Enum:
[Browser]

methodNotificationUrl
string
default: https://api.judopay.com/order/3ds/methodNotification
example: https://api.judopay.com/order/3ds/methodNotification
URL of the page that will receive a POST event after a device details call is completed. Example is a Judo-hosted page that triggers a call to POST /transactions/{receiptId}/resume3ds (which will be used if this attribute is not set)



challengeNotificationUrl ▾ string
default: https://api.judopay.com/order/3ds/challengeNotification
example: https://api.judopay.com/order/3ds/challengeNotification
URL of the page that will receive a POST event after a challenge is completed. Example is a Judo-hosted page that triggers a call to POST /transactions/{receiptId}/complete3ds (which will be used if this attribute is not set)

}

threeDSecureMpi ▾ {

description: Optional block to pass in 3DS2 authentication results performed outside of Judopay.

dsTransId* ▾ string
minLength: 36
maxLength: 64
pattern: ^[0-9a-zA-Z-]$*
example: 41aadf4c-e73f-4bd1-a0c4-acb2615a32af
REQUIRED. Unique identifier for the authentication assigned by the Directory Server (Card Scheme).

cavv* ▾ string
minLength: 28
maxLength: 28
pattern: ^[a-zA-Z0-9+/=]$*
example: AJkBCWUyZwAAAABugmKTdAAAAAA=
REQUIRED. Unique value for the authentication provided by the ACS. May be used to provide proof of authentication.

eci* ▾ string
minLength: 2
maxLength: 2
pattern: ^\d{2}$
example: 02
REQUIRED. Electronic Commerce Indicator (ECI) is the value returned by Directory Servers (namely Visa, MasterCard, JCB, and American Express) indicating the outcome of authentication attempted on transactions enforced by 3DS.

threeDSecureVersion* ▾ string
minLength: 5
example: 2.2.0
REQUIRED. Version of 3DSecure that the card is enrolled into.

}

}



threeDSecureTwoRequest {

cardHolderName
string
pattern: ^.{0,45}$
example: John Doe
The cardholder's full name. Mastercard recommends providing this field for Mastercard 3DS-authenticated transactions from 1 July 2026. Supplying the cardholder name may improve issuer authentication and authorisation decisions. Note: When testing in the sandbox environment, it is the cardHolderName that is used to determine the test card for 3D Secure 2 authentication.

emailAddress
string
pattern: ^.{0,254}$
example: test.user@judopay.com
The cardholder's email address. Mastercard recommends providing at least one contact method for Mastercard 3DS-authenticated transactions from 1 July 2026. A contact method can be an email address or supported telephone number. Required for Visa Network Token transactions to allow successful token provisioning.

shippingAddress
{
description: (Optional) Shipping address

address1
string
example: CardHolder House
First line of the shipping address.

address2
string
example: 1 CardHolder Street
Second line of the shipping address.

town
string
example: CardHolder Town
Town of the shipping address.

postCode
string
example: AB1 2CD
Postcode of the shipping address.

countryCode
string
pattern: ^\d{3}$
example: 826
ISO 3166-1 (3 digit) country code of the shipping address.

isBillingAddress
boolean
example: true
Indicator of whether the billing address is the same as the shipping address.

}

mobileNumber
string
pattern: ^\d{0,15}$
example: 7999999999
The cardholder's mobile telephone number. Mastercard recommends providing at least one contact method for Mastercard 3DS-authenticated transactions from 1 July 2026. A contact method can be an email address or supported telephone number. Must be set if phoneCountryCode is set.

phoneCountryCode
string
default: 44
pattern: ^\d{0,3}$
example: 44
The international dialling code associated with the mobile telephone number. Should be set if mobileNumber is set, defaulted to 44 if not.

threeDSecure
{
authenticationSource
string
example: Browser
Type of device that will be used for challenge authentication.

Enum:
[Browser]

methodNotificationUrl
string
default: https://api.judopay.com/order/3ds/methodNotification
example: https://api.judopay.com/order/3ds/methodNotification
URL of the page that will receive a POST event after a device details call is completed. Example is a Judo-hosted page that triggers a call to POST /transactions/{receiptId}/resume3ds (which will be used if this attribute is not set)



challengeNotificationUrl
string
default: https://api.judopay.com/order/3ds/challengeNotification
example: https://api.judopay.com/order/3ds/challengeNotification
URL of the page that will receive a POST event after a challenge is completed. Example is a Judo-hosted page that triggers a call to POST /transactions/{receiptId}/complete3ds (which will be used if this attribute is not set)

challengeRequestIndicator
challengeRequestIndicatorRequest
challengeRequestIndicatorRequest string
nullable: true
default: noPreference
example: challengePreferred
Indicates whether a challenge is requested for this transaction (this may be over-ruled by the issuer). Should not be specified in the same request as scaExemption

Enum:
[noPreference, noChallenge, challengePreferred, challengeAsMandate, challengeWithWhitelistPrompt, null]

scaExemption
scaExemptionRequest scaExemptionRequest string
nullable: true
example: transactionRiskAnalysis
Indicates reason why challenge may not be necessary (this may be over-ruled by the issuer). Should not be specified in the same request as challengeRequestIndicator

Enum:
[trustedBeneficiary, transactionRiskAnalysis, dataShareOnly, scaAlreadyPerformed, null]

}

threeDSecureMpi
{
description: Optional block to pass in 3DS2 authentication results performed outside of Judopay.

dsTransId*
string
minLength: 36
maxLength: 64
pattern: ^[0-9a-zA-Z-]$*
example: 41aadf4c-e73f-4bd1-a0c4-acb2615a32af
REQUIRED. Unique identifier for the authentication assigned by the Directory Server (Card Scheme).

cavv*
string
minLength: 28
maxLength: 28
pattern: ^[a-zA-Z0-9+/=]$*
example: AJkBCWUyZwAAAABugmKTdAAAAAA=
REQUIRED. Unique value for the authentication provided by the ACS. May be used to provide proof of authentication.

eci*
string
minLength: 2
maxLength: 2
pattern: ^\d{2}$
example: 02
REQUIRED. Electronic Commerce Indicator (ECI) is the value returned by Directory Servers (namely Visa, MasterCard, JCB, and American Express) indicating the outcome of authentication attempted on transactions enforced by 3DS.

threeDSecureVersion*
string
minLength: 5
example: 2.2.0
REQUIRED. Version of 3DSecure that the card is enrolled into.

}
}



Receipt {

receiptId
string
example: 1001131610340495360
Judopay's reference for the transaction.

originalReceiptId
string
example: 1001124307998347264
[Conditional] Judopay's reference for any linked transaction (for collections, refunds and voids)

yourPaymentReference
string
example: 6482c678-cad3-4efd-b081-aeae7a89a134
Your unique reference for this payment.

type
string
example: PreAuth
The type of the transaction.

Enum:
[Payment, Refund, PreAuth, VOID, Collection, Save, Register, CheckCard]

createdAt
string($date-time)
example: 2025-02-05T16:28:32.8596+00:00
Date and time of creation.

result
string
example: Success
Result of transaction.

message
string
example: AuthCode: 123456
Message detailing the outcome.

judoId
integer
example: 100100100
Unique merchant and/or location ID supplied by Judopay.

merchantName
string
example: Test Merchant
Merchant's trading name.

appearsOnStatementAs
string
example: APL/TestMerchant*
Merchant description as it appears on the consumer's statement.

originalAmount
string
example: 10.99
Amount of original transaction (not affected by refunds or collections).

amountCollected
string
example: 1.99
[Conditional - PreAuth and Void only] Amount that has been collected on original pre-authorisation.

netAmount
string
example: 1.00
For payments, preauths and voids this is equal to the amount on the request. For incremental auths this is the original preauth amount plus the total of all incremental auths. For collections, this is the original preauth amount plus the total of all incremental auths minus the total of all collections. For refunds, this is the original sale amount minus the total of all refunds.

amount
string
example: 1.00
The amount of this transaction, for refunds and collections the amount that has been refunded or collected in this transaction.

currency
string
example: GBP
The ISO-217 alphabetic code of the currency of the transaction

recurringPaymentType
string
example: MIT
[Conditional] Type of recurring payment (RECURRING for scheduled, MIT for non-scheduled)

acquirerTransactionId
string



example: 33666277256892153705
The unique ID of the transaction set by the acquirer.

externalBankResponseCode string
example: A
Response code set by the acquirer.

authCode string
example: 123456
Authorisation code set by acquirer

walletType integer
example: 1
[Conditional] For digital wallet transactions, the type of wallet (1=ApplePay, 4=GooglePay, 5=ClickToPay)

riskScore integer
minimum: 0
maximum: 100
example: 0
Consumers risk score

paymentNetworkTransactionId string
example: 123456789012345
[Conditional] Unique identifier of the transaction returned by the payment network/ card scheme.

allowIncrement boolean
example: true
[Conditional] true for customer initiated pre-auths that had the allowIncrement request attribute set to true

isIncrementalAuth boolean
example: true
[Conditional] true for calls to POST /transactions/incrementalAuth

emailAddress string
example: test.user@judopay.com
[Conditional] email address of consumer, if supplied on the request

cardDetails cardDetailsResponse {

- **cardLastfour** string
  example: 1234
  Last four digits of card
- **endDate** string
  example: 1225
  Card expiry date in format MMYY
- **cardToken** string
  example: Ck3AeNlBfjvs9d61MNZiG0gtCvijqvKr
  Tokenized representation of card which can be used in future transactions instead of card number
- **cardType** integer
  example: 2
  Integer representing the type of card, see Reference Codes - Card Types on https://docs.judopay.com/Content/Developer%20Tools/Codes.htm#CardTypes
- **startDate** string
  example: 0121
  Card start date in format MMYY
- **cardScheme** string
  example: Mastercard
  Name of card scheme (determined from matching start of card number against issuer data)
- **cardFunding** string
  example: Credit
  Type of card (determined from matching start of card number against issuer data)
- **cardCategory** string
  example: Acquirer Only
  Category of card (determined from matching start of card



category of card (determined from matching start of card number against issuer data)

cardCountry
string
example: GB
ISO 3166-1 alpha-2 code of country that issued the card (determined from matching start of card number against issuer data)

bank
string
example: Santander Uk Plc
Name of bank that issued the card (determined from matching start of card number against issuer data)

cardHolderName
string
example: John Doe
Name of card holder as supplied on request

ownerType
string
example: Personal
[Conditional] Indicator of whether a card is personal or commercial, if known

Enum:
[Personal, Commercial, null]

}

billingAddress
cardAddressResponseAttributes {
description: Card holder address.

address1
string
pattern: ^[^<>\\]+$
example: CardHolder House
The first line of the cardholder's billing address. Mastercard recommends providing this field for Mastercard 3DS-authenticated transactions from 1 July 2026. Supplying the billing address may improve issuer authentication and authorisation decisions. Max 50 characters for a 3DS2 enabled API token.

address2
string
pattern: ^[^<>\\]+$
example: 1 CardHolder Street
Second line of the card holder's address. Max 50 characters for a 3DS2 enabled API token.

town
string
pattern: ^[^<>\\]+$
example: CardHolder Town
Town of the card holder's address. Max 40 characters for a 3DS2 enabled API token.

postCode*
string
pattern: ^[0-9a-zA-Z -]$*
example: AB1 2CD
Postcode of the card holder's address.

state
string
pattern: ^[A-Z]{2}$
example: FL
This attribute can be used by merchants to provide state/province/territory ISO codes (2-char alpha) - specifically for countries where this is required as part of the address (Eg: US, Canada, India, China).

countryCode
integer
pattern: ^\d{2,3}$
example: 826
ISO 3166-1 (3 digit) country code of the card holder's address. If the first digit of the of the code is 0 this will not be shown

}

consumer
consumerResponse {
yourConsumerReference
string
example: 2b45fd3f-cee5-4e7e-874f-28051db65408
Your unique reference to anonymously identify your customer.

}

device
deviceResponse {



deviceResponse {

description: [Conditional] If clientDetails was provided containing a kdeviceid

identifier: string
example: d73b4a7b58ce4e54a3bd73b7eda061e6
Judopay generated GUID associated with the kdeviceid passed in the clientDetails request attribute

}

yourPaymentMetaData {

description: Merchant metadata passed with transaction request.

}
example: { "internalLocationRef": "Example", "internalId": 99 }

threeDSecure threeDSecureCompletedResponse {

description: [Conditional] Only returned for transactions using 3DS

attempted: boolean
example: true
Did the consumer attempt to authenticate through 3DSecure

result: string
example: PASSED
[Conditional] Outcome of the 3DSecure authentication, only present on initial receipt response, not historic receipts

eci: string
example: 05
Electronic Commerce Indicator returned by 3DS servers indicating outcome of authentication

challengeRequestIndicator: string
nullable: true
example: ChallengeMandated
[Conditional] Challenge request indicator that was used for 3DS authentication

Enum:
[NoPreference, NoChallenge, ChallengePreferred, ChallengeAsMandate, ChallengeWithWhitelistPrompt, null]

scaExemption: string
nullable: true
example:
[Conditional] SCA exemption that was used for 3DS authentication

Enum:
[TrustedBeneficiary, TransactionRiskAnalysis, DataShareOnly, ScaAlreadyPerformed, null]

}

risks riskParametersResponse {

postCodeCheck: string
example: PASSED
Indication of whether supplied postCode was verified

Enum:
[PASSED, FAILED, UNKNOWN]

cv2Check: string
example: PASSED
Indication of whether supplied cv2 was verified

Enum:
[PASSED, FAILED, NOT_PROCESSED, NOT_SUBMITTED, NOT_SUPPORTED, NOT_CHECKED, SUSPICIOUS, UNKNOWN]

merchantSuggestion: string
example: Allow
Suggestion action for the merchant from the risk engine

}

disableNetworkTokenisation boolean
example: true
[Conditional] true if the request explicitly disabled network tokens for this transaction

networkTokenisationDetails networkTokenisationDetailsResponse {



networkTokenProvisioned
boolean
example: true
Indicates if a new network token was created during the course of this transaction

networkTokenUsed
boolean
example: true
Indicates if this transaction was completed using a network token

virtualPan
{
lastFour
string
example: 1234
Last four digits of the virtual PAN used

expiryDate
string
example: 0129
Expiry date of the virtual PAN in format MMYY

}
}
}



transactionReceiptHistoricResponse ⌄ {

receiptId
⌄ string
example: 1001131610340495360
Judopay's reference for the transaction.

originalReceiptId
⌄ string
example: 1001124307998347264
[Conditional] Judopay's reference for any linked transaction (for collections, refunds and voids)

yourPaymentReference
⌄ string
example: 6482c678-cad3-4efd-b081-aeae7a89a134
Your unique reference for this payment.

type
⌄ string
example: PreAuth
The type of the transaction.

Enum:
⌄ [Payment, Refund, PreAuth, VOID, Collection, Save, Register, CheckCard]

createdAt
⌄ string($date-time)
example: 2025-02-05T16:28:32.8596+00:00
Date and time of creation.

result
⌄ string
example: Success
Result of transaction.

message
⌄ string
example: AuthCode: 123456
Message detailing the outcome.

judoId
⌄ integer
example: 100100100
Unique merchant and/or location ID supplied by Judopay.

merchantName
⌄ string
example: Test Merchant
Merchant's trading name.

appearsOnStatementAs
⌄ string
example: APL/TestMerchant*
Merchant description as it appears on the consumer's statement.

originalAmount
⌄ string
example: 10.99
Amount of original transaction (not affected by refunds or collections).

amountCollected
⌄ string
example: 1.99
[Conditional - PreAuth and Register only] Amount that has been collected on original pre-authorisation.

netAmount
⌄ string
example: 1.00
For payments and refunds, this is the original sale amount mines the total of all refunds. For preauths, incremental auths and collections, this is the original preauth amount plus the total of all incremental auths minus the total of all collections. For voids, this is equal to the original preauth amount.

amount
⌄ string
example: 1.00
The amount of this transaction, for refunds and collections the amount that has been refunded or collected in this transaction.

currency
⌄ string
example: GBP
The ISO-217 alphabetic code of the currency of the transaction

recurringPaymentType
⌄ string
example: MIT
[Conditional] Type of recurring payment (RECURRING for scheduled, MIT for non-scheduled)

acquirerTransactionId
⌄ string
example: 33666277256892153705



The unique ID of the transaction set by the acquirer.

externalBankResponseCode — string
example: A
Response code set by the acquirer.

authCode — string
example: 123456
Authorisation code set by acquirer

postCodeCheckResult — string
example: Passed
Show results of post code checks

walletType — integer
example: 1
[Conditional] For digital wallet transactions, the type of wallet (1=ApplePay, 4=GooglePay, 5=ClickToPay)

acquirer — string
example: MyBank
Name of acquirer

webPaymentReference — string
example: 5QcAAAMAAAASAAAADAAAAHaKm0p0p-Ew6VrBhPxjxvMLinJOMZEJWG_K7kJ53UdvRGQ3qA
[Conditional] Judo identifier for associated web payment reference (if the transaction was associated with a payment session)

noOfAuthAttempts — integer
example: 1
[Conditional] This indicates the number of authentication attempts made for transactions using the associated payment session reference.

paymentNetworkTransactionId — string
example: 123456789012345
[Conditional] Unique identifier of the transaction returned by the payment network/ card scheme.

allowIncrement — boolean
example: true
[Conditional] true for customer initiated pre-auths that had the allowIncrement request attribute set to true

isIncrementalAuth — boolean
example: true
[Conditional] true for calls to POST /transactions/incrementalAuth

disableNetworkTokenisation — boolean
example: true
[Conditional] true if the request explicitly disabled network tokens for this transaction

cardDetails — cardDetailsResponse {

- **cardLastfour** — string
  example: 1234
  Last four digits of card
- **endDate** — string
  example: 1225
  Card expiry date in format MMYY
- **cardToken** — string
  example: Ck3AeNlBfjvs9d61MNZiG0gtCvijqvKr
  Tokenized representation of card which can be used in future transactions instead of card number
- **cardType** — integer
  example: 2
  Integer representing the type of card, see Reference Codes - Card Types on https://docs.judopay.com/Content/Developer%20Tools/Codes.htm#CardTypes
- **startDate** — string
  example: 0121
  Card start date in format MMYY
- **cardScheme** — string



example: Mastercard
Name of card scheme (determined from matching start of card number against issuer data)

cardFunding
string
example: Credit
Type of card (determined from matching start of card number against issuer data)

cardCategory
string
example: Acquirer Only
Category of card (determined from matching start of card number against issuer data)

cardCountry
string
example: GB
ISO 3166-1 alpha-2 code of country that issued the card (determined from matching start of card number against issuer data)

bank
string
example: Santander Uk Plc
Name of bank that issued the card (determined from matching start of card number against issuer data)

cardHolderName
string
example: John Doe
Name of card holder as supplied on request

ownerType
string
example: Personal
[Conditional] Indicator of whether a card is personal or commercial, if known

Enum:
[Personal, Commercial, null]

}

billingAddress
cardAddressResponseAttributes {
description: Card holder address.

address1
string
pattern: ^[^<>\\]+$
example: CardHolder House
The first line of the cardholder's billing address. Mastercard recommends providing this field for Mastercard 3DS-authenticated transactions from 1 July 2026. Supplying the billing address may improve issuer authentication and authorisation decisions. Max 50 characters for a 3DS2 enabled API token.

address2
string
pattern: ^[^<>\\]+$
example: 1 CardHolder Street
Second line of the card holder's address. Max 50 characters for a 3DS2 enabled API token.

town
string
pattern: ^[^<>\\]+$
example: CardHolder Town
Town of the card holder's address. Max 40 characters for a 3DS2 enabled API token.

postCode*
string
pattern: ^[0-9a-zA-Z -]$*
example: AB1 2CD
Postcode of the card holder's address.

state
string
pattern: ^[A-Z]{2}$
example: FL
This attribute can be used by merchants to provide state/province/territory ISO codes (2-char alpha) - specifically for countries where this is required as part of the address (Eg: US, Canada, India, China).

countryCode
integer
pattern: ^\d{2,3}$
example: 826
ISO 3166-1 (3 digit) country code of the card holder's



address. If the first digit of the of the code is 0 this will not be shown

}

consumer

consumerResponse {

yourConsumerReference

string

example: 2b45fd3f-cee5-4e7e-874f-28051db65408

Your unique reference to anonymously identify your customer.

}

device

deviceResponse {

description: [Conditional] If clientDetails was provided containing a kdeviceid

identifier

string

example: d73b4a7b58ce4e54a3bd73b7eda061e6

Judopay generated GUID associated with the kdeviceid passed in the clientDetails request attribute

}

yourPaymentMetaData

{

description: Merchant metadata passed with transaction request.

}

example: { "internalLocationRef": "Example", "internalId": 99 }

threeDSecure

threeDSecureCompletedResponse {

description: [Conditional] Only returned for transactions using 3DS

attempted

boolean

example: true

Did the consumer attempt to authenticate through 3DSecure

result

string

example: PASSED

[Conditional] Outcome of the 3DSecure authentication, only present on initial receipt response, not historic receipts

eci

string

example: 05

Electronic Commerce Indicator returned by 3DS servers indicating outcome of authentication

challengeRequestIndicator

string

nullable: true

example: ChallengeMandated

[Conditional] Challenge request indicator that was used for 3DS authentication

Enum:

[NoPreference, NoChallenge, ChallengePreferred, ChallengeAsMandate, ChallengeWithWhitelistPrompt, null]

scaExemption

string

nullable: true

example:

[Conditional] SCA exemption that was used for 3DS authentication

Enum:

[TrustedBeneficiary, TransactionRiskAnalysis, DataShareOnly, ScaAlreadyPerformed, null]

}

risks

riskParametersResponse {

postCodeCheck

string

example: PASSED

Indication of whether supplied postCode was verified

Enum:

[PASSED, FAILED, UNKNOWN]

cv2Check

string

example: PASSED

Indication of whether supplied cv2 was verified

Enum:

[PASSED, FAILED, NOT_PROCESSED, NOT_SUBMITTED, NOT SUPPORTED. NOT CHECKED. SUSPICIOUS. UNKNOWN]



```
                merchantSuggestion                string
                                                  example: Allow
                                                  Suggestion action for the merchant from the risk engine
            }
    networkTokenisationDetails  networkTokenisationDetailsResponse {
                networkTokenProvisioned           boolean
                                                  example: true
                                                  Indicates if a new network token was created during the
                                                  course of this transaction
                networkTokenUsed                  boolean
                                                  example: true
                                                  Indicates if this transaction was completed using a network
                                                  token
                virtualPan                        {
                  lastFour                          string
                                                    example: 1234
                                                    Last four digits of the virtual
                                                    PAN used
                  expiryDate                        string
                                                    example: 0129
                                                    Expiry date of the virtual PAN in
                                                    format MMYY
                                                  }
            }
}
```



transactionReceiptListResponse {

receiptId
string
example: 1001131610340495360
Judopay's reference for the transaction.

originalReceiptId
string
example: 1001124307998347264
[Conditional] Judopay's reference for any linked transaction (for collections, refunds and voids)

yourPaymentReference
string
example: 6482c678-cad3-4efd-b081-aeae7a89a134
Your unique reference for this payment.

type
string
example: PreAuth
The type of the transaction.

Enum:
[Payment, Refund, PreAuth, VOID, Collection, Save, Register, CheckCard]

createdAt
string($date-time)
example: 2025-02-05T16:28:32.8596+00:00
Date and time of creation.

result
string
example: Success
Result of transaction.

message
string
example: AuthCode: 123456
Message detailing the outcome.

judoId
integer
example: 100100100
Unique merchant and/or location ID supplied by Judopay.

appearsOnStatementAs
string
example: APL/TestMerchant*
Merchant description as it appears on the consumer's statement.

originalAmount
string
example: 10.99
Amount of original transaction (not affected by refunds or collections).

amountCollected
string
example: 1.99
[Conditional - PreAuth and Register only] Amount that has been collected on original pre-authorisation.

netAmount
string
example: 1.00
For payments and refunds, this is the original sale amount mines the total of all refunds. For preauths, incremental auths and collections, this is the original preauth amount plus the total of all incremental auths minus the total of all collections. For voids, this is equal to the original preauth amount.

amount
string
example: 1.00
The amount of this transaction, for refunds and collections the amount that has been refunded or collected in this transaction.

currency
string
example: GBP
The ISO-217 alphabetic code of the currency of the transaction

postCodeCheckResult
string
example: Passed
Show results of post code checks

walletType
integer
example: 1
[Conditional] For digital wallet transactions, the type of wallet (1=ApplePay, 4=GooglePay, 5=ClickToPay)

allowIncrement
boolean



example: true
[Conditional] true for customer initiated pre-auths that had the allowIncrement request attribute set to true

isIncrementalAuth
boolean
example: true
[Conditional] true for calls to POST /transactions/incrementalAuth

cardDetails
cardDetailsListResponse **{**
cardLastfour **string**
example: 1234
Last four digits of card
endDate **string**
example: 1225
Card expiry date in format MMYY
cardType **integer**
example: 2
Integer representing the type of card, see Reference Codes - Card Types on https://docs.judopay.com/Content/Developer%20Tools/Codes.htm#CardTypes
}

consumer
consumerResponse **{**
yourConsumerReference **string**
example: 2b45fd3f-cee5-4e7e-874f-28051db65408
Your unique reference to anonymously identify your customer.
}

yourPaymentMetaData
{
description: Merchant metadata passed with transaction request.
}
example: { "internalLocationRef": "Example", "internalId": 99 }

threeDSecure
threeDSecureCompletedResponse **{**
description: [Conditional] Only returned for transactions using 3DS
attempted **boolean**
example: true
Did the consumer attempt to authenticate through 3DSecure
result **string**
example: PASSED
[Conditional] Outcome of the 3DSecure authentication, only present on initial receipt response, not historic receipts
eci **string**
example: 05
Electronic Commerce Indicator returned by 3DS servers indicating outcome of authentication
challengeRequestIndicator **string**
nullable: true
example: ChallengeMandated
[Conditional] Challenge request indicator that was used for 3DS authentication
Enum:
[NoPreference, NoChallenge, ChallengePreferred, ChallengeAsMandate, ChallengeWithWhitelistPrompt, null]
scaExemption **string**
nullable: true
example:
[Conditional] SCA exemption that was used for 3DS authentication
Enum:
[TrustedBeneficiary, TransactionRiskAnalysis, DataShareOnly, ScaAlreadyPerformed, null]
}

disableNetworkTokenisation
boolean
example: true
[Conditional] true if the request explicitly disabled network tokens for this transaction



networkTokenisationDetails **networkTokenisationDetailsResponse** {

networkTokenProvisioned boolean
example: true
Indicates if a new network token was created during the course of this transaction

networkTokenUsed boolean
example: true
Indicates if this transaction was completed using a network token

virtualPan {
lastFour string
example: 1234
Last four digits of the virtual PAN used

expiryDate string
example: 0129
Expiry date of the virtual PAN in format MMYY

}

}

}

voidRequest {

receiptId* string
example: 1001131610340495360
REQUIRED. Judopay's reference for the pre-authorisation transaction that is to be voided.

amount number($float)
minimum: 0.01
example: 10.99
The amount to void, must match the amount of the original pre-authorisation. Format for default currencies is to two decimal places. For currencies using a different structure please contact Judopay for support. If not specified, the pre-authorised amount will be voided.

yourPaymentReference string
example: aa648425-2230-4e4e-8a61-27f06df54542
Your unique reference for this void (not the yourPaymentReference of the original transaction)

}



2. Test

Random text ANdrei



3. Untitled

Notes (important pour Archbee)

- accepte une **requête GET avec un corps JSON**, c'est pourquoi -Méthode GET et -Corps sont tous deux utilisés.
- ConvertTo-Json est nécessaire pour que PowerShell envoie une charge utile JSON appropriée.
- La réponse inclura généralement un **lien de téléchargement** lorsque exportAsLink = true.

Si vous le souhaitez, je peux aussi :

c'est un titre h1

- Convertir cela en **POST** (au cas où Archbee changerait de comportement)
- Ajouter **la gestion des erreurs**
- Enregistrer le lien d'exportation directement dans un fichier ou déclencher un téléchargement

```
$headers = @{
   "Accept"        = "application/json"
   "Content-Type"  = "application/json"
   "Authorization" = "Bearer SFFfdkhvMEJSLVN2RVRvYXowMkROfkp2NVZwUW5iWXB0VF9XUkxscWZrLQ=="
}

$body = @{
   teamId = "511c8QBH-VHiwWnyzIwUb"
   exportThisSpaceOnly = $true
   exportAsLink = $true
} | ConvertTo-Json

$response = Invoke-RestMethod `
   -Uri $uri `
   -Method GET `
   -Headers $headers `
   -Body $body
```



4. c'est un titre h1

asdasdasdasd

asdasdasdasdasasdsadas

asdasdasdasdasdas

asta sont du contenu à l'intérieur si e prins

asdasdasdas

dasdasdas

dasdasdsadasd

comme

Titre 1

asdasdasd

asdasdasdasd

sadas

Titre 2

asdasdasd

asdas

dasdsa

deuxième

asta e deschis

Ceci est l'en-tête	Ceci est l'en-tête 2	Ceci est l'en-tête 3
asdasdas	asddasd	sadasd
asdasdas	asdasdas	asdas
asdasd	asdasd	asdas

asdasdasdasd

asdasdasdasdasasdsadas



asdasdasdasdasdas

le contenu à l'intérieur est pris

deuxième

c'est ouvert



5. Baguette parce que car cela ouais

Frenchtech autre comme même vin baguette parce que car cela ouais. Guillotine car pour ouais dernier parce que. Avec camembert épicé.

Le client est très important merci, le client sera suivi par le client. Énée n'a pas de justice, pas de résultat, pas de ligula, et la vallée veut la sauce. Morbi mais qui veut vendre une couche de contenu triste d'internet. Être ivre maintenant, mais ne pas être ivre maintenant, mon urne est d'une grande beauté, mais elle n'est pas aussi bien faite que dans un livre. Mécène dans la vallée de l'orc, dans l'élément même. Certaines des exigences faciles du budget, qu'il soit beaucoup de temps pour dignissim et. Je ne m'en fais pas chez moi, ça va être moche dans le vestibule. Mais aussi des protéines de Pour avant la fin de la semaine, qui connaît le poison, le résultat.

le client sera suivi par le client. Énée n'a pas de justice, pas de résultat, pas de ligula, et la vallée veut la sauce. M dasdasdas Archbee Être ivre maintenant, mais ne pas être ivre maintenant, mon urne est d'une grande beauté, mais elle n'est pas aussi bien faite que dans un livre. Mécène dans la vallée de l'orc, dans l'élément même. Certaines des exigences faciles du budget, qu'il soit beaucoup de temps pour dignissim variabila

asdasdasd asdasdsadsa



5.1. Test doc

Nabucodonosor

asdasdas

1.0.7 OAS 2.0

Swagger Petstore

This is a sample server Petstore server. You can find out more about Swagger at http://swagger.io or on irc.freenode.net, #swagger. For this sample, you can use the api key `special-key` to test the authorization filters.

Terms of service
Contact the developer
Apache 2.0
Find out more about Swagger

Schemes

HTTPS

Authorize

pet Everything about your Pets — Find out more

POST `/pet/{petId}/uploadImage` uploads an image

POST `/pet` Add a new pet to the store

PUT `/pet` Update an existing pet

GET `/pet/findByStatus` Finds Pets by status



GET ~~/pet/findByTags~~ Finds Pets by tags

Warning: Deprecated

GET /pet/{petId} Find pet by ID

POST /pet/{petId} Updates a pet in the store with form data

DELETE /pet/{petId} Deletes a pet

store Access to Petstore orders

GET /store/inventory Returns pet inventories by status



POST /store/order Place an order for a pet

GET /store/order/{orderId} Find purchase order by ID

DELETE /store/order/{orderId} Delete purchase order by ID

user Operations about user

Find out more about our store

POST /user/createWithList Creates list of users with given input array

GET /user/{username} Get user by user name



PUT **/user/{username}** Updated user

DELETE **/user/{username}** Delete user

GET **/user/login** Logs user into the system

GET **/user/logout** Logs out current logged in user session

POST **/user/createWithArray** Creates list of users with given input array

POST **/user** Create user



Models

ApiResponse {
- code — integer($int32)
- type — string
- message — string

}

Category {
- id — integer($int64)
- name — string

}

Pet {
- id — integer($int64)
- category — **Category** {
  - id — integer($int64)
  - name — string

  }
- name* — string
  example: doggie
- photoUrls* — [
  xml: { "wrapped": true }
  string
  xml: { "name": "photoUrl" }
  xml:
  name: photoUrl]
- tags — [
  xml: { "wrapped": true }
  Tag {
  - id — integer($int64)
  - name — string

  }]
- status — string
  pet status in the store
  Enum:
  [available, pending, sold]

}

Tag {
- id — integer($int64)
- name — string

}



Order {

id	integer($int64)
petId	integer($int64)
quantity	integer($int32)
shipDate	string($date-time)
status	string Order Status Enum: [placed, approved, delivered]
complete	boolean

}

User {

id	integer($int64)
username	string
firstName	string
lastName	string
email	string
password	string
phone	string
userStatus	integer($int32) User Status

}

> Bucuresti

Lorem Ipsum: est simplement un texte factice de l'industrie de l'impression et de la composition. Lorem Ipsum est le texte factice standard de l'industrie depuis les années 1500, lorsque un imprimeur inconnu a pris une galée de caractères et l'a mélangée pour créer un livre de spécimens typographiques. Il a survécu non seulement cinq siècles, mais aussi le saut vers la composition électronique, restant essentiellement inchangé. Il a été popularisé dans les années 1960 avec la sortie de feuilles Letraset contenant des passages de Lorem Ipsum, et plus récemment avec des logiciels de publication assistée par ordinateur comme Aldus PageMaker incluant des versions de Lorem Ipsum.

6. Indenting

6.1. Une autre page

asdasdsadasdas

6.1.1. Filtrage avancé

Dans la section "facettage", nous mettons en évidence comment on peut obtenir les filtres disponibles pour une requête de recherche et appliquer des filtres pour affiner les résultats de recherche. Par exemple, après avoir recherché "chaussures", un acheteur peut vouloir affiner les résultats en appliquant un filtre sur l'attribut couleur et en choisissant une valeur telle que rouge. Ce faisant, seules les chaussures "rouges" apparaîtront dans les résultats. Si à ce stade, l'acheteur choisit une autre couleur, par exemple, bleu, Klevu affichera maintenant à la fois les chaussures "rouges" et les "bleues". Ainsi, il appliquera l'opération OU entre les valeurs du même attribut. Si à ce stade, l'acheteur choisit "coton" comme matériau, le système affichera à la fois les chaussures rouges et bleues, mais il s'assurera également que tous les produits ont "coton" comme matériau. En d'autres termes, il applique l'opération ET entre les différents types d'attributs sélectionnés.

Cependant, il arrive un moment où l'on peut vouloir effectuer des opérations de filtrage complexes, ou en d'autres termes, des opérations de filtrage avancées. Par exemple, vous pouvez vouloir afficher un groupe de chaussures où "la couleur est rouge", et "la marque n'est PAS Adidas". Vous pouvez également vouloir ajouter un autre groupe de chaussures à cet ensemble de résultats. Par exemple, où "la couleur est bleue ou noire", "la marque est Nike", et "la fourchette de prix est entre 100 et 400". Comme vous pouvez le voir, contrairement au comportement du "facettage", dans notre exemple ici, nous souhaitons avoir un ou plusieurs groupes de produits correspondant à différents groupes de conditions pour l'ensemble final des produits à retourner.

C'est avec cette intention de permettre des opérations de filtrage avancées sur l'ensemble des résultats que nous avons introduit une nouvelle fonctionnalité appelée "filtrage avancé" dans notre API.

Avant l'introduction du filtrage avancé, si un utilisateur avait demandé que des filtres soient retournés ou que certains soient appliqués, Klevu cherchait à trouver un ensemble de produits pertinents correspondant à la requête de l'utilisateur. Il incluait/excluait ensuite tous les produits demandés explicitement dans cet ensemble de résultats. C'est après cette étape qu'il calcule tous les filtres possibles à retourner et applique également les filtres sélectionnés pour affiner les résultats de recherche. Ainsi, si un produit ajouté explicitement ne correspondait pas au filtre appliqué, il serait exclu de l'ensemble final des résultats.

Avec l'ajout du filtrage avancé, Klevu identifie maintenant un ensemble de produits pertinents correspondant à la requête. Il applique ensuite le filtrage avancé à cela. C'est après cette étape qu'il inclut/exclut tous les produits demandés explicitement, calcule les filtres basés sur cet ensemble de résultats, et enfin applique les filtres sélectionnés pour affiner les résultats de recherche. De cette manière, s'il y a un filtrage à appliquer avant d'inclure/exclure des produits explicitement, le filtrage avancé doit être utilisé.

> Veuillez noter que le filtrage avancé peut également être utilisé avec les API de marchandisage par catégorie et de recommandation.

Le **groupCondition** l'objet peut être utilisé pour appliquer les conditions de filtrage avancé afin que vos clients puissent affiner leurs résultats en fonction des attributs pertinents.

Apply Advanced Filtering

Example is as below:

POST https://{{hostname}}/cs/v2/search

Request

BODY PARAMETERS

groupOperator String optional

Defining the condition of filtering. Here are the options available.
ALL_OF:This is used to filter data using AND query i.e. the search results are matching with all the attributes.

ANY_OF:This is used to filter data using OR query i.e. the search results are matching with at least one attributes.

NONE_OF: This is used as negation AND query i.e. when search results are matching with none of the attribute values.

key String optional

The ID of the attribute to filter by, eg. color

valueOperator String optional

Defining an attribute's operation, Here are the options available.

INCLUDE : This is used to include specific attribute for the filter purpose.

EXCLUDE : This is used to exclude specific attribute for the filter purpose.

EXISTS: This is used to find if the specific indexed attribute value exists with product.

NOT_EXISTS : This is used to find if the specific indexed attribute value does not exist at all.

singleSelect Boolean optional

The behaviour when specifying multiple filters or values.

If it is set to "true", the products returned must have both the values.
If it is set to "false", the products returned must have at least one of the values.

excludeValuesInResult Boolean optional

For numeric filter like price, can be used to exclude data of the starting or ending values. For example if the range is defined as "1200 - 1232" in values, and excludeValuesInResult is set to true, Klevu will use range as “1201 - 1231“.

This can be used to achieve greater than or less than condition. For example,if you want to get the products greater than price 200, the values field will be “200 - *“ and the excludeValuesInResult will be set to true.

values `Array` `optional`

An array of values to filter by, eg. Red, Blue. When using range filters, specify the first value as the minimum value and the second as the max value. For example to retrieve records with prices between 60 to 80, use: "key": "klevu_price", "values": [60, 80].
To retrieve steps of prices, for example those with values between 0-50 or 150-200, use: "key": "klevu_price", "values": ["0-50", "150-200"].

To retrieve exact values, for example records with exact values 100 or 200, use: "key": "klevu_price", "values": ["100-100", "200-200"].
By default all attributes submitted to Klevu are indexed as STRING attributes, which means they cannot be used as range filters. The product sale price field is the only exception to this rule, which is filtered using the key klevu_price. If you have explicitly requested and Klevu has approved that certain attributes be indexed as numerical attributes, you can also retrieve those as range filters.

Request Body (JSON)

```
{
   "context":{
```



Responses 200

```
{
    "meta": {
        "qTime": 9,
        "responseCode": 200
    },
    "queryResults": [
        {
            "id": "productSearch",
            "meta": {
                "qTime": 7,
                "noOfResults": 2,
                "totalResultsFound": 6,
                "typeOfSearch": "WILDCARD_AND",
                "offset": 0,
                "debuggingInformation": {},
                "notificationCode": 1,
                "searchedTerm": "tee",
                "apiKey": "klevu-156925593843210765",
                "isPersonalised": false
            },
            "records": [
                {
                    "discount": "",
                    "hideGroupPrices": "",
                    "type": "Tees",
                    "itemGroupId": "4384027344958",
                    "freeShipping": "",
                    "storeBaseCurrency": "GBP",
                    "price": "24.00",
                    "toPrice": "",
                    "imageUrl": "https://cdn.shopify.com/s/files/1/0116/9457/1582/products/ws05-yellow_main_fe09e105-6b7e-4269-9
                    "currency": "GBP",
                    "inStock": "yes",
                    "id": "31366447038526",
                    "imageHover": "",
                    "sku": "WS05-XS-Yellow",
                    "brand": "KKE",
                    "basePrice": "24.0",
```



CodeSandbox

try_out_apis

Team Klevu

487.8k 0 21

Edit Sandbox

Files

- .codesandbox
- public
- src
  - App.js
  - index.js
  - payload.js
  - payloadList.js
  - styles.css
- package.json

Dependencies

https://4ktfrp.csb.app/?payload=advanceFiltering

Open Sandbox

Console 0 Problems 0 React DevTools 0

6.2. Nouvelle page

asdasdasdas



7. page de test

Obtenir des gâteaux

Obtenir un gâteau par son ID

GET https://api.cakes.com

Request

BODY PARAMETERS

id Array required

ID du gâteau à obtenir

Nom du paramètre ▶ Object optional

JS

```
var myHeaders = new Headers();
myHeaders.append("Accept", "application/json");
myHeaders.append("Content-Type", "application/json");

var raw = JSON.stringify({
   "id": "String"
});

var requestOptions = {
   method: 'GET',
   headers: myHeaders,
   body: raw,
   redirect: 'follow'
};

fetch("https://api.cakes.com", requestOptions)
   .then(response => response.text())
   .then(result => console.log(result))
   .catch(error => console.log('error', error));
```

Responses 200

```
{
  "name": "Nom du gâteau",
}
```

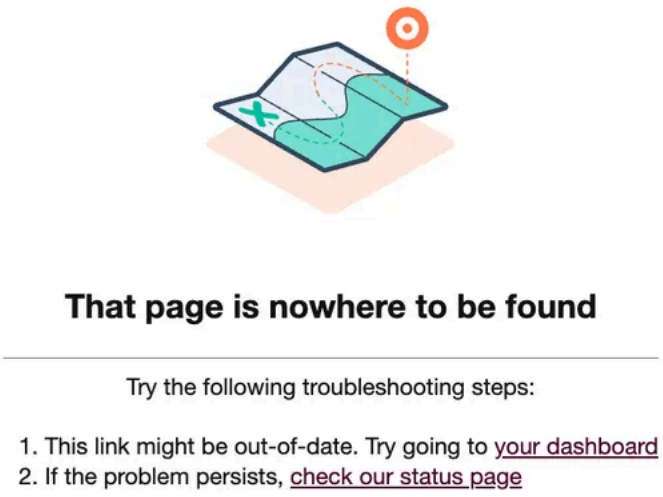
8. Reminder

8.1. Langage Toxique

Le garde-fou de langage toxique détecte le contenu nuisible, y compris les discours de haine, les menaces, les insultes et d'autres communications qui pourraient nuire à votre communauté ou à votre marque. Contrairement aux simples filtres de mots-clés, ce garde-fou utilise l'IA pour comprendre le contexte, le ton et l'intention.

knmknknkjn

Quand utiliser ce garde-fou



L'avantage clé par rapport au simple filtrage par mots-clés est que ce garde-fou comprend les nuances. Quelqu'un peut être hostile sans utiliser de grossièretés, et il peut utiliser un langage fort sans être hostile. Le garde-fou analyse l'intention et le ton, pas seulement le choix des mots.

Comprendre les niveaux de sensibilité

Le paramètre de sensibilité contrôle la rigueur de la barrière de protection. Pensez-y comme à l'ajustement du seuil de ce qui constitue une violation. C'est l'un des choix de configuration les plus importants que vous ferez car il change fondamentalement le

contenu qui passe ou échoue.

Une sensibilité faible ne signale que les violations graves comme les menaces explicites et les discours de haine extrêmes. Lorsque vous réglez la sensibilité sur faible, vous dites que vous vous attendez à des opinions fortes et à des désaccords robustes, et que vous ne voulez bloquer que le contenu qui franchit une ligne claire vers un territoire menaçant ou haineux. Par exemple, "Je suis fortement en désaccord avec cette approche et je pense qu'elle est mal orientée" passerait à faible sensibilité, tout comme "C'est une terrible idée." Seul un contenu comme "Je te trouverai et je te ferai du mal" ou un discours de haine explicite échouerait.

La sensibilité faible fonctionne bien pour les forums publics où le débat est attendu, les environnements de retour professionnel où la franchise est valorisée, et les communautés techniques où les gens discutent de sujets controversés. Le compromis est que certains contenus qui mettent les gens mal à l'aise pourraient encore passer.

La sensibilité moyenne est le paramètre par défaut et représente une approche équilibrée. À ce niveau, la barrière de protection signale les violations claires, y compris les insultes et le langage hostile, tout en permettant encore le désaccord professionnel et la critique constructive. Un message comme "Je ne suis pas d'accord avec votre raisonnement" passerait, mais "Tu es un idiot" ou "Des gens comme vous sont le problème" échoueraient.

La sensibilité moyenne fonctionne bien pour la plupart des applications, y compris les systèmes de service client, les communications professionnelles, les outils collaboratifs et les plateformes sociales. Elle trouve un équilibre entre la possibilité d'un discours significatif et le maintien d'un environnement respectueux.

La haute sensibilité crée l'environnement le plus strict en signalant tout contenu potentiellement toxique, y compris la légèreté de la grossièreté et le langage désinvolte. À ce niveau, même un contenu comme "Peu importe, mec" ou "C'est plutôt stupide" échouerait. Seul un contenu respectueux et neutre passe à haute sensibilité.

Une haute sensibilité est appropriée pour les applications destinées aux enfants où vous avez besoin d'une protection maximale, les plateformes éducatives où vous souhaitez modéliser une communication respectueuse, les espaces sûrs et les communautés de soutien où les gens ont besoin de se sentir en sécurité, et les contextes critiques en matière de conformité où tout problème potentiel doit être détecté.

Options de configuration

Le garde-fou contre le langage toxique accepte plusieurs options de configuration qui vous permettent d'ajuster son comportement pour votre cas d'utilisation spécifique.

TypeScript

```typescript
// Available options:
// sensitivity: "low", "medium", or "high" (default: "medium")
// model: model identifier (default: Claude 3.5 Haiku)
// temperature: 0-1, lower values = more consistent (default: 0.1)
// maxTokens: response length limit (default: 200)

await abv.guardrails.toxicLanguage.validate(text, {
  sensitivity: "medium",
  model: "model-name",
  temperature: 0.1,
  maxTokens: 200,
});
```

Python

```
# Available options:
# sensitivity: "low", "medium", or "high" (default: "medium")
# model: model identifier (default: Claude 3.5 Haiku)
# temperature: 0-1, lower values = more consistent (default: 0.1)
# maxTokens: response length limit (default: 200)

abv.guardrails.toxic_language.validate(text, {
    "sensitivity": "medium",
    "model": "model-name",
    "temperature": 0.1,
    "maxTokens": 200
})
```

L'option de sensibilité est la plus importante et vous l'utiliserez fréquemment. Les options de modèle, de température et de maxTokens sont des paramètres avancés que vous n'aurez généralement pas besoin de modifier. Le modèle par défaut est optimisé pour les tâches de garde-fou et offre le meilleur équilibre entre vitesse, précision et coût. La température par défaut de 0.1 garantit des résultats cohérents. Le maxTokens par défaut de 200 est suffisant pour le champ d'explication.

Exemples du monde réel

Examinons des exemples concrets de la manière dont différents niveaux de sensibilité traitent divers types de contenu. Comprendre ces schémas vous aidera à choisir la bonne sensibilité pour votre application.

Considérez un message comme "Je ne suis pas d'accord avec votre approche de ce problème." Il s'agit d'un désaccord professionnel qui passe à tous les niveaux de sensibilité. Le langage est neutre et respectueux malgré l'expression du désaccord.

Maintenant, considérez "C'est une terrible idée et montre un mauvais jugement." Cela passe à faible et moyen niveau de sensibilité car, bien que critique, cela se concentre sur l'idée plutôt que d'attaquer la personne. Cependant, cela pourrait échouer à un niveau de sensibilité élevé car "terrible" et "mauvais jugement" pourraient être considérés comme dédaigneux.

Un message comme "Tu ne sais pas de quoi tu parles" échoue à un niveau de sensibilité moyen et élevé car il attaque directement la compétence de la personne. Cela pourrait passer à faible sensibilité puisqu'il ne contient pas de menaces explicites ou de discours de haine, bien que ce soit limite.

Un contenu comme "Tu es un idiot" ou "Les gens comme toi sont le problème" échoue à tous les niveaux de sensibilité. Ce sont des attaques personnelles claires sans valeur constructive.

Enfin, des menaces explicites comme "Je te trouverai et je te ferai du mal" échouent à tous les niveaux de sensibilité avec une confiance maximale. C'est un contenu toxique sans ambiguïté.

Modèles de mise en œuvre

Voici comment vous utiliseriez typiquement la détection de langage toxique dans différentes parties de votre application.

Pour la validation des entrées, vous vérifiez les messages des utilisateurs avant de les envoyer à votre IA ou de les afficher à d'autres utilisateurs :

TypeScript

```typescript
async function validateUserMessage(message: string): Promise<boolean> {
  const result = await abv.guardrails.toxicLanguage.validate(
    message,
    { sensitivity: "medium" }
  );

  if (result.status === "pass") {
    return true;
  }

  // Log the reason for monitoring, but don't expose it to the user
  console.log("Blocked message:", result.reason);
  return false;
}

// Usage in your message handler
if (await validateUserMessage(userInput)) {
  await processMessage(userInput);
} else {
  return { error: "Your message violates our community guidelines." };
}
```

Python

```python
async def validate_user_message(message: str) -> bool:
    result = await abv.guardrails.toxic_language.validate_async(
        message,
        {"sensitivity": "medium"}
    )

    if result["status"] == "pass":
        return True

    # Log the reason for monitoring, but don't expose it to the user
    print(f"Blocked message: {result['reason']}")
    return False

# Usage in your message handler
if await validate_user_message(user_input):
    await process_message(user_input)
else:
    return {"error": "Your message violates our community guidelines."}
```

Pour la validation des sorties, vous vérifiez les réponses générées par l'IA avant de les montrer aux utilisateurs :

TypeScript

```typescript
async function generateSafeResponse(prompt: string): Promise<string> {
  // Generate initial response
  let response = await callAI(prompt);

  // Validate the response
  const validation = await abv.guardrails.toxicLanguage.validate(
    response,
    { sensitivity: "high" }
  );

  // If toxic, regenerate with explicit safety instruction
  if (validation.status === "fail") {
    response = await callAI(
      prompt + "\n\nIMPORTANT: Respond in a professional, respectful tone."
    );
  }

  return response;
}
```

Python

```python
async def generate_safe_response(prompt: str) -> str:
    # Generate initial response
    response = await call_ai(prompt)

    # Validate the response
    validation = await abv.guardrails.toxic_language.validate_async(
        response,
        {"sensitivity": "high"}
    )

    # If toxic, regenerate with explicit safety instruction
    if validation["status"] == "fail":
        response = await call_ai(
            f"{prompt}\n\nIMPORTANT: Respond in a professional, respectful tone."
        )

    return response
```

Pour gérer les cas ambigus, vous pourriez mettre en place une file d'attente de révision pour les résultats incertains :

TypeScript

```typescript
async function handleUserContent(content: string) {
  const result = await abv.guardrails.toxicLanguage.validate(
    content,
    { sensitivity: "medium" }
  );

  if (result.status === "pass") {
    // Content is clearly acceptable
    await publishContent(content);
  } else if (result.status === "fail" && result.confidence > 0.8) {
    // High-confidence violation, auto-reject
    await rejectContent(content, "Community guidelines violation");
  } else {
    // Low confidence or unsure - flag for human review
    await flagForModeration(content, result);
  }
}
```

Python

```python
async def handle_user_content(content: str):
    result = await abv.guardrails.toxic_language.validate_async(
        content,
        {"sensitivity": "medium"}
    )

    if result["status"] == "pass":
        # Content is clearly acceptable
        await publish_content(content)
    elif result["status"] == "fail" and result["confidence"] > 0.8:
        # High-confidence violation, auto-reject
        await reject_content(content, "Community guidelines violation")
    else:
        # Low confidence or unsure - flag for human review
        await flag_for_moderation(content, result)
```

Optimisation des performances

Puisque la détection de langage toxique utilise l'IA, cela prend une à trois secondes par vérification et consomme des jetons. Vous pouvez optimiser les performances en effectuant d'abord une vérification rapide basée sur des règles pour attraper les violations évidentes avant de faire l'appel coûteux à l'IA.

TypeScript

```typescript
async function efficientToxicCheck(text: string): Promise<boolean> {
  // Quick check for explicitly forbidden terms (under 10ms, free)
  const quickCheck = await abv.guardrails.containsString.validate(
    text,
    {
      strings: ["explicit-slur", "forbidden-term"],
      mode: "none",
    }
  );

  // If quick check fails, no need for expensive AI check
  if (quickCheck.status === "fail") {
    return false;
  }

  // Only run AI check if quick check passed
  const deepCheck = await abv.guardrails.toxicLanguage.validate(text);
  return deepCheck.status === "pass";
}
```

Python

```python
async def efficient_toxic_check(text: str) -> bool:
    # Quick check for explicitly forbidden terms (under 10ms, free)
    quick_check = await abv.guardrails.contains_string.validate_async(
        text,
        {
            "strings": ["explicit-slur", "forbidden-term"],
            "mode": "none"
        }
    )

    # If quick check fails, no need for expensive AI check
    if quick_check["status"] == "fail":
        return False

    # Only run AI check if quick check passed
    deep_check = await abv.guardrails.toxic_language.validate_async(text)
    return deep_check["status"] == "pass"
```

Meilleures pratiques de sécurité

Ne jamais exposer le champ de raison aux utilisateurs finaux. La raison explique pourquoi le contenu a échoué à la validation, et exposer cette information aide les acteurs malveillants à apprendre comment contourner vos garde-fous. Au lieu de cela, utilisez des messages d'erreur génériques tout en enregistrant la raison détaillée en interne pour le suivi et l'amélioration.

TypeScript

```typescript
// Bad - exposes validation logic
if (result.status === "fail") {
  return { error: result.reason };  // Don't do this!
}

// Good - generic message, internal logging
if (result.status === "fail") {
  logger.info("Blocked toxic content", { reason: result.reason });
  return { error: "Your message violates our community guidelines." };
}
```

Python

```python
# Bad - exposes validation logic
if result["status"] == "fail":
    return {"error": result["reason"]}  # Don't do this!

# Good - generic message, internal logging
if result["status"] == "fail":
    logger.info(f"Blocked toxic content: {result['reason']}")
    return {"error": "Your message violates our community guidelines."}
```

Choisir la bonne sensibilité

Voici un cadre de décision pour choisir la sensibilité en fonction de votre type d'application. Si vous construisez pour des enfants ou des populations vulnérables, utilisez toujours une haute sensibilité. Le potentiel de dommages causés par l'autorisation de contenu toxique l'emporte de loin sur le coût des faux positifs.

Si vous construisez une application destinée aux clients comme un service client, des réseaux sociaux ou des outils collaboratifs, une sensibilité moyenne est généralement appropriée. Elle détecte les violations claires tout en permettant des désaccords professionnels.

Si vous construisez pour des publics professionnels ou techniques où un débat robuste est attendu, envisagez une faible sensibilité. Les forums techniques, les systèmes de révision de code et les outils de retour professionnel bénéficient de la possibilité d'exprimer des opinions fortes.

Vous pouvez également ajuster la sensibilité en fonction du contexte de l'utilisateur. Les utilisateurs authentifiés avec un bon historique pourraient avoir une sensibilité plus faible tandis que les utilisateurs anonymes obtiennent une sensibilité plus élevée. Les utilisateurs qui s'identifient comme mineurs obtiennent automatiquement une haute sensibilité, quelle que soit la configuration par défaut.

Prochaines étapes

La barrière de langage toxique est souvent utilisée avec d'autres barrières pour une validation complète du contenu. Envisagez de la combiner avec la détection de langage biaisé pour une solution de sécurité du contenu plus complète. Vous voudrez peut-être également utiliser une chaîne de contenu pour attraper rapidement les termes interdits explicites avant d'exécuter le contrôle de langage toxique plus coûteux.

Pour des conseils d'implémentation plus détaillés, consultez la documentation des meilleures pratiques qui couvre les stratégies d'optimisation, la gestion des erreurs et les approches de surveillance.

JS

```
asdasdasdasdasdasdasdasdasdsa
```

8.2. ASDASDSDA

ASDASDASDAS

ASDASDASDAS

> ASDSADASDSAD

8.3. Test de numérotation sur les images problème (cloné)

Le paquet des Détails Bancaires de l'Entité (EBD) s'intègre avec l'enregistrement standard des Fournisseurs de NetSuite pour stocker en toute sécurité toutes les informations nécessaires à la soumission des paiements. Une fois installé, les données des fournisseurs et des employés importées sont accessibles via **Configuration → Détails Bancaires de l'Entité → Aperçu**. Découvrez comment importer, exporter et mettre à jour l'EBD pour les fournisseurs et les employés dans ce guide.

Le processus d'importation et d'exportation des enregistrements dans les onglets Fournisseur et Employé est le même. La seule différence se trouve dans l'onglet Fournisseur, qui propose un utilitaire d'exportation spécialisé pour les utilisateurs du paquet de Paiements Bancaires Électroniques de NetSuite, leur permettant de transférer et de re-télécharger facilement des données dans l'EBD.

Importer les Données Bancaires de l'Entité

Le processus d'importation des données EBD est le même pour les Enregistrements de Fournisseurs et les Enregistrements d'Employés.

1 Go to the Import/Export Tab

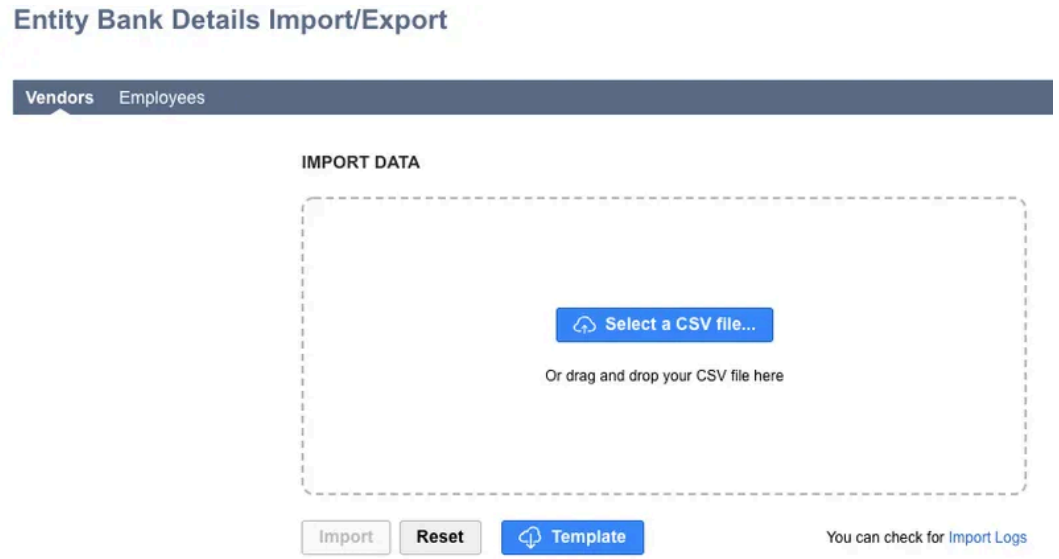
Aller à **Configuration → Entité Détails bancaires → Import/Export**



Go to Import/Export

2 Import CSV template

Sélectionner **Modèle** pour télécharger un modèle CSV vierge. Utilisez ce CSV pour remplir les détails bancaires pour un téléchargement en masse.



Import Data section. Click on the Download Template button to get a blank CSV

> **REMARQUE :** Utilisez cette fonctionnalité uniquement pour créer de nouveaux enregistrements. Pour mettre à jour des enregistrements existants, vous devez utiliser le **Exporter CSV** fonctionnalité, car elle inclut la colonne ID nécessaire, qui est absente du modèle CSV vierge.

3 Format and Fill Out CSV

Veuillez consulter le **Exigences de champ pour CSV** section de cette page pour un tableau contenant toutes les exigences de formatage pour le fichier CSV.

> **REMARQUE :** Confirmez que vous utilisez le code pays IBAN (par exemple, CA).

> **CONSEIL :** Avez-vous des zéros en tête dans vos données ? Des apostrophes ['] devant les zéros en tête ou le formatage des cellules en tant que "texte brut" conserveront les zéros en tête dans Microsoft Excel ou Google Sheets, mais PAS dans Apple Numbers.

4 Upload the Formatted CSV

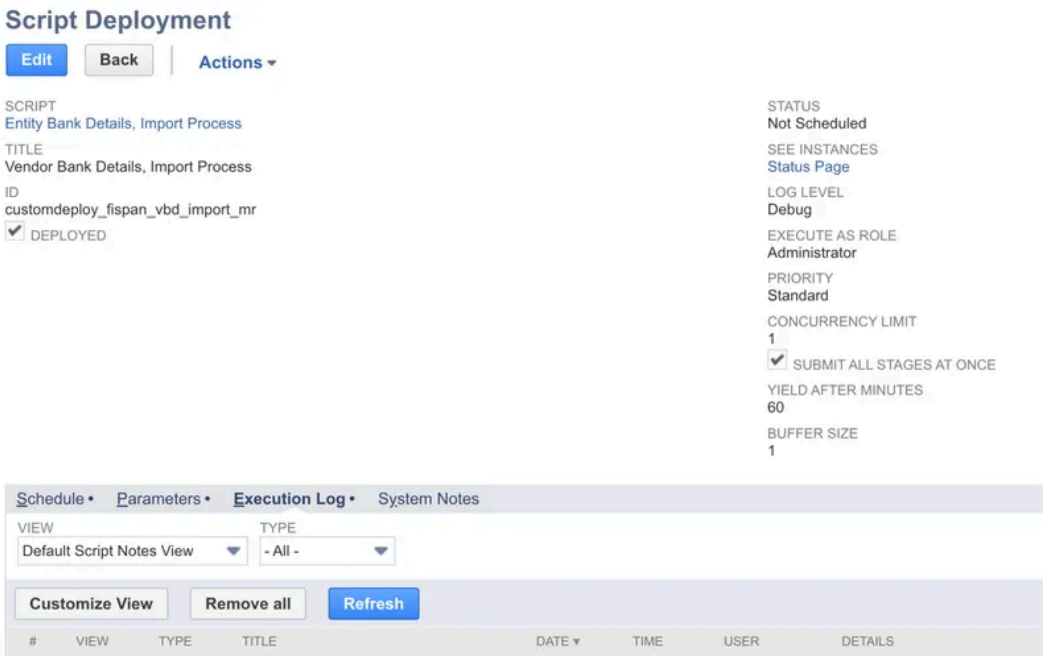
Dans le même écran d'importation/exportation, téléchargez le document CSV formaté.

Confirmez le téléchargement réussi en vérifiant le **Importation Logs** pour vous assurer que tous les enregistrements ont été téléchargés avec succès.



Import Data section

Sélectionner **Importer les journaux → Journaux d'exécution** pour confirmer l'achèvement de l'importation.



Execution Logs

5 Check Information

Avant d'envoyer des paiements, vérifiez le **Entité Détails bancaires** de l'onglet du dossier du fournisseur pour vous assurer que toutes les informations sont correctement mises à jour.

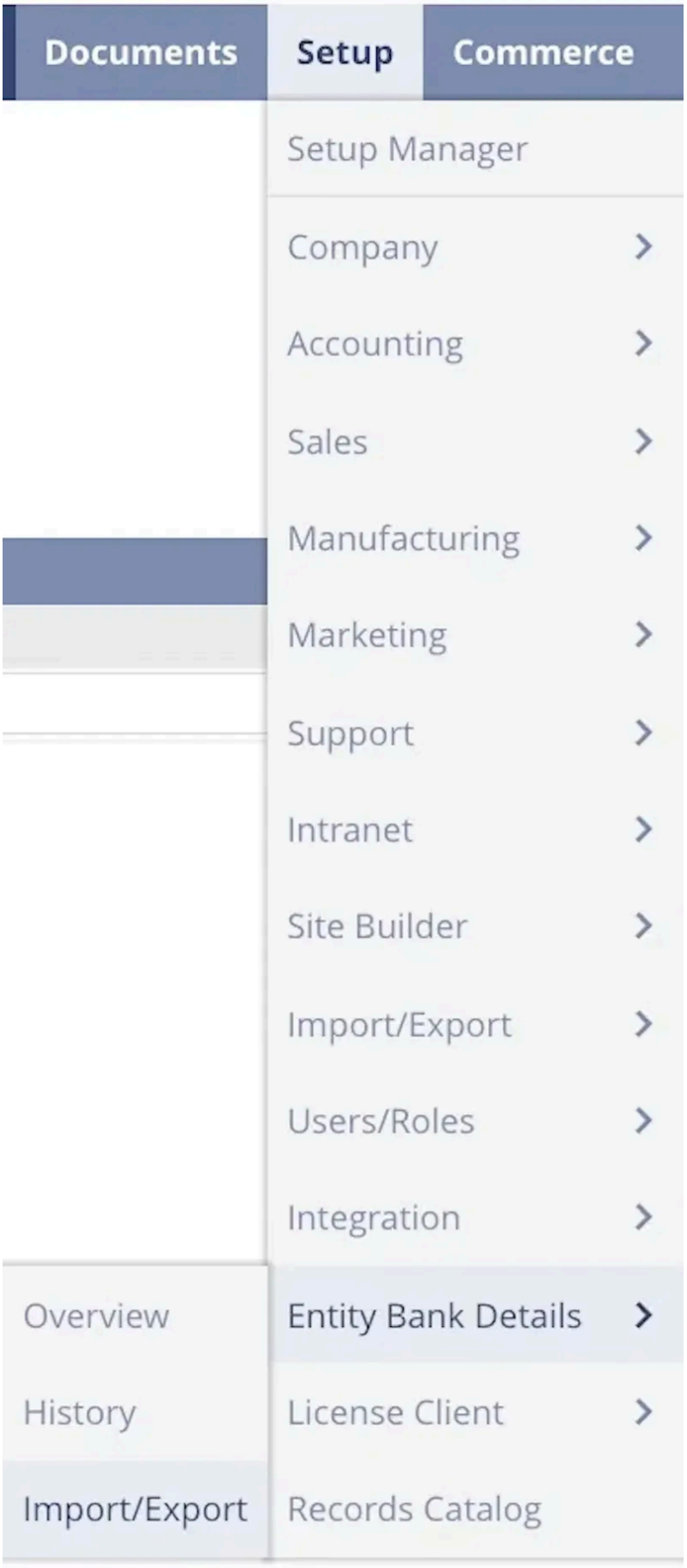
> **IMPORTANT :** Tous les détails bancaires des entités pour les fournisseurs et les employés, ainsi que tous les journaux d'historique, sont stockés dans votre environnement NetSuite. Veuillez vous assurer que vous exportez régulièrement vos enregistrements de détails bancaires des entités et vos journaux d'historique pour éviter toute perte potentielle, comme par exemple lors de la désinstallation du bundle.

Exporter les données bancaires de l'entité

Le processus d'exportation des données EBD est le même pour les dossiers des fournisseurs et les dossiers des employés.

1 Allez à l'onglet Import/Export

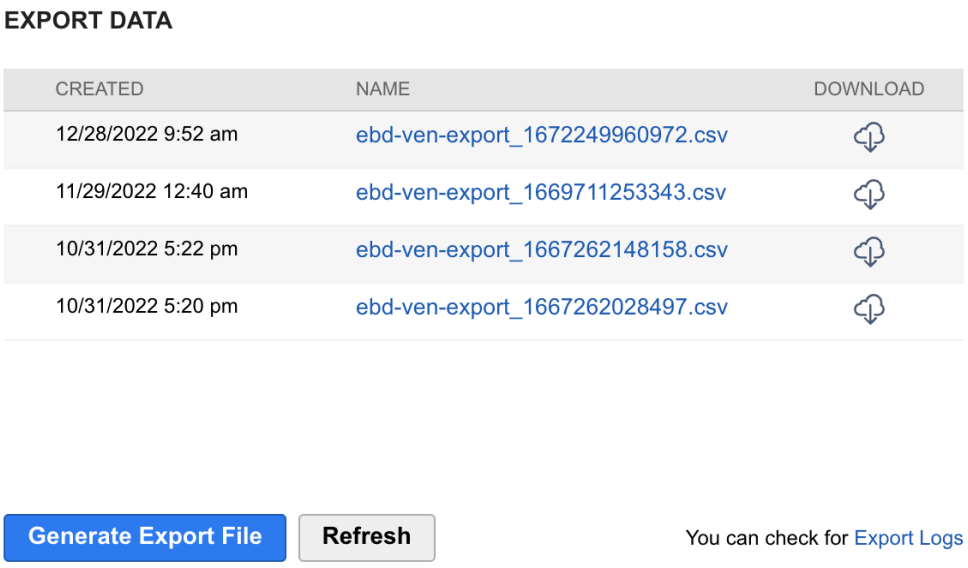
Allez à **Configuration → Entité Détails bancaires → Import/Export**


Documents
Setup
Commerce
Setup Manager
Company
Accounting
Sales
Manufacturing
Marketing
Support
Intranet
Site Builder
Import/Export
Users/Roles
Integration
Overview
History
Import/Export
Entity Bank Details
License Client
Records Catalog

Allez à Import/Export

2 Modèle d'exportation CSV

Cliquez sur le **Générer le fichier d'exportation** bouton pour déclencher un téléchargement CSV de tous les enregistrements conservés dans EBD.



Section Exporter des données de la page Import/Export

Le tableau affichera un historique de toutes les exportations créées, avec des horodatages et des boutons pour télécharger chacun de ces fichiers.

Les journaux d'exportation contiennent plus d'informations sur les fichiers d'exportation générés. Cela peut être trouvé sous l'onglet **Journal d'exécution**. Pour chaque ligne, il indiquera qui a initié l'exportation, le nombre d'enregistrements exportés dans un fichier, ainsi que le nom du fichier pour cette exportation particulière.

Mettre à jour les enregistrements EBD existants

Utilisez la fonction Importer/Exporter pour modifier en masse les détails de la banque d'entités (EBD) existants pour plusieurs enregistrements d'employés et de fournisseurs à la fois.

1 Exporter le fichier CSV

Pour mettre à jour les enregistrements de détails de banque d'entités (EBD) existants, vous devez d'abord exporter un fichier CSV de ces enregistrements.

Le fichier CSV exporté est crucial car il contient une colonne d'ID unique. Cette colonne d'ID n'est pas présente dans le modèle d'importation CSV vierge et est nécessaire pour que le système identifie et mette à jour les données existantes.

2 Apportez des modifications au CSV

À l'aide de votre application préférée (comme Microsoft Excel ou Google Sheets), ouvrez le fichier CSV exporté pour apporter vos modifications.



ebd-ven-export_

	id	vendor	label	primary	country	currency	method	bic	accountType	accountNumber	routingNumber
2	1	1305	US-DOMESTIC-USD	T	US	USD	DOMESTIC		CHECKING		

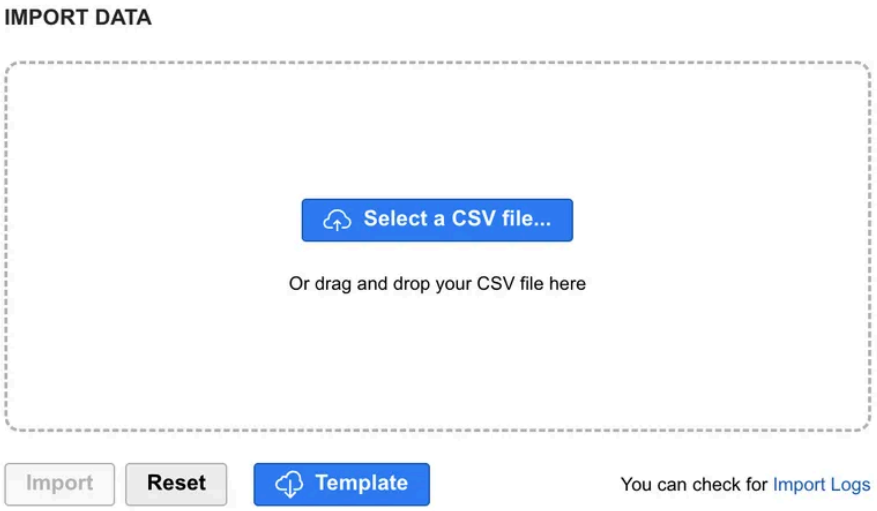
EBD Export CSV du fournisseur ouvert dans Numbers. Notez la colonne ID à gauche

Assurez-vous que la **colonne ID** à gauche reste intacte.

Vous pouvez également ajouter de nouveaux enregistrements à la liste ; il vous suffit de les entrer comme une nouvelle ligne en bas du fichier CSV. Le système attribuera un ID à ces nouveaux enregistrements lors de l'importation.

3 Importer le fichier Export modifié

Après avoir apporté des modifications au fichier précédemment exporté, vous pouvez maintenant importer le fichier CSV dans le bundle des détails de la banque d'entités.



Section Importer des données. Cliquez sur "Sélectionner un fichier CSV" ou faites glisser et déposez un fichier CSV dans cette section pour importer votre fichier CSV

Soit **Sélectionner un fichier CSV** à télécharger, ou faites glisser et déposez-en un dans la section **Importer des données** de la page Importer/Exporter.

Dépannage des problèmes d'importation courants

Problème	Résolution
Aucun message de succès n'apparaît dans NetSuite après l'importation des données bancaires de l'entité.	Vérifiez les journaux d'importation directement pour vérifier l'état de l'importation.
Les enregistrements ont été créés avec succès mais semblent incomplets ou inexacts.	Les enregistrements peuvent être créés même s'ils manquent de champs requis ou ont un format incorrect. Validez les enregistrements importés pour leur exactitude après chaque téléchargement.
Des enregistrements bancaires en double sont créés pour un fournisseur/un employé.	Cela se produit si les mêmes enregistrements sont importés plus d'une fois. Assurez-vous d'importer des comptes bancaires de fournisseurs ou d'employés uniques avec chaque fichier d'importation.
La méthode de paiement internationale est mal définie ou manquante.	Lors de l'importation, la méthode de paiement pour tous les comptes bancaires internationaux doit être saisie comme INTERNATIONAL_WIRE dans le modèle EBD. De plus, assurez-vous que les comptes bancaires pour les pays de virement international ne sont pas définis sur la méthode de paiement DOMESTIC dans le profil NetSuite du fournisseur.

FAQs

Est-il possible d'ajouter des champs supplémentaires ?

Non. NetSuite ne permet actuellement pas l'ajout de champs à l'enregistrement EBD.

Pourquoi les champs Solde bancaire et Solde au date dans la page de correspondance des données bancaires sont-ils vides après une importation réussie ?

C'est une limitation de NetSuite. Le système ne remplit pas les informations de solde bancaire car il dépend de l'utilisation d'un flux bancaire, qui n'est pas intégré ici.

Exigences de champ pour CSV

Assurez-vous que votre fichier d'importation CSV respecte ces exigences.

IMPORTANT : Assurez-vous que toute donnée avec des zéros en tête est précédée d'un apostrophe `'` ou formatée en texte brut dans Microsoft Excel ou Google Sheets.

REMARQUE: Si vous utilisez des méthodes de paiement qui nécessitent des codes de pays obligatoires pour les détails bancaires de l'entité, il peut y avoir des colonnes supplémentaires qui doivent être incluses dans l'importation.

Pour garantir une importation réussie, nous vous recommandons d'abord d'ajouter manuellement les détails bancaires de l'entité au système ERP. Ensuite, téléchargez le modèle ; il affichera déjà les colonnes correctes nécessaires pour entrer les détails restants.

Nom	Valeurs	Description
identifiant	Numérique	Ceci est le numéro d'identification pour cet enregistrement spécifique. Cette colonne n'est pas disponible dans le modèle d'importation car elle est générée par le système.
vendeur (ou employé)	ID de fournisseur NetSuite ou ID d'employé	Requis. L'identifiant du fournisseur ou de l'employé peut être trouvé dans l'URL de l'enregistrement de l'entité ou sur la page d'aperçu EBD.
étiquette	Tout	Requis. Cela peut être des informations sur l'entité ou la banque. Ce champ est utilisé pour différencier les enregistrements lors des téléchargements en masse.
primaire	OUI/NON, O/N, VRAI/FAUX, V/F	Requis. T = Vrai, F = Faux. S'il y a plusieurs détails bancaires pour un seul mode de paiement sur un seul fournisseur importé, marquez-en un comme T et tous les autres comme F.
pays	Code de pays ISO à 2 lettres (par exemple, US, CA, UK)	Requis. Le champ accepte des valeurs en majuscules, minuscules et mixtes.
monnaie	Code de devise à 3 lettres requis (par exemple, USD, CAD)	Requis. Le champ accepte des valeurs en majuscules, minuscules et en casse mixte.
méthode	DOMESTIQUE, TRANSFERT INTERNATIONAL	Requis. Doit être dans le même format car il est sensible à la casse. Veuillez utiliser INTERNATIONAL_WIRE pour les comptes bancaires configurés pour les paiements par virement international. Veuillez utiliser DOMESTIC pour les comptes basés aux États-Unis/Canada.
nomDeLaBanque	Tout	
adresse.ligne1	Tout	
adresse.ville	Tout	
adresse.provinceÉtat	Tout	
code postal	Tout	
bic	Code BIC ou SWIFT valide pour le pays et la banque	
typeDeCompte	VÉRIFICATION, ÉPARGNE	Le champ accepte des valeurs en majuscules, en minuscules et en cas mixte.
numéroDeCompte	Numéro de compte valide pour cette banque	
codeDeBrancheLocale	Code de branche valide pour cette banque	Utilisez ce champ pour les codes BSB si le pays l'exige.
messageDeButDeP	Tout	

Nom	Valeurs	Description
aiement		
codeDeButDePaiement	Code de but valide	
codeIsoDePaiement.par défautsDePaiement	Code ISO valide	
code de paiement.code mot clé	Code de paiement valide	
typeDePartieDePaiement.par défautsDePaiement	Type de partie valide	P = Parent, T = Filiale, G = Groupe, N = Non lié
statutRésidentiel.paiementDéfauts	Statut résidentiel valide	
codeBanque	Code bancaire valide pour cette banque	
iban	Code pays à 2 lettres, suivi de deux chiffres de contrôle, et jusqu'à 35 caractères alphanumériques	
code de tri	Code de tri valide pour cette banque	
numéro d'institution	Numéro d'institution valide pour cette banque	
numéro de transit	Numéro de transit valide pour cette banque	
numéroDeRoutage	Numéro de routage valide pour cette banque	

Transfert des données de paiements bancaires électroniques

Avant l'installation du bundle Détails de la Banque de l'Entité (EBD), vous êtes limité à effectuer uniquement des paiements ACH et CPA. Cela se fait via le plugin bancaire, en utilisant les informations des fournisseurs stockées dans le bundle de Paiements Bancaires Électroniques (EBP) de NetSuite.

Une fois l'EBD installé, le bundle EBP n'est plus utilisé pour extraire les données bancaires des fournisseurs pour les paiements de factures. Au lieu de cela, l'EBD devient la nouvelle source d'informations de paiement.

Le bundle EBD permet de transférer les informations bancaires des fournisseurs ACH et CPA du bundle EBP via CSV.

Les informations bancaires ACH et CPA stockées dans les formats de fichiers de paiement par défaut suivants sont prises en charge :

- ACH - CCD/PPD
- ACH - CTX (Texte libre)
- CPA-005

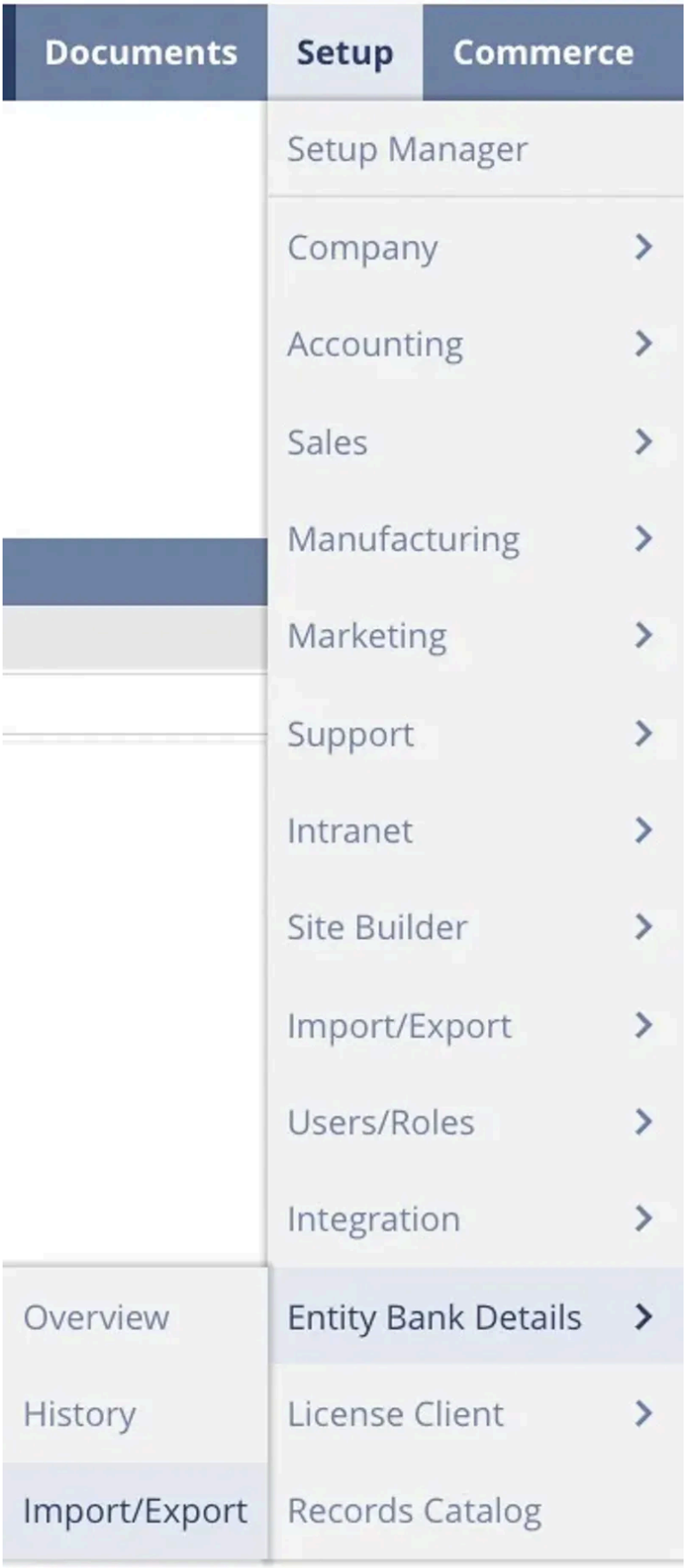
Le transfert des informations bancaires ACH et CPA qui sont stockées dans des formats de fichiers de paiement personnalisés n'est pas pris en charge.

Pour transférer rapidement vos données prises en charge entre les ensembles, suivez les étapes ci-dessous.



1 Go to the Import/Export Tab

Aller à **Configurer → Détails bancaires de l'entité → Import/Export**

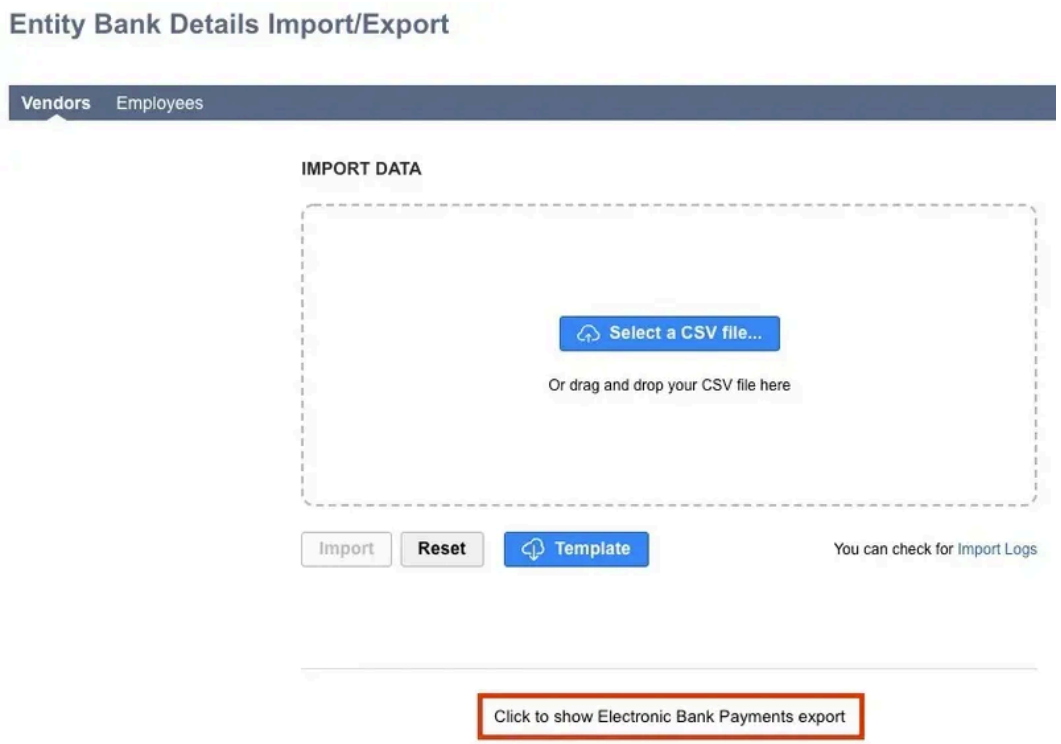

Documents
Setup
Commerce
Setup Manager
Company
Accounting
Sales
Manufacturing
Marketing
Support
Intranet
Site Builder
Import/Export
Users/Roles
Integration
Overview
History
Import/Export
Entity Bank Details
License Client
Records Catalog



Go to Import/Export

2 Generate CSV(s)

Sélectionner **Cliquez pour afficher l'exportation des paiements bancaires électroniques**

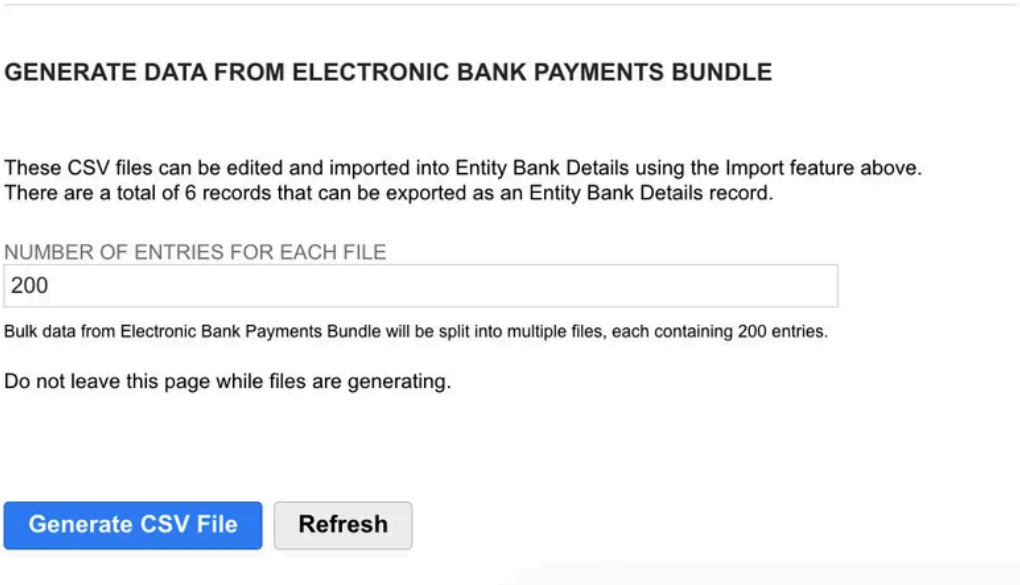


Click to show Electronic Bank Payments export

Sélectionner **Générer un fichier CSV.**

Cela générera un ou plusieurs fichiers CSV en fonction du **Nombre d'Entrées pour Chaque Fichier**, défini ci-dessous.

NetSuite limite le nombre d'enregistrements pouvant être extraits par fichier. Par exemple, si vous avez 1100 enregistrements et que vous définissez le nombre d'entrées à 200, le bundle générera cinq fichiers CSV contenant 200 enregistrements et un fichier CSV contenant 100 enregistrements.

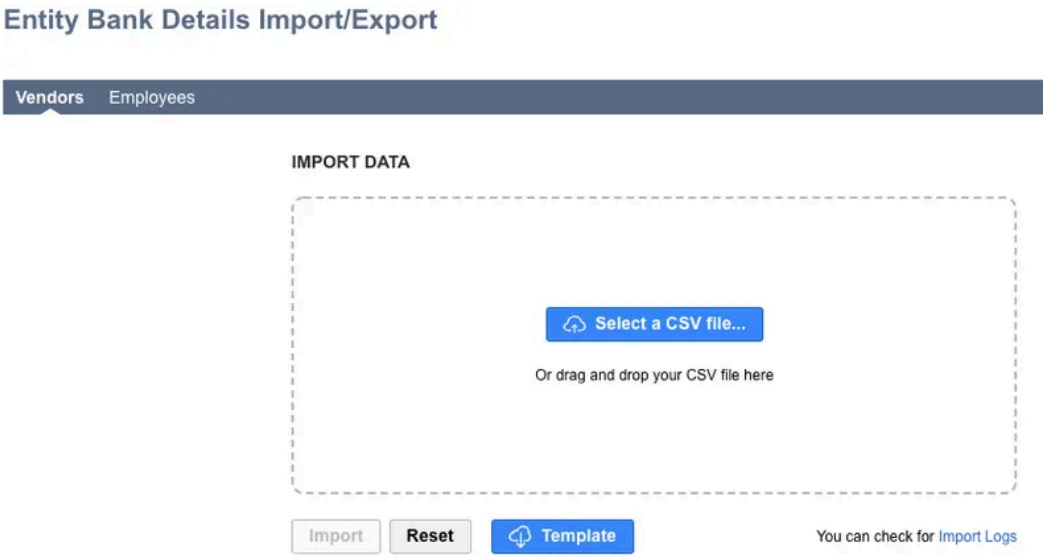


Generate CSV file

> **REMARQUE :** Le **Générer des données à partir du bundle de paiements bancaires électroniques** ne peut extraire que les détails de paiement ACH et CPA. Tout détail de paiement supplémentaire doit être transféré dans le bundle des détails bancaires de l'entité soit par saisie manuelle, soit en utilisant la fonction d'importation.

3 Import CSV(s)

Sélectionnez un fichier CSV...ou faites glisser et déposez le(s) fichier(s) CSV dans le **Importer des données** boîte.



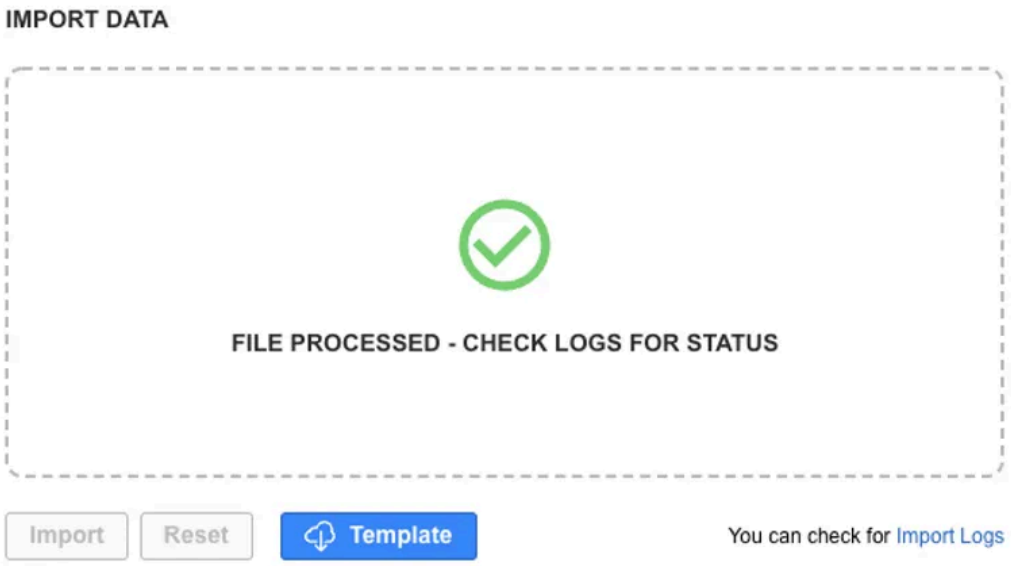
Import Data section

Ensuite, sélectionnez **Importer.**

Attendez quelques instants que l'importation soit terminée.

4 Confirm Import Completion

Une fois terminé, sélectionnez **Importer les journaux → Journaux d'exécution** pour confirmer l'achèvement de l'importation.



Confirm Status

Vous pouvez également vérifier les endroits suivants pour confirmer que les détails bancaires ont été enregistrés :

- Onglet Détails de la Banque de l'Entité sur le Dossier du Fournisseur ou de l'Employé
- Aperçu des détails de la banque de l'entité
- Historique des détails de la banque de l'entité

8.3.1. asdsadas

About Profile

Profile at a Glance

Que vous commenciez tout juste ou que vous recherchiez des conseils rapides, cette section vous guidera à travers les bases de l'utilisation de Profile. Chaque article comprend des liens vers des sujets connexes, afin que vous puissiez explorer davantage si nécessaire.

Cette section présente les fonctionnalités et fonctions essentielles de Profile qui soutiennent l'utilisation quotidienne. Elle couvre des informations générales telles que la connexion et la déconnexion, le verrouillage de votre écran, la navigation entre les fenêtres et la personnalisation de votre espace de travail. Vous trouverez également des conseils sur l'utilisation des filtres et des outils de recherche pour localiser des informations, des touches de raccourci, des paramètres de mise en page pour améliorer l'efficacité, et plus encore.

Ces ressources sont conçues pour aider tous les utilisateurs à gagner en confiance et à rester productifs lors de l'utilisation de Profile.

Get Started

Si vous souhaitez en savoir plus sur les informations générales liées au Profil, veuillez contacter votre Responsable du Succès Client.

Job Aids

Gardez les informations clés à portée de main. Cliquez ci-dessous pour télécharger des guides de référence rapide imprimables :

JobAid-IntroductionToProfile.pdf

Related Articles

Lisez les articles connexes ci-dessous pour plus d'informations générales sur le profil :

- **Connexion au profil**
- **Déconnexion du profil**
- **Aperçu de la navigation dans le profil**
- **Utiliser la recherche dans le profil**
- **Naviguer dans la fenêtre du centre de travail**
- **Naviguer dans les dossiers médicaux dans le profil**
- **Travailler avec les fenêtres de profil**

- Utilisez la fenêtre des détails cliniques
- Utiliser des listes de segments et des vues d'enregistrement
- Insérer et modifier des informations dans le profil
- Utiliser des filtres dans le profil
- Changement de lieu de service
- Déconnexion du profil
- Verrouillez votre écran dans le profil
- Enregistrez les paramètres de votre profil Windows
- Basculer entre les fenêtres ouvertes dans le profil
- Utiliser les touches de raccourci

Frequently Asked Questions

Frequently Asked Questions:

Access & Security

Why is it important to log out or lock my Profile session?

Se déconnecter à la fin de votre session et verrouiller votre écran lorsque vous vous éloignez garantit qu'aucune personne non autorisée ne peut accéder aux informations des patients ou à votre espace de travail.

What's the difference between logging out and locking my screen?

Se déconnecter met fin à votre session complètement. Verrouiller votre écran garde votre session ouverte mais empêche les autres d'y accéder jusqu'à ce que vous déverrouilliez avec vos identifiants.

Windows & Layout

How do I manage multiple Profile windows at once?

Vous pouvez basculer entre les fenêtres ouvertes en utilisant le menu Fenêtre ou en appuyant sur F2. Vous pouvez également redimensionner, réorganiser et enregistrer les paramètres de la fenêtre afin que votre disposition préférée se charge automatiquement.

Can I save my window setup so I don't have to rearrange every time?

Oui. Utilisez **Édition > Paramètres par défaut > Définir les paramètres par défaut pour cette fenêtre** pour enregistrer la taille, la position et la disposition. Celles-ci seront appliquées automatiquement lorsque la fenêtre se rouvrira.

Navigation & Views

What are segment lists and record views used for?

Les listes de segments vous permettent de passer d'une catégorie d'informations à l'autre dans le Centre de travail, tandis que les vues d'enregistrement affichent des détails sur une sélection spécifique. Ensemble, elles vous aident à organiser et à accéder efficacement aux données des patients ou des tâches.

How do I quickly find the information I need in Profile?

Utilisez des filtres pour réduire les listes aux éléments pertinents, ou utilisez des champs de recherche pour effectuer des recherches dans l'ensemble de la base de données. Les champs de recherche sont marqués d'une ellipse (...) et permettent à la fois une saisie directe et une sélection dans un menu déroulant

Productivity Tools

Are there shortcut keys to speed up my work in Profile?

Oui. Les raccourcis courants incluent Ctrl+N (nouveau patient), Ctrl+F (trouver des patients), F2 (changer de fenêtres) et F1 (aide). L'utilisation de raccourcis réduit la dépendance aux menus et accélère la saisie des données.

Can I restore Profile to default settings if I change too much?

Oui, vous pouvez réinitialiser les paramètres par défaut sous le **Édition > Paramètres par défaut** menu, ou revenir manuellement aux dispositions de fenêtres et aux paramètres de filtre.

Related Articles

Lisez les articles connexes ci-dessous pour plus d'informations générales sur le Profil :

- **Connexion au profil**
- **Déconnexion du profil**
- **Aperçu de la navigation du profil**
- **Utiliser la recherche dans le profil**
- **Naviguer dans la fenêtre du centre de travail**
- **Naviguer dans les dossiers médicaux dans le profil**
- **Travailler avec les fenêtres de profil**
- **Utilisez la fenêtre des détails cliniques**
- **Utiliser des listes de segments et des vues d'enregistrement**
- **Insérer et modifier des informations dans le profil**
- **Utiliser des filtres dans le profil**
- **Changement de lieu de service**
- **Déconnexion du profil**
- **Verrouillez votre écran dans le profil**
- **Enregistrez les paramètres de votre profil Windows**

- **Basculer entre les fenêtres ouvertes dans le profil**
- **Recherche de Patients**
- **Utiliser les touches de raccourci**

8.3.2. Langage Toxique

Le garde-fou contre le langage toxique détecte le contenu nuisible, y compris les discours de haine, les menaces, les insultes et d'autres communications qui pourraient nuire à votre communauté ou à votre marque. Contrairement aux simples filtres de mots-clés, ce garde-fou utilise l'IA pour comprendre le contexte, le ton et l'intention.

Quand utiliser ce garde-fou

Vous devriez utiliser la détection de langage toxique lorsque vous devez protéger votre communauté des interactions nuisibles, empêcher votre IA de recevoir un contexte empoisonné qui pourrait influencer son comportement, ou maintenir la sécurité de la marque dans les résultats générés par l'IA. Ce garde-fou est particulièrement précieux dans les applications avec du contenu généré par les utilisateurs, comme les forums, les systèmes de chat et les sections de commentaires, ainsi que dans les scénarios de service client où vous devez attraper des messages hostiles avant qu'ils n'atteignent les agents ou les systèmes d'IA.

L'avantage clé par rapport à un simple filtrage par mots-clés est que ce garde-fou comprend les nuances. Quelqu'un peut être hostile sans utiliser de grossièretés, et il peut utiliser un langage fort sans être hostile. Le garde-fou analyse l'intention et le ton, pas seulement le choix des mots.

Comprendre les niveaux de sensibilité

Le paramètre de sensibilité contrôle la rigueur du garde-fou. Pensez-y comme à l'ajustement du seuil de ce qui constitue une violation. C'est l'un des choix de configuration les plus importants que vous ferez, car il change fondamentalement le contenu qui passe ou échoue.

Une faible sensibilité ne signale que les violations graves comme les menaces explicites et les discours de haine extrêmes. Lorsque vous réglez la sensibilité sur faible, vous dites que vous vous attendez à des opinions fortes et à des désaccords robustes, et que vous ne voulez bloquer que le contenu qui franchit une ligne claire vers un territoire menaçant ou haineux. Par exemple, "Je ne suis pas d'accord avec cette approche et je pense qu'elle est mal orientée" passerait à faible sensibilité, tout comme "C'est une terrible idée." Seul un contenu comme "Je te trouverai et je te ferai du mal" ou un discours de haine explicite échouerait.

Une faible sensibilité fonctionne bien pour les forums publics où le débat est attendu, les environnements de retour professionnel où la franchise est valorisée, et les communautés techniques où les gens discutent de sujets controversés. Le compromis est que certains contenus qui mettent les gens mal à l'aise peuvent encore passer.

La sensibilité moyenne est le paramètre par défaut et représente une approche équilibrée. À ce niveau, les garde-fous signalent les violations claires, y compris les insultes et le langage hostile, tout en permettant encore le désaccord professionnel et la critique constructive. Un message comme "Je ne suis pas d'accord avec votre raisonnement" passerait, mais "Tu es un idiot" ou "Des gens comme vous sont le problème" échouerait.

La sensibilité moyenne fonctionne bien pour la plupart des applications, y compris les systèmes de service client, les communications professionnelles, les outils collaboratifs et les plateformes sociales. Elle trouve un équilibre entre la possibilité d'un discours significatif et le maintien d'un environnement respectueux.

Une haute sensibilité crée l'environnement le plus strict en signalant tout contenu potentiellement toxique, y compris la légèreté de la grossièreté et le langage désinvolte. À ce niveau, même des contenus comme "Quoi qu'il en soit, mec" ou "C'est plutôt stupide" échoueraient. Seul un contenu respectueux et neutre passe à haute sensibilité.

Une haute sensibilité est appropriée pour les applications destinées aux enfants où vous avez besoin d'une protection maximale, les plateformes éducatives où vous souhaitez modéliser une communication respectueuse, les espaces sûrs et les communautés de

soutien où les gens ont besoin de se sentir en sécurité, et les contextes critiques en matière de conformité où tout problème potentiel doit être détecté.

Options de configuration

Le garde-fou de langage toxique accepte plusieurs options de configuration qui vous permettent d'ajuster son comportement pour votre cas d'utilisation spécifique.

TypeScript

```typescript
// Available options:
// sensitivity: "low", "medium", or "high" (default: "medium")
// model: model identifier (default: Claude 3.5 Haiku)
// temperature: 0-1, lower values = more consistent (default: 0.1)
// maxTokens: response length limit (default: 200)

await abv.guardrails.toxicLanguage.validate(text, {
  sensitivity: "medium",
  model: "model-name",
  temperature: 0.1,
  maxTokens: 200,
});
```

Python

```python
# Available options:
# sensitivity: "low", "medium", or "high" (default: "medium")
# model: model identifier (default: Claude 3.5 Haiku)
# temperature: 0-1, lower values = more consistent (default: 0.1)
# maxTokens: response length limit (default: 200)

abv.guardrails.toxic_language.validate(text, {
    "sensitivity": "medium",
    "model": "model-name",
    "temperature": 0.1,
    "maxTokens": 200
})
```

L'option de sensibilité est la plus importante et vous l'utiliserez fréquemment. Les options de modèle, de température et de maxTokens sont des paramètres avancés que vous n'aurez généralement pas besoin de modifier. Le modèle par défaut est optimisé pour les tâches de garde-fou et offre le meilleur équilibre entre vitesse, précision et coût. La température par défaut de 0.1 garantit des résultats cohérents. Le maxTokens par défaut de 200 est suffisant pour le champ d'explication.

Exemples du monde réel

Examinons des exemples concrets de la manière dont différents niveaux de sensibilité gèrent divers types de contenu. Comprendre ces modèles vous aidera à choisir la bonne sensibilité pour votre application.

Considérez un message comme "Je ne suis pas d'accord avec votre approche de ce problème." Il s'agit d'un désaccord professionnel et il passe à tous les niveaux de sensibilité. Le langage est neutre et respectueux malgré l'expression d'un désaccord.

Considérez maintenant "C'est une terrible idée et montre un mauvais jugement." Cela passe à faible et moyenne sensibilité car, bien que critique, cela se concentre sur l'idée plutôt que d'attaquer la personne. Cependant, cela pourrait échouer à haute sensibilité car "terrible" et "mauvais jugement" pourraient être considérés comme désinvoltes.

Un message comme "Tu ne sais pas de quoi tu parles" échoue à moyenne et haute sensibilité car il attaque directement la compétence de la personne. Cela pourrait passer à faible sensibilité puisque cela ne contient pas de menaces explicites ou de discours de haine, bien que ce soit limite.

Un contenu comme "Tu es un idiot" ou "Les gens comme toi sont le problème" échoue à tous les niveaux de sensibilité. Ce sont des attaques personnelles claires sans valeur constructive.

Enfin, des menaces explicites comme "Je te trouverai et je te ferai du mal" échouent à tous les niveaux de sensibilité avec une confiance maximale. C'est un contenu toxique sans ambiguïté.

Modèles de mise en œuvre

Voici comment vous utiliseriez typiquement la détection de langage toxique dans différentes parties de votre application.

Pour la validation des entrées, vous vérifiez les messages des utilisateurs avant de les envoyer à votre IA ou de les afficher à d'autres utilisateurs :

TypeScript

```typescript
async function validateUserMessage(message: string): Promise<boolean> {
  const result = await abv.guardrails.toxicLanguage.validate(
    message,
    { sensitivity: "medium" }
  );

  if (result.status === "pass") {
    return true;
  }

  // Log the reason for monitoring, but don't expose it to the user
  console.log("Blocked message:", result.reason);
  return false;
}

// Usage in your message handler
if (await validateUserMessage(userInput)) {
  await processMessage(userInput);
} else {
  return { error: "Your message violates our community guidelines." };
}
```

Python

```python
async def validate_user_message(message: str) -> bool:
    result = await abv.guardrails.toxic_language.validate_async(
        message,
        {"sensitivity": "medium"}
    )

    if result["status"] == "pass":
        return True

    # Log the reason for monitoring, but don't expose it to the user
    print(f"Blocked message: {result['reason']}")
    return False

# Usage in your message handler
if await validate_user_message(user_input):
    await process_message(user_input)
else:
    return {"error": "Your message violates our community guidelines."}
```

Pour la validation des sorties, vous vérifiez les réponses générées par l'IA avant de les montrer aux utilisateurs :

TypeScript

```typescript
async function generateSafeResponse(prompt: string): Promise<string> {
  // Generate initial response
  let response = await callAI(prompt);

  // Validate the response
  const validation = await abv.guardrails.toxicLanguage.validate(
    response,
    { sensitivity: "high" }
  );

  // If toxic, regenerate with explicit safety instruction
  if (validation.status === "fail") {
    response = await callAI(
      prompt + "\n\nIMPORTANT: Respond in a professional, respectful tone."
    );
  }

  return response;
}
```

Python

```python
async def generate_safe_response(prompt: str) -> str:
    # Generate initial response
    response = await call_ai(prompt)

    # Validate the response
    validation = await abv.guardrails.toxic_language.validate_async(
        response,
        {"sensitivity": "high"}
    )

    # If toxic, regenerate with explicit safety instruction
    if validation["status"] == "fail":
        response = await call_ai(
            f"{prompt}\n\nIMPORTANT: Respond in a professional, respectful tone."
        )

    return response
```

Pour gérer les cas ambigus, vous pourriez mettre en œuvre une file d'attente de révision pour les résultats incertains :

TypeScript

```typescript
async function handleUserContent(content: string) {
  const result = await abv.guardrails.toxicLanguage.validate(
    content,
    { sensitivity: "medium" }
  );

  if (result.status === "pass") {
    // Content is clearly acceptable
    await publishContent(content);
  } else if (result.status === "fail" && result.confidence > 0.8) {
    // High-confidence violation, auto-reject
    await rejectContent(content, "Community guidelines violation");
  } else {
    // Low confidence or unsure - flag for human review
    await flagForModeration(content, result);
  }
}
```

Python

```python
async def handle_user_content(content: str):
    result = await abv.guardrails.toxic_language.validate_async(
        content,
        {"sensitivity": "medium"}
    )

    if result["status"] == "pass":
        # Content is clearly acceptable
        await publish_content(content)
    elif result["status"] == "fail" and result["confidence"] > 0.8:
        # High-confidence violation, auto-reject
        await reject_content(content, "Community guidelines violation")
    else:
        # Low confidence or unsure - flag for human review
        await flag_for_moderation(content, result)
```

Optimisation des performances

Puisque la détection de langage toxique utilise l'IA, cela prend de une à trois secondes par vérification et consomme des jetons. Vous pouvez optimiser les performances en effectuant d'abord une vérification rapide basée sur des règles pour attraper les violations évidentes avant de faire l'appel coûteux à l'IA.

TypeScript

```typescript
async function efficientToxicCheck(text: string): Promise<boolean> {
  // Quick check for explicitly forbidden terms (under 10ms, free)
  const quickCheck = await abv.guardrails.containsString.validate(
    text,
    {
      strings: ["explicit-slur", "forbidden-term"],
      mode: "none",
    }
  );

  // If quick check fails, no need for expensive AI check
  if (quickCheck.status === "fail") {
    return false;
  }

  // Only run AI check if quick check passed
  const deepCheck = await abv.guardrails.toxicLanguage.validate(text);
  return deepCheck.status === "pass";
}
```

Python

```python
async def efficient_toxic_check(text: str) -> bool:
    # Quick check for explicitly forbidden terms (under 10ms, free)
    quick_check = await abv.guardrails.contains_string.validate_async(
        text,
        {
            "strings": ["explicit-slur", "forbidden-term"],
            "mode": "none"
        }
    )

    # If quick check fails, no need for expensive AI check
    if quick_check["status"] == "fail":
        return False

    # Only run AI check if quick check passed
    deep_check = await abv.guardrails.toxic_language.validate_async(text)
    return deep_check["status"] == "pass"
```

Meilleures pratiques de sécurité

Ne jamais exposer le champ de raison aux utilisateurs finaux. La raison explique pourquoi le contenu a échoué à la validation, et exposer cette information aide les acteurs malveillants à apprendre comment contourner vos garde-fous. Au lieu de cela, utilisez des messages d'erreur génériques tout en enregistrant la raison détaillée en interne pour le suivi et l'amélioration.

TypeScript

```typescript
// Bad - exposes validation logic
if (result.status === "fail") {
  return { error: result.reason };  // Don't do this!
}

// Good - generic message, internal logging
if (result.status === "fail") {
  logger.info("Blocked toxic content", { reason: result.reason });
  return { error: "Your message violates our community guidelines." };
}
```

Python

```python
# Bad - exposes validation logic
if result["status"] == "fail":
    return {"error": result["reason"]}  # Don't do this!

# Good - generic message, internal logging
if result["status"] == "fail":
    logger.info(f"Blocked toxic content: {result['reason']}")
    return {"error": "Your message violates our community guidelines."}
```

Choisir la bonne sensibilité

Voici un cadre de décision pour choisir la sensibilité en fonction de votre type d'application. Si vous construisez pour des enfants ou des populations vulnérables, utilisez toujours une haute sensibilité. Le potentiel de préjudice causé par l'autorisation de contenu toxique l'emporte de loin sur le coût des faux positifs.

Si vous construisez une application orientée client comme un service client, des réseaux sociaux ou des outils collaboratifs, une sensibilité moyenne est généralement appropriée. Elle détecte les violations claires tout en permettant des désaccords professionnels.

Si vous construisez pour des publics professionnels ou techniques où un débat robuste est attendu, envisagez une faible sensibilité. Les forums techniques, les systèmes de révision de code et les outils de retour professionnel bénéficient de la possibilité d'exprimer des opinions fortes.

Vous pouvez également ajuster la sensibilité en fonction du contexte de l'utilisateur. Les utilisateurs authentifiés avec un bon historique peuvent bénéficier d'une sensibilité plus faible tandis que les utilisateurs anonymes obtiennent une sensibilité plus élevée. Les utilisateurs qui s'identifient comme mineurs obtiennent automatiquement une haute sensibilité, quelle que soit la configuration par défaut.

Prochaines étapes

La barrière de langage toxique est souvent utilisée en parallèle avec d'autres barrières pour une validation complète du contenu. Envisagez de la combiner avec la détection de langage biaisé pour une solution de sécurité du contenu plus complète. Vous voudrez peut-être également utiliser une chaîne de contenu pour attraper rapidement les termes interdits explicites avant d'effectuer le contrôle de langage toxique plus coûteux.

Pour des conseils d'implémentation plus détaillés, consultez la documentation des meilleures pratiques qui couvre les stratégies d'optimisation, la gestion des erreurs et les approches de surveillance.

8.4. asdsadasdas (cloné)

Obtenir des gâteaux

Try it

Obtenir un gâteau par son ID

GET https://api.cakes.com

Request

BODY PARAMETERS

id String required

ID du gâteau à obtenir

Nom du paramètre Object optional

Nom du paramètre Object optional

JS

```
var myHeaders = new Headers();
myHeaders.append("Accept", "application/json");
myHeaders.append("Content-Type", "application/json");

var raw = JSON.stringify({
   "id": "String"
});

var requestOptions = {
   method: 'GET',
   headers: myHeaders,
   body: raw,
   redirect: 'follow'
};

fetch("https://api.cakes.com", requestOptions)
   .then(response => response.text())
   .then(result => console.log(result))
   .catch(error => console.log('error', error));
```

Responses 200

```
{
  "name": "Nom du gâteau",
}
```

