API Specification

Version 1.7

The purpose of this document is to provide a description of Intelligent Payments' gateway API, and is specifically intended for merchant integration purposes.

1 Integration Options

The Intelligent Payments API offers several types of integrations for payment processing. These include:

- ❏ PCI Compliant Payment Form Integration (hosted payment pages);
- ❏ PCI Compliant Payment Form Plugin (For third party shopping cart platforms);
- ❏ Non PCI Compliant Direct API Integration (Merchant uses own payment form);
- ❏ Virtual Terminal Integration (Within merchant's backoffice systems);
- ❏ Standalone Backoffice + Virtual Terminal.

The following section will explain the use of each of the features above, one by one.

1.1 PCI Compliant Payment Form Integration

- ❏ The Payment Form is called by the merchant shopping cart platform directly after obtaining a valid session token from the IPG Session Token Provider;
- ❏ The Payment Form is PCI Compliant as it resides within the IPG platform, which sits in a Level 1 PCI Compliant environment;
- ❏ The Payment Form has a pre-defined layout and presentation. However these features may be customised by introducing the CSS code within the session token request; (Customisation might incur extra fees in your bill)
- ❏ The Payment Form is to be used ONLY for E-Commerce Type Card Payments;
- ❏ The Payment Form will also provide 3D-Secure authentication if the 3D-Secure is enabled on the backoffice settings or the merchant forces the secure payment in the Session Token Request.

1.2 PCI Compliant Payment Form Plugin

- ❏ This Payment Form contains the same features as specified in the previous section (Integration) however the Payment Form is installed as a plugin within the merchant's shopping cart platform;
- ❏ The plugin for the IPG Payment Form may be obtained from https://www.sellxed.com/shop/en/usd/extensions/module/payment-service-provider/intelligent-payments.html

1.3 Non PCI Compliant Direct API Integration

- The merchant takes total control of what is displayed to the customers as a Payment Page (the merchant develops own payment form);
- The merchant will be able to call the IPG services directly to the API;
- Direct API Integration will also provide 3D-Secure authentication if the 3D-Secure is enabled on the backoffice settings or the merchant forces the secure payment in the Session Token Request;
- With Direct API Integration the merchant is also able to choose which payment operation to perform. These payment operations include:
  - ***TOKENIZE*** *- Securely store card details in IPG Database and obtain Card Token to be used for subsequent transactions, if needed;*
  - ***AUTH*** *- Perform an authorisation only type payment (would need to be captured in a separate operation);*
  - ***CAPTURE*** *- Perform a capture operation on a previous AUTH type payment;*
  - ***VOID*** *- Undo a previous AUTH type payment (a voided AUTH cannot be captured)*
  - ***PURCHASE*** *- Perform a full authorise and capture payment in one-step. (PURCHASE type transactions cannot be voided)*
  - ***REFUND*** *- Perform a refund (partial or full) on a previously captured transaction.*
- For each of the above operations, a separate session token is to be obtained from the IPG session token provider.

1.4 Virtual Terminal Integration

- The merchant integrates the IPG Virtual terminal within the merchant backoffice system;
- The Virtual Terminal may only be used for MOTO type payments;
- The Virtual Terminal comes in a pre-set layout and presentation and may not be modified or customised.

1.5 Standalone Backoffice + Virtual Terminal

- The Standalone Backoffice + Virtual Terminal does not require any integration work from the merchant. It may only be used for MOTO type payments.
- On the Standalone Backoffice + Virtual Terminal, the following operations may be processed:
  - ***AUTH*** *- Perform an authorisation only type payment (would need to be captured in a separate operation);*
  - ***CAPTURE*** *- Perform a capture operation on a previous AUTH type payment;*
  - ***VOID*** *- Undo a previous AUTH type payment (a voided AUTH cannot be captured)*
  - ***PURCHASE*** *- Perform a full authorise and capture payment in one-step. (PURCHASE type transactions cannot be voided)*
  - ***REFUND*** *- Perform a refund (partial or full) on a previously captured transaction.*

2 Gateway Interface

2.1 Addresses

Integration Addresses:
Session Token Request URL: https://apiuat.test.intelligent-payments.com/token
Payment Operation Action URL: https://apiuat.test.intelligent-payments.com/payments

Production Addresses:
Session Token Request URL: https://api.intelligent-payments.com/token
Payment Operation Action URL: https://api.intelligent-payments.com/payments

2.2 HTTP Specification

- Protocol: https
- Method: POST
- Content Type: application/x-www-form-urlencoded

2.3 Example HTTP Request

POST: https://api.intelligent-payments.com/token
Host: api-turnkeyuat.test.myriadpayments.com
Content-Type: application/x-www-form-urlencoded
Content-Length: 415

POST data:
merchantId=160001&action=PURCHASE&password=*to_be_provided*&allowOriginUrl=www.merchantsite.com&timestamp=1459767453376&channel=ECOM&userDevice=DESKTOP&amount=25.96&currency=GBP&country=DE&paymentSolutionId=500&specinCreditCardToken=123456781111&customerId=9876543&brandId=670&merchantNotificationUrl=https%3A%2F%2Fwww.posttestserver.com%2Fpost.php%2FipgTesting%3Fdir%3DJCTesting&merchantLandingPageUrl=https://www.merchantsite.com%2FlandingPage&forceSecurePayment=true

3 Performing API Operations

3.1 Tokenize

3.1.1 Intro:

The card token is the only card identifier accepted as an input when a payment operation takes place. Therefore in order to avoid having merchants handling card data directly, the card data is first tokenized. The customer fills a form with all the required card data values and submits it. Upon submission, the merchant needs to first call the session token request and then, using the obtained session token, call the card tokenization request, which will return the card token to be used for subsequent transactions.

3.1.2 Requesting the Session Token

Input:

Parameter	Data Type	Mandatory Y/N/C	Description
merchantId	Integer(18)	Y	Merchant id that identifies the merchant in IPG.
password	String(64)	Y	Merchant password in IPG.
action	String(enum)	Y	Action, must be *TOKENIZE*.
timestamp	Integer(18)	Y	Milliseconds since 1970-01-01 00:00:00
customerId	String(20)	N	Customer identifier in merchant system. Generated by IPG if merchant doesn't provide it.
allowOriginUrl	String(253)	Y	Merchant's web page URL, the one that would make the 2nd request using the token (CORS headers will allow only this origin).

Output

Parameter	Data Type	Description
token	String(40)	One time use Session token.
merchantId	Integer(18)	Merchant identifier in IPG.
result	String	example, "success"
resultId	String	example, "38e2d66e-6c7b-4152-bf87-fdbecb7cfa2f"

3.1.3 Performing the Tokenize Action

Input

Parameter	Data Type	Mandatory Y/N/C	Description
merchantId	Integer(18)	Y	Merchant id that identifies the merchant in IPG.
token	String(40)	Y	One time session token received in the first step.
number	String(100)	Y	Card number.
nameOnCard	String(150)	Y	Cardholder name.
expiryMonth	String(2)	Y	Card expiration month, e.g. “04”.
expiryYear	String(4)	Y	Card expiration year, e.g. “2019”.
startMonth	String(2)	C	Card issued month, e.g. “04”. Mandatory if card type is Maestro, optional in other cases.
startYear	String(4)	C	Card issued year, e.g. “2014”. Mandatory if card type is Maestro, optional in other cases.
issueNumber	String(2)	C	Card issue number. Mandatory if card type is Maestro.
cardDescription	String(50)	N	Any data can be inserted here (eg merchant’s internal card’s sequence id to associate it with the given card number) for reconciliation purpose.

Output:
Response is a JSON with the following format:

Parameter/Label	Data Type	Description
merchantId	Integer(18)	Merchant identifier in IPG.
cardToken	String(100)	Credit card token. This is the only valid credit card identifier allowed in further actions (PURCHASE,AUTH etc,...).
customerId	String(20)	Customer identifier in merchant system. Generated by IPG if merchant didn't provide it.
cardType	Integer(enum)	Card type code: • 100 : MAESTRO • 200 : MASTERCARD CREDIT • 300 : MASTERCARD DEBIT • 400 : VISA CREDIT • 500 : VISA DEBIT • 600 : VISA ELECTRON
cardIssuer	String	Card issuer name.
country	String(enum)	Card issuer country. The value is the alpha-2 code as defined in ISO 3166 standard (visit http://www.iso.org/iso/country_codes).

Example response:

{"result": "success", "cardToken" : "4520123498701", "cardType" : 400, "cardIssuer" : "HSBC","country" : "GB", "customerId" : "CusId-123456"}

In case an error occurs and the card couldn't be tokenized, the output will be:

Parameter/Label	Data Type	Description
merchantId	Integer(18)	Merchant identifier in IPG.
errors	List	List of errors.

Example error response:

{"result":"failure","errors" : ["invalid_card_number","invalid_expiry_year"]}

List of errors:

AUTHENTICATION ERRORS	
Error code	**Description**
error_no_merchant_id	Merchant ID parameter was not provided
error_access_denied	The IP address for this merchant is not white-listed
error_tokeniser_api_is_off	Tokenizer API for this merchant is switched off
invalid_merchant_password	Invalid merchant password
VALIDATION ERRORS	
Error code	**Description**
invalid_card_number	Invalid card number: must be between 12 and 19 digits and pass Luhn check
invalid_name_on_card	Cardholder name was not provided
invalid_card_type_code	Invalid card type code: must be digits only
invalid_bin_range	Could not retrieve card type code from DB - BIN range of the card number may be invalid
invalid_expiry_month	Invalid card expiry month: must be 2 digits
invalid_expiry_year	Invalid card expiry year: must be 4 digits
invalid_expiry_date	Expiry date is in the past
invalid_start_month	Invalid issue date month: must be 2 digits
invalid_start_year	Invalid issuer date year: must be 4 digits
invalid_start_date	Issue date is in the future
invalid_issue_number	Invalid issue number: must be digits only
invalid_active_yn_value	Invalid "activeYN" value: must be "0" or "1"

invalid_customer_id	Customer ID was not provided
OTHER ERRORS	
Error code	**Description**
general_error	Unexpected system error, please contact the service provider

3.2 Auth/Purchase

3.2.1 Intro

The difference between Auth and Purchase is the final status of the transaction. Auth transactions only “block” the funds from the customer card account and funds are not moved and settled until the Capture is done. Purchase transactions perform the two operations (Auth and Capture) at one go.

- ❏ Action is either AUTH or PURCHASE;
- ❏ While PURCHASE results in a finished payment, AUTH requires an additional stage: CAPTURE (see Capture section in this document);
- ❏ Transaction status value after PURCHASE can be CAPTURED or ERROR or DECLINED;
- ❏ Transaction status value after an AUTH transaction can be NOT_SET_FOR_CAPTURE or ERROR or DECLINED;
- ❏ NOT_SET_FOR_CAPTURE means the payment is authorised and the customer money is blocked but requires confirmation (capture);

3.2.2 Requesting the Session Token

Input

Parameter	Data Type	Mandatory Y/N/C	Description
merchantId	Integer(18)	Y	Merchant id that identifies the merchant in IPG.
merchantTxId	String(50)	N	Transaction identifier (or Order ID) in merchant system. If not provided, IPG will generate it.
password	String(64)	Y	Merchant password in IPG.
allowOriginUrl	String(253)	Y	Merchant's web page URL, the one that would make the 2nd request using the token (CORS headers will allow only this origin)
action	String(enum)	Y	Possible actions: • “AUTH” • “PURCHASE”
timestamp	Integer(18)	Y	Milliseconds since 1970-01-01 00:00:00
customerId	String(20)	C	Customer identifier in merchant system or identifier generated by IPG in credit card tokenization. Mandatory if payment method is Credit Cards, optional if other method (IPG would generate a one time use customer id if not provided).
operatorId	String(20)	N	If the operation is performed by the merchant’s operator or agent on behalf of the end customer

			and if merchant wants to track which operator performed which transaction this field should be filled in.
brandId	Integer(18)	N	Contact IPG to get your brand ID associated. If not provided default value will be used.
channel	String(enum)	Y	Transaction channel which decides if VT or Cashier will be used in this transaction. Possible values: ● "ECOM" (for e-commerce type transactions) ● "MOTO" (for card not present transactions)
userDevice	String(enum)	N	Type of device used: ● "MOBILE" ● "DESKTOP" ● "UNKNOWN" (default value)
userAgent	String(1024)	N	The user agent of the browser from which the transaction was performed.
amount	BigDecimal(15.2 or 15.3)	Y	Transaction amount. Includes tax, shipping, surcharge and discount amounts.
taxAmount	BigDecimal(15.2 or 15.3)	N	Tax amount.
shippingAmount	BigDecimal(15.2 or 15.3)	N	Shipping amount.
chargeAmount	BigDecimal(15.2 or 15.3)	N	Charge amount.
discountAmount	BigDecimal(15.2 or 15.3)	N	Discounts applied.
currency	String(enum)	Y	Currency alphabetic code as defined in ISO 4217 standard. http://www.iso.org/iso/home/standards/currency_codes.htm
country	String(enum)	Y	This will be the country where the transaction takes place if *customerAddressCountry* is not provided. The value is the alpha-2 code as defined in ISO 3166 standard. http://www.iso.org/iso/country_codes
paymentSolutionId	Integer(18)	C	Payment solution identifier in IPG. Eg Credit card id = 500. Must be provided always for Direct API integration. Can be omitted in Payment Form Integration (Cashier UI) in this case all of the suitable payment solutions will be offered to the customer in the UI HTML form.
language	String(enum)	N	Language. Customer's prefered language. The value must be a 2 letter code as defined in ISO 639-1 standard (small letters only like: en, pl, es) http://www.iso.org/iso/catalogue_detail?csnumber=22109 This value will be sent to some payment acquirers so they can show their site to the customer in the language requested. If not provided, language will be the payment acquirer's default.
s_text1, s_text2... s_text5	String(200)	N	5 text fields for general use, example "late payment"

d_date1, d_date2... d_date5	Date(DD/MM/YYYY hh:mm:ss)	N	5 date fields for general use. The time part can be omitted, resulting in 00:00:00
b_bool1, b_bool2... b_bool5	boolean (true, false)	N	5 boolean fields for general use. The accepted representation is "true" and "false"
n_num1, n_num2... n_num5	BigDecimal(7.2)	N	5 numeric fields for general use. Use decimal dot "." and not the comma ",", avoid thousand separators.
merchantNotificationUrl	String(200)	Y	URL where the merchant will receive transaction status messages.
merchantLandingPageUrl	String(200)	N	Merchant success/error landing page URL.
firstTimeTransaction	boolean (true, false)	N	Customer's first time transaction flag. Default: ● if customerId provided, false ● if customerId not provided, true
customerDocumentType	String(enum)	N	Type of document used by the customer to identify himself in merchant's side. ● PASSPORT ● NATIONAL_ID ● DRIVING_LICENSE ● UNIQUE_TAXPAYER_REFERENCE ● OTHER
customerDocumentNumber	String(30)	C	Customer document number. Mandatory if customerDocumentType provided.
merchantReference	String(200)	N	Customer merchant reference.
customerFirstName	String(50)	N	Customer first name.
customerLastName	String(100)	N	Customer last name.
customerSex	String(enum)	N	Customer sex: ● M (male) ● F (female)
customerDateOfBirth	Calendar (DD/MM/YYYY)	N	Customer date of birth.
customerRegistrationDate	Calendar (DD/MM/YYYY)	N	Customer registration date on merchant's site.
customerEmail	String(60)	N	Customer email address.
customerPhone	String(100)	N	Customer phone number.
customerIPAddress	String(39)	N	Customer IP address from where purchase is made. Only IPv4 supported.
customerAddressHouseName	String(50)	N	Customer address house name.
customerAddressHouseNumber	String(5)	N	Customer address house number.
customerAddressFlat	String(5)	N	Customer address flat.

customerAddressStreet	String(50)	N	Customer address street.
customerAddressCity	String(50)	N	Customer address city.
customerAddressDistrict	String(50)	N	Customer address district.
customerAddressPostalCode	String(30)	N	Customer address postal code.
customerAddressCountry	String(enum)	N	Customer address country. If provided, it will be considered the country where the transaction takes place The value is the alpha-2 code as defined in ISO 3166 standard. http://www.iso.org/iso/country_codes
customerAddressState	String(40)	N	Customer address state.
customerAddressPhone	String(100)	N	Customer address phone.
customerShippingAddressHouseName	String(50)	N	Customer shipping address house name.
customerShippingAddressHouseNumber	String(5)	N	Customer shipping address house number.
customerShippingAddressFlat	String(5)	N	Customer shipping address flat.
customerShippingAddressStreet	String(50)	N	Customer shipping address street.
customerShippingAddressCity	String(50)	N	Customer shipping address city.
customerShippingAddressDistrict	String(50)	N	Customer shipping address district.
customerShippingAddressPostalCode	String(30)	N	Customer shipping address postal code.
customerShippingAddressCountry	String(enum)	N	Customer shipping address country. The value is the alpha-2 code as defined in ISO 3166 standard. http://www.iso.org/iso/country_codes
customerShippingAddressState	String(40)	N	Customer shipping address state.
customerShippingAddressPhone	String(100)	N	Customer shipping address phone.
customerBillingAddressHouseName	String(50)	N	Customer billing address house name.
customerBillingAddressHouseNumber	String(5)	N	Customer billing address house number.
customerBillingAddressFlat	String(5)	N	Customer billing address flat.
customerBillingAddressStreet	String(50)	N	Customer billing address street.
customerBillingAddressCity	String(50)	N	Customer billing address city.
customerBillingAddressDistrict	String(50)	N	Customer billing address district.

customerBillingAddressPostalCode	String(30)	N	Customer billing address postal code.
customerBillingAddressCountry	String(enum)	N	Customer billing address country. The value is the alpha-2 code as defined in ISO 3166 standard. http://www.iso.org/iso/country_codes
customerBillingAddressState	String(40)	N	Customer billing address state.
customerBillingAddressPhone	String(100)	N	Customer billing address phone.
payerFirstName	String(50)	N	Payer first name. In case no payer info provided, customer info will be taken instead.
payerLastName	String(100)	N	Payer last name.
payerEmail	String(60)	N	Payer email.
payerDateOfBirth	Calendar (DD/MM/YYYY)	N	Payer date of birth.
payerPhone	String(100)	N	Payer phone.
payerDocumentType	String(enum)	N	Type of document used by the payer to identify himself in merchant's side. ● PASSPORT ● NATIONAL_ID ● DRIVING_LICENSE ● UTR ● OTHER
payerDocumentNumber	String(30)	C	Payer document number. Mandatory if payerDocumentType provided.
payerCustomerId	String(20)	N	If the payer is a registered customer of the merchant this is his customer identifier merchant system.
forceSecurePayment	boolean (true, false)	N	For Credit Card transactions: forces or skips 3DSecure process no matter the routing rules. If not provided or null, the 3DS routing rules take over.
processUnknownSecurePayment	boolean (true, false)	N	For Credit Card transactions in which 3D secure is involved, decides the way of processing "U" (Unkown) responses from 3DS. If true, 3DSecure authentication response U is considered a success and auth/sale is requested. If the parameter value is false, transaction fails at that stage. If not provided or null, value from 3DSecure routing rules is used.
numberOfInstallments	Integer(2)	N	Now deprecated, replaced by `selectedInstallmentsPlanId`, but still supported. Number of installments for this payment, if payment solution supports them.
selectedInstallmentsPlanId	String(30)		If the value is empty, whitespace or "`noInstallments`", then a single payment transaction is assumed. If sent with the session token request, it determines the choice and (front-end user) cannot change it later. Normally

			however it's expected in the second request, after the user is presented a selection of installment plans. Has to match one of the ids provided in the JSON of `availableInstallments` value.

availableInstallments			JSON String, the installment plans available for the user to choose. The list needs to be stored under the "*data*" key, and the minimum properties of each plan are id, label and description. The label and description should contain user-friendly information about the plan, as they are going to be presented in the UI. Their language should match that of the rest of the page. Note that the id can be numeric and match the number of installments. An important fact is that the list needs to contain the single payment option if it's allowed by the merchant. In the following example, the "`numberOfInstallments`" and "`schemaName`" are acquirer specific parameters, so they do not necessarily apply to all cases, please consult customer service for details. The list does not include the single payment, so the dropdown is not going to offer it either.
			
			Example JSON value of `availableInstallments` sent in the request for session token:
			`{`
			`"data" : [`
			`{`
			`"id" : "max2",`
			`"label" : "2 x 33 EUR, total: 66",`
			`"description" : "5% TAE",`
			`"numberOfInstallments" : 2,`
			`"schemaName" : "maxcommerce"`
			`},`
			`{`
			`"id" : "max1_3",`
			`"label" : "1 x 30 + 3 x 33 EUR, total: 129 EUR",`
			`"description" : "4.5% TAE",`
			`"numberOfInstallments" : 4,`
			`"schemaName" : "maxcommerce"`
			`}`
			`]`
			`}`

specinCreditCardToken	String(100)	N	Credit card token. Applies to Credit/Debit Card payments only. See "*Tokenize*" section.
specinProcessWithoutCvv2	boolean (true, false)	N	Allows to process Credit Cards transactions excluding security code CVV2. Default: false. This requires prior authorization by IPG and acquirer.
bankMid	String(50)	N	If merchant wishes to control which acquirer bank MID will be used for any given transaction this is the place to put it.This is considered advanced feature only for very specific scenarios.
storeCard	boolean (true, false)	N	Does not store the card for the customer if set to false. Stores card if true. If unspecified the card is stored by default.

Output

Successful output example:


```
{
"result":"success",
"merchantId":123,
"token":"abcd-1234-abcd-1234"
}
```


Parameter	Data Type	Description
result	String(40)	success
token	String(40)	One time use Session token.
merchantId	Integer(18)	Merchant identifier in IPG.

Failure output example:


```
{
"result":"failure",
"errors":["Access denied"]
}
```


Parameter	Data Type	Description
result	String(40)	failure
errors	Array	List of issues

3.2.3 Executing Payment Operation

Input

Parameter	Data Type	Mandatory Y/N/C	Description
merchantId	Integer(18)	Y	Merchant id that identifies the merchant in IPG.
token	String(40)	Y	Session token received in the first step.
freeText	String(200)	N	Merchant free text or comments. If not received in step 1.
numberOfInstallments	Integer(2)	N	Number of installments for this payment, if payment solution supports them. This value will be ignored if it was already provided in step 1.
customerIPAddress	String(39)	N	Customer IP address from where purchase is made. Only IPv4 supported. This value will be ignored if it was already provided in step 1.
fraudToken	String(50)	C	Antifraud token in case an antifraud tool has been executed before and analysis identifier is required by payment acquirer. This parameter value is mandatory only for transactions conducted in LATAM countries and only in case merchant wishes the transaction to be conducted as direct integration (server-server) as opposed to browser-redirection based integration.
paymentSolutionId	Integer(18)	C	Payment solution identifier in IPG. If it was not provided in Requesting the Session Token parameters it has to be provided here.
specinCreditCardCVV	String(5)	C	Credit card CVV. If payment solution is credit card, channel ECOM. Configurable by merchant, card type,... in database.

Output

The output response is in JSON. Depending on whether customer redirection is required, the JSON format is as follows:

Redirection required (in case of 3DS flow transactions):

{
"result":"redirection",
"merchantId":1231231,
"merchantTxId":"abc-123",
"txId":123,
"redirectionUrl":"https://mpi.bank.com/123123123-abc-123123123"
}

Parameter/Label	Data Type	Description
result	String (enum)	Static value: “redirection”
merchantId	Integer(18)	Merchant identifier in IPG.
merchantTxId	String(50)	Transaction identifier (or Order ID) in merchant system. Generated by IPG if merchant didn’t provide it in first step.
txId	Integer(18)	Transaction id in IPG.
redirectionUrl	String(URL)	Merchant must redirect customer browser to this URL.

Redirection is not required, transaction ends here (no 3DS flow transactions):


```
{
"result":"success",
"merchantId":123,
"merchantTxId":"abc-123",
"txId":"123",
"acquirerTxId":"0009312",
"amount":12.50,
"currency":"GBP",
"customerId":"mgn-456",
"action":"sale",
"pan":"4111111111111",
"brandId":3,
"paymentSolutionId":500,
"freeText":"Addded 10% discount on the item",
"language":"en",
"acquirerAmount":16.7,
"acquirerCurrency":"EUR",
"paymentSolutionDetails":{
   authCode:"1234"
}
```

}

Parameter/Label	Data Type	Description
merchantId	Integer(18)	Merchant identifier in IPG.
merchantTxId	String(50)	Transaction identifier (or Order ID) in merchant system. Generated by IPG if merchant didn't provide it in first step.
txId	Integer(18)	Transaction id in IPG.
acquirerTxId	String(100)	Transaction identifier in acquirer system, if acquirer returns it.
amount	BigDecimal (15.2 or 15.3)	Transaction amount. Includes tax, shipping, charge and discount amounts.
currency	String(enum)	Currency alphabetic code as defined in ISO 4217 standard. http://www.iso.org/iso/home/standards/currency_codes.htm
customerId	String(20)	Customer identifier in merchant system or generated by IPG if merchant didn't provide it in first step and payment method is not Credit Cards.
action	String(enum)	Action executed as provided in session token request call.
pan	String(100)	Customer account number or identifier used in the transaction. If payment solution is Credit Cards, it is the specinCreditCardToken received in first step.
brandId	Integer(18)	Brand Id as received in session token request or default value if not provided there.
paymentSolutionId	Integer(18)	Payment solution id.
freeText	String(200)	Merchant free text.
language	String(enum)	Language. Customer's prefered language. The value must be a 2 letter code as defined in ISO 639-1 standard. http://www.iso.org/iso/catalogue_detail?csnumber=22109 This value will be sent to some payment acquirers so they can show their site to the customer in the language requested. If not provided, language will be the payment acquirer's default.

acquirerAmount	BigDecimal (15.2 or 15.3)	Amount processed by payment acquirer. May be different than initial amount requested.
acquirerCurrency	String(enum)	Transaction currency in payment acquirer side, may be different than transaction currency requested (e. g. if a currency conversion applied). Currency alphabetic code as defined in ISO 4217 standard. http://www.iso.org/iso/home/standards/currency_codes.htm
paymentSolutionDetails		JSON block. Specific payment solution content, it may vary depending on the payment solution. See ***"Payment Solution Details"*** section for further info.

Payment Solution Details:

The paymentSolutionDetails section forms part of the JSON response if redirection is not required and only applies to some payment solutions as can be seen below:

Parameter/Label	Data Type	Payment Solution	Description
authCode	String	Credit Cards	Transaction authorisation code as received from acquirer.

Failure Response example:

```
{
"result":"failure",
"merchantId":1231231,
"merchantTxId":"abc-123",
"txId":123,
"errors":[
 "insufficient funds"
]
}
```

3.2.4 Customer Redirection (when applicable):

In some cases the payment cannot be completed in one step and customer redirection to payment acquirer site is required for authentication (e. g. 3DSecure process). This adds an extra security layer to the transaction. If the payment execution response contains the parameter ***redirectionUrl***, the merchant must redirect the customer browser to the specified URL (hosted by IPG or directly by the card issuing bank) so the customer can log in/confirm identification into the payment acquirer site. Once the customer has completed the action required by the acquirer (log in, registration, etc.), IPG will process the payment and redirects the customer back to merchant URL provided as ***merchantLandingPageUrl*** specified in the session token request. The merchant will be informed about transaction status in a server-to-server call to the ***merchantNotificationUrl***, also specified in the session token request.

3.3 Refund/Capture/Void

Capture and Void operations can only be performed on Auth type transactions.

3.3.1 Refund

Based on the amount captured and previous refunds over the same original payment, the system decides if it is a partial refund, a full refund or if the refund is exceeding the captured amount, in which case the refund is rejected. Refunds can only be performed on captured transactions.

3.3.1.1 Requesting the Session Token for Refund

Input:

Parameter	Data Type	Mandatory Y/N/C	Description
merchantId	Integer(18)	Y	Merchant id that identifies the merchant in IPG.
originalMerchantTxId	String(50)	Y	Merchant transaction id of the transaction that is going to be refunded.
allowOriginUrl	String(253)	Y	Merchant's web page URL, the one that would make the 2nd request using the token (CORS headers will allow only this origin)
action	String(enum)	Y	Action required, must be REFUND.

timestamp	Integer(18)	Y	Milliseconds since 1970-01-01 00:00:00
password	String(64)	Y	Merchant password in IPG.
originalTxId	Integer(18)	N	Transaction id of the initial transaction in IPG.
agentId	String(18)	N	Id of the merchant's representative that requests the refund.
amount	BigDecimal (10.2 or 10.3) BigDecimal (15.2 or 15.3)	Y	Amount to refund.

Output

Parameter	Data Type	Description
result	String (enum)	Static value: "succes"
token	String(40)	One time use Session token.
merchantId	Integer(18)	Merchant identifier in IPG.

3.3.1.2 Execute Refund

Input

Parameter	Data Type	Mandatory Y/N/C	Description
merchantId	Integer(18)	Y	Merchant id that identifies the merchant in IPG.
token	String(40)	Y	Session token received in the first step.

Output

Response is a JSON with the following format:

Parameter/Label	Data Type	Description
merchantId	Integer(18)	Merchant identifier in IPG.
originalMerchantTxId	String(50)	Transaction identifier of the original transaction in merchant system.
originalTxId	Integer(18)	Transaction id of the original transaction in IPG.
txId	Integer(18)	Transaction id in IPG.
amount	BigDecimal (15.2 or 15.3)	Amount refunded.
currency	String(enum)	Currency alphabetic code as defined in ISO 4217 standard. http://www.iso.org/iso/home/standards/currency_codes.htm
customerId	String(20)	Customer identifier in merchant system. Generated by IPG if merchant didn't provide it when original payment was created.
action	String(enum)	Action performed: REFUND
pan	String(100)	Customer account value or number used in the transaction. If payment solution is Credit Cards, it is the specinCreditCardToken used when original payment was created.
brandId	Integer(18)	Brand Id of the transaction refunded.
paymentSolutionId	Integer(18)	Payment solution id.
status	String(enum)	Transaction status: ● SET_FOR_REFUND if refund was created successfully. Will be automatically executed at a further stage (fulfillment). ● ERROR if any error happened so refund did not take place.
errors	String(400)	Only applies to ERROR transactions. A brief description of error causes.

3.3.2 Capture

Based on the amount authorised in an AUTH type transaction, the capture operation is performed in the full authorised amount. Partial captures are not supported at the moment.

3.3.2.1 Requesting the Session Token for Capture

Input

Parameter	Data Type	Mandatory Y/N/C	Description
merchantId	Integer(18)	Y	Merchant id that identifies the merchant in IPG.
originalMerchantTxId	String(50)	Y	Merchant transaction id of the transaction to be captured.
password	String(64)	Y	Merchant password in IPG.
allowOriginUrl	String(253)	Y	Merchant's web page URL, the one that would make the 2nd request using the token (CORS headers will allow only this origin)
action	String(enum)	Y	Action, must be CAPTURE.
timestamp	Integer(18)	Y	Milliseconds since 1970-01-01 00:00:00
originalTxId	Integer(18)	N	Transaction id in IPG.
agentId	String(18)	N	Id of the merchant's representative that requests the capture.
amount	BigDecimal (10.2 or 10.3) BigDecimal (15.2 or 15.3)	M	Amount to capture. At first stage and in the meantime partial captures are not allowed, any amount lower or different than initial transaction amount will result in refusal of the capture.

Output

Parameter	Data Type	Description
token	String(40)	One time use Session token.
merchantId	Integer(18)	Merchant identifier in IPG.

3.3.2.2 Execute Capture

Input

Parameter	Data Type	Mandatory Y/N/C	Description
merchantId	Integer(18)	Y	Merchant id that identifies the merchant in IPG.
token	String(40)	Y	Session token received in the first step.

Output

Response is a JSON with the following format:

Parameter/Label	Data Type	Description
merchantId	Integer(18)	Merchant identifier in IPG.
originalMerchantTxId	String(50)	Transaction identifier (or Order ID) in merchant system.
originalTxId	Integer(18)	Transaction id in IPG.
amount	BigDecimal (15.2 or 15.3)	Amount captured.
currency	String(enum)	Currency alphabetic code as defined in ISO 4217 standard. http://www.iso.org/iso/home/standards/currency_codes.htm
customerId	String(20)	Customer identifier in merchant system. Generated by IPG if merchant didn't provide it when original payment was created.

action	String(enum)	Action performed: CAPTURE
pan	String(100)	Customer account value or number used in the transaction. If payment solution is Credit Cards, it is the specin_creditCardToken used when original payment was created.
brandId	Integer(18)	Brand Id of transaction captured.
paymentSolutionId	Integer(18)	Payment solution id.
status	String(enum)	Transaction status: ● SET_FOR_CAPTURE if capture action was executed successfully. The transaction now is on a queue to be captured by a batch. ● ERROR if any error happened so capture did not take place.
errors	String(400)	Only applies to ERROR transactions. A brief description of error causes.

3.3.3 Void

3.3.3.1 Requesting the Session Token for Void

Input

Parameter	Data Type	MandatorY/N/C	Description
merchantId	Integer(18)	Y	Merchant id that identifies the merchant in IPG.
originalMerchantTxId	String(50)	Y	Merchant transaction id of the transaction to be voided.
password	String(64)	Y	Merchant password in IPG.
action	String(enum)	Y	Action requested, must be VOID.

allowOriginUrl	String(253)	Y	Merchant's web page URL, the one that would make the 2nd request using the token (CORS headers will allow only this origin)
timestamp	Integer(18)	Y	Milliseconds since 1970-01-01 00:00:00
originalTxId	Integer(18)	N	Transaction id in IPG.
agentId	String(18)	N	Id of the merchant's representative that requests the void.

Output

Parameter	Data Type	Description
token	String(40)	One time use Session token.
merchantId	Integer(18)	Merchant identifier in IPG.

3.3.3.2 Execute Void

Input

Parameter	Data Type	Mandatory Y/N/C	Description
merchantId	Integer(18)	Y	Merchant id that identifies the merchant in IPG.
token	String(40)	Y	Session token received in the first step.

Output

Response is a JSON with the following format:

Parameter/Label	Data Type	Description
merchantId	Integer(18)	Merchant identifier in IPG.
originalMerchantTxId	String(50)	Transaction identifier (or Order ID) in merchant system.
originalTxId	Integer(18)	Transaction id in IPG.
amount	BigDecimal (15.2 or 15.3)	Amount voided.
currency	String(enum)	Currency alphabetic code as defined in ISO 4217 standard. http://www.iso.org/iso/home/standards/currency_codes.htm
customerId	String(20)	Customer identifier in merchant system. Generated by IPG if merchant didn't provide it when original payment was created.
action	String(enum)	Action performed: VOID.
pan	String(100)	Customer account value or number used in the transaction. If payment solution is Credit Cards, it is the specinCreditCardToken used when original payment was created.
brandId	Integer(18)	Brand id of the transaction voided.

paymentSolutionId	Integer(18)	Payment solution id.
status	String(enum)	Transaction status: ● VOID if void was executed successfully. ● ERROR if any error happened so void did not take place.
errors	String(400)	Only applies to ERROR transactions. A brief description of error causes.

3.4 Transaction Status Check

There is another way to check the status of the transaction independent of the payment flow - status check. Just like the other methods, it requires a session token and a subsequent call to actually query the payment platform about the current status of a transaction. The transaction can be queried by its ***merchantTxId*** or the ***txId.***

3.4.1 Requesting the Session Token for Status Check

Parameter	**Data Type**	**Mandatory Y/N/C**	**Description**
merchantId	Integer(18)	Y	Merchant id that identifies the merchant in IPG.
password	String(64)	Y	Merchant password in IPG.
allowOriginUrl	String(253)	Y	Merchant's web page URL, the one that would make the 2nd request using the token (CORS headers will allow only this origin)
action	String(enum)	Y	Action, must be GET_STATUS.
timestamp	Integer(18)	Y	Milliseconds since 1970-01-01 00:00:00

3.4.2 Status Check Request

Parameter	Data Type	Mandatory Y/N/C	Description
merchantId	Integer(18)	Y	Merchant id that identifies the merchant in IPG.
token	String(40)	Y	Session token received in the first step.
action	String(enum)	Y	Action, must be GET_STATUS.
txId	Integer(18)	C	transaction ID in IPG system
merchantTxId	Integer(50)	C	transaction ID in merchant's system

3.4.3 Status Check Output

Parameter/Label	Data Type	Description
result	String(enum)	Static value: "success"
merchantId	Integer(18)	Merchant identifier in IPG.
merchantTxId	String(50)	Transaction identifier (or Order ID) in merchant system.
txId	Integer(18)	Transaction identifier in IPG system
status	String(enum)	Current status of the transaction in IPG platform (For possible statuses refer to Page 37 - "status" parameter)

3.5 Get Available Payment Solutions

This allows the merchant to check which payment solutions are available for a given country, currency and brand. Like methods described before, it requires a session token and a subsequent call to actually query the payment platform about the payment solutions.

3.5.1 Requesting the Session Token for Get Available Payment Solutions

Parameter	Data Type	Mandatory Y/N/C	Description
merchantId	Integer(18)	Y	Merchant id that identifies the merchant in IPG.
password	String(64)	Y	Merchant password in IPG.
allowOriginUrl	String(253)	Y	Merchant's web page URL, the one that would make the 2nd request using the token (CORS headers will allow only this origin)
action	String(enum)	Y	Action, must be GET_AVAILABLE_PAYSOLS.
timestamp	Integer(18)	Y	Milliseconds since 1970-01-01 00:00:00
currency	String(enum)	Y	Currency alphabetic code as defined in ISO 4217 standard. http://www.iso.org/iso/home/standards/currency_codes.htm
country	String(enum)	Y	Country alpha-2 code as defined in ISO 3166 standard. http://www.iso.org/iso/country_codes
brandId	Integer(18)	N	Contact IPG to get your brand ID associated. If not provided default value will be used.

3.5.2 Get Available Payment Solutions Request

Parameter	Data Type	Mandatory Y/N/C	Description

merchantId	Integer(18)	Y	Merchant id that identifies the merchant in IPG.
token	String(40)	Y	Session token received in the first step.

3.5.3 Get Available Payment Solutions Response

Response is a JSON with the following format:

Parameter/Label	Data Type	Description
result	String(enum)	Static value: “success”
resultId	String	example, "38e2d66e-6c7b-4152-bf87-fdbecb7cfa2f"
data	Array	List of payment solutions available.
data:ID	Integer(18)	Payment solution identifier in IPG.
data:NAME	String(150)	Payment solution name.

Response example:

{result=success, data=[{NAME=CreditDebitCards, ID=500}, {NAME=Neteller, ID=100}], additionalDetails={}, resultId=e79e9506-a38a-403e-b435-be0c91b436db}

4 PCI Compliant Payment Form Integration

4.1 Loading the Payment Form

Once merchant has requested and received the session token for AUTH/PURCHASE transaction, the payment form may be called to its URL to open/load it inside an iFrame.

To load the payment form, the merchant ID and the session token are required:

Parameter	Data Type	Mandatory Y/N/C	Description
merchantId	Integer(18)	Y	Merchant id that identifies the merchant in IPG.
token	String(40)	Y	Session token received in the first step.
containerId	String	C	ID of the html element to wrap the cashier UI. Required for JavaScript based integration.
successCallback	String	C	Name of the JS function that will be called in case of successful transaction. Sample function: *function handleSuccess(data){ /* code */ }*
failureCallback	String	C	Name of the JS function that will be called in case of failed/declined transaction. Sample function: *function handleFailure(data){ /* code */ }*
cancelCallback	String	C	Name of the JS function that will be called in case of user cancelled transaction. Sample function: *function handleCancel(data){ /* code */ }*
bannerUrl	String	N	A valid URL of a resource (html page, image) to replace the default footer with logo
integrationMode	String(enum)	C	Possible values: - inject - iframe - standalone

4.1.1 Sample Payment Form Invocation:

UAT JavaScript URL: https://cashierui-apiuat.test.intelligent-payments.com/js/api.js
UAT baseUrl: https://cashierui-apiuat.test.intelligent-payments.com/ui/cashier

LIVE JavaScript URL: https://cashierui-api.intelligent-payments.com/js/api.js
LIVE baseUrl: https://cashierui-api.intelligent-payments.com/ui/cashier

Template for Cashier CSS customisation:
https://cashierui-apiuat.test.intelligent-payments.com/cashier/css/cashier-customisation-template.css
(Note that the customised file is subject to IPG review and sign off for security perspective. It needs to be put on IPG server in order to work. For customisation guidelines please refer to Appendix A - Payment Page Customisation Policy)

Sample code for redirection based integration:

```
<!DOCTYPE html>
<html><head>
<script type="text/javascript">
window.onerror = function myErrorHandler(errorMsg, url, lineNumber) {
    alert("Error occured: " + errorMsg + "\nurl: "+ url + "\nline: " + lineNumber);    return
false;
}
</script>
<style>
label {
    width: 10em;
    display: inline-block;
    margin: 0 0 0.5em 0;
}
input {
    width: 15em;
}
</style>
</head>
<body>
    <h3>Redirection example</h3>
    <h4>redirectedFullWindow</h4>
<form method="get" action="https://cashier-turnkeyuat.test.myriadpayments.com/ui/cashier">
    <label>token:</label><input name="token"/><br/>
    <label>merchantId:</label><input name="merchantId" value="666"/><br/>
    <label>paymentSolutionId:</label><input name="paymentSolutionId" value="500"/><br/>
    <input type="hidden" name="integrationMode" value="standalone"/>

    <button type="submit" >Pay </button>
</form>
</body>
</html>
```

Sample code for JavaScript based integration:

```
<!DOCTYPE html>
<html><head>
<style>
#ipgCashierDiv{
    width: 600px;
    height: 400px;
    border: 1px solid gray;
    margin: 10px;
    }
label {
    width: 10em;
    display: inline-block;
    margin: 0 0 0.5em 0;
}
input {
    width: 15em;
}
</style>
<script type="text/javascript"
src="https://cashierui-apiuat.test.intelligent-payments.com/js/api.js"></script>
<script type="text/javascript">
var cashier = com.myriadpayments.api.cashier();
cashier.init(
    { baseUrl:'https://cashierui-apiuat.test.intelligent-payments.com/ui/cashier' }
);
function handleResult(data){
    alert(JSON.stringify(data));
}
function pay(){
    var token = document.getElementById("tokenIn").value;
    var merchantId = document.getElementById("merchantIdIn").value;
    cashier.show(
        {
            containerId:"ipgCashierDiv",
            merchantId: merchantId,
            token:token,
            successCallback: handleResult,
            failureCallback: handleResult,
            cancelCallback: handleResult,
            styleSheetUrl: "/cashier/css/optional-customisation.css"
        }
    );
};
</script>
</head>
<body>
    <h3>Simple Javascript integration example</h3>
    <h4>simpleJsIntegration</h4>
<div>
    <label>token:</label><input id="tokenIn"/><br/>
    <label>merchantId:</label><input id="merchantIdIn" value="666"/><br/>
    <button onclick="pay()">Pay</button>
</div>
<div id="ipgCashierDiv"></div>
```

```
</body>
</html>
```

5 Virtual Terminal Integration

5.1 Requesting the Session Token for Virtual Terminal

Input

Parameter	Data Type	Mandatory Y/N/C	Description
merchantId	Integer(18)	Y	Merchant id that identifies the merchant in IPG.
merchantTxId	String(50)	C	Transaction identifier (or Order ID) in merchant system. If not provided, IPG will generate it. Optional if action is AUTH, PURCHASE, REFUND. Not required in other cases.
originalMerchantTxId	String(50)	C	Merchant transaction id of the transaction is going to be updated. Mandatory if action is VOID, REFUND, CAPTURE. Not required in other cases.
originalTxId	Integer(18)	C	Transaction id of the initial transaction in IPG. Optional if action is VOID, REFUND, CAPTURE. Not required in other cases.
password	String(64)	Y	Merchant password in IPG.
action	String(enum)	Y	Action required. Must be PURCHASE if Sale, AUTH if Authorisation: • AUTH • PURCHASE • VOID • REFUND • CAPTURE
agentId	String(18)	N	Id of the merchant's representative that requests the refund.
customerId	String(20)	N	Customer identifier in merchant system. If not provided, IPG will generate a one time use customer id.
operatorId	String(20)	N	If the operation is performed by the merchant's operator or agent on behalf of the end customer and if merchant wants to track which operator performed which transaction this field should be filled in.

brandId	Integer(18)	N	Contact IPG to get your brand ID associated. If not provided default value will be used.
channel	String (enum)	Y	MOTO only for VT
amount	BigDecimal(10.2 or 10.3) BigDecimal(15.2 or 15.3)	C	Transaction amount. Includes tax, shipping, charge and discount amounts. It's optional if action is AUTH, PURCHASE, REFUND. If amount is sent as input param at this stage, merchant representative will be unable to update its value at a further stage. Not required for VOID, CAPTURE.
taxAmount	BigDecimal(10.2 or 10.3) BigDecimal(15.2 or 15.3)	N	Tax amount.
shippingAmount	BigDecimal(10.2 or 10.3) BigDecimal(15.2 or 15.3)	N	Shipping amount.
chargeAmount	BigDecimal(10.2 or 10.3) BigDecimal(15.2 or 15.3)	N	Charge amount.
discountAmount	BigDecimal(10.2 or 10.3) BigDecimal(15.2 or 15.3)	N	Discounts applied.
currency	String(enum)	C	Currency alphabetic code as defined in ISO 4217 standard. http://www.iso.org/iso/home/standards/currency_codes.htm Mandatory if action is AUTH, PURCHASE. Not needed in other cases.
country	String(enum)	Y	Transaction country. This will be the country where the transaction takes place. The value is the alpha-2 code as defined in ISO 3166 standard. http://www.iso.org/iso/country_codes Mandatory if action is AUTH, PURCHASE. Not needed in other cases.
language	String(enum)	N	Language. Merchant representative prefered language. The value must be a 2 letter code as defined in ISO 639-1 standard. http://www.iso.org/iso/catalogue_detail?csnumber=22109 This value will be sent to some payment acquirers so they can show their site to the customer in the language requested. If not provided, language will be the payment acquirer's default.
freeText	String(200)	N	Merchant free text or comments.
merchantNotificationUrl	String(200)	C	URL where the merchant will receive transaction status messages.

			Not required if action is VOID, CAPTURE, REFUND as merchantNotificationUrl received when original payment was created will be used. If it's not defined, status messages will be skipped.
firstTimeTransaction	boolean (true, false)	N	Customer's first time transaction flag. Default: ● if customerId provided, false ● if customerId not provided, true
forceSecurePayment	boolean (true, false)	N	Only applies for Credit Card transactions and action AUTH, PURCHASE. Forces 3DSecure process no matter the routing rules. Default value is false and still 3DSecure can be executed if transaction matches 3DSecure rules set in BackOffice.
process3DSecureU	boolean (true, false)	N	Only applies for Credit Card transactions and action AUTH, PURCHASE in which 3D secure is involved (because forceSecurePayment param is true or matches 3DSecure routing rules). If true, 3DSecure authentication response U (Unknown) is considered success and auth/sale is requested, if false, transaction fails at that stage. If not provided, value from 3DSecure routing rules is used.

Output

Parameter	Data Type	Description
result	String(enum)	success or failure
merchantId	Integer(18)	Merchant identifier in IPG.
token	String(40)	One time use Session token.

5.2 Loading the Virtual Terminal

Once the merchant has received the session token, a call is performed to the Virtual Terminal URL to open/load it inside an iFrame.

To load the Virtual Terminal, the merchant ID and the session token are required:

Parameter	Data Type	Mandatory Y/N/C	Description
merchantId	Integer(18)	Y	Merchant id that identifies the merchant in IPG.
token	String(40)	Y	Session token received in the first step.

6 Transaction Result Calls

When an operation is completed (successfully or not), a notification is sent to inform the merchant about the result and the current status of the payment. This notification is sent to the url provided as ***merchantNotificationUrl*** at the session token request by the merchant.

For merchants using the Integrated Payment Forms this is provided when requesting the session token to open/load the forms. Other merchants might provide it while calling AUTH/SALE operations. Please note that further operations affecting the same payment (VOID, REFUND, CAPTURE) will use the same ***merchantNotificationUrl*** as the related AUTH/SALE.

If there is no ***merchantNotificationUrl*** this notification is skipped and merchants have to reconcile their payment data by running the detailed transaction reports from the Backoffice.

The notification or result call is sent as a POST request message with the following parameters:

Parameter	Data Type	Description
merchantId	Integer(18)	Merchant id that identifies the merchant in IPG.
merchantTxId	String(50)	Transaction identifier (or Order ID) in merchant system. Generated by IPG if merchant didn't provide when payment was requested.
txId	Integer(18)	Transaction id in IPG.
acquirerTxId	String(100)	Transaction identifier in acquirer system, if acquirer returns it.
amount	BigDecimal (15.2 or 15.3)	Transaction amount. Includes tax, shipping, charge and discount amounts.
currency	String(enum)	Currency alphabetic code as defined in ISO 4217 standard. http://www.iso.org/iso/home/standards/currency_codes.htm
customerId	String(20)	Customer identifier in merchant system. Generated by IPG if merchant didn't provide it in first step.
action	String(enum)	Action executed: • AUTH • PURCHASE • REFUND • CAPTURE • VOID

pan	String(100)	Customer account value or number used in the transaction. If payment solution is Credit Cards, it is the credit card token, not real card number.
brandId	Integer(18)	Brand Id as received in session token request or default value if not provided there.
paymentSolutionId	Integer(18)	Payment solution id.
status	String(enum)	Transaction status: • SET_FOR_CAPTURE • NOT_SET_FOR_CAPTURE • DECLINED • CAPTURED • SET_FOR_REFUND • COMPLETED_REFUND • VOID • ERROR
acquirer	String(100)	The acquirer name in case of a Credit Card payment or the payment solution name if an alternative payment method has been used.
acquirerAmount	BigDecimal (15.2 or 15.3)	Amount processed by payment acquirer. May be different toinitial amount requested.
acquirerCurrency	String(enum)	Transaction currency in payment acquirer side, may be different totransaction currency requested (e. g. if a currency conversion applied). Currency alphabetic code as defined in ISO 4217 standard. http://www.iso.org/iso/home/standards/currency_codes.htm
country	String(enum)	Transaction country. This will be the country where the transaction takes place. The value is the alpha-2 code as defined in ISO 3166 standard. http://www.iso.org/iso/country_codes
freeText	String(200)	Merchant free text.
language	String(enum)	Language. Customer's prefered language. The value must be a 2 letter code as defined in ISO 639-1 standard. http://www.iso.org/iso/catalogue_detail?csnumber=22109
errorMessage	String(400)	Only applies to ERROR transactions. It's a brief description of error cause.
paymentSolutionDetails		JSON block. Specific payment solution content, it may vary depending on the payment solution. See paymentSolutionDetails section for further info.

Additional Details:

Parameter/Label	Data Type	Payment solution	Description
authCode	String	Credit Cards	Transaction authorisation code as received from acquirer. Length of this field is acquirer specific.

8 Testing

In order to test the payment platform, there are two options depending on the integration type that you require for your payments.:

1. Integrated payment pages testing
2. Direct API integration testing

Below is the data that must be used in the test environment:

8.1 URLs:

Session Token Request url (1st step of payment process):
https://apiuat.test.intelligent-payments.com/token

Payment Action url (2nd step of payment process):

https://apiuat.test.intelligent-payments.com/payments

8.2 Test Merchant Details:

Merchant ID	160001
Brand ID	1600010000
Test Merchant Password	*to_be_provided*

8.3 Test Card Details:

Card Number	Card Type	Expiry	CVV	Amount	Comments
5454545454545454	Visa Credit	2018/12	111	20.00	Error responses might be received if other amounts/card/transaction details are used
4111111111111111	Visa Credit	2018/12	111	20.00	Error responses might be received if other amounts/card/transaction details are used

8.4 Integrated Payment Pages Testing:

To start the payment you need to obtain a session token by submitting all the required parameters specified in section 3.2 to the URL specified in Section 8.1. Then you may use the obtained session token to load the payment page (cashier) to complete the test payment. Alternatively you can use the test harness to simulate a cashier integration as an example. Integrated Payment Pages are only available for the purpose of E-Commerce type transactions. For CNP (Card Not Present) type transactions such as MO/TO, one can use either the Direct API or the Virtual Terminal application provided by IPG. For more information, you may contact your account manager/representative.

Reference/demo cashier integration test harness:

https://cashierui-apiuat.test.intelligent-payments.com/ui/checkout-demo/

JavaScript for cashier integration:

https://cashierui-apiuat.test.intelligent-payments.com/js/api.js

Sample cashier invocation:

```
<script type="text/javascript" >

        var cashier =
com.myriadpayments.api.cashier({baseUrl:'https://cashierui-apiuat.test.intelligent-payments.com
'});

        params = {

            containerId:'divForCashier',
            merchantId:'160001',
            token:'sessionTokenFromRequestToSessionTokenURLSection8.1'
        };
        cashier.show(params);

</script>
```

8.5 Direct API Testing

To start the payment you need to obtain a session token by submitting all the required parameters specified in section 3.2 to the Session Token URL specified in Section 8.1. In the case of a card transaction, you first need to tokenise the test card provided in section 8.3 by using the tokenise operation specified in section 3.1. When you have the session token and the card token (if required) available, then you can complete the payment by using the Payment Action URL specified in section 8.1.

Sample 1: Credit Card Tokenisation Test

Session Token Request
POST https://apiuat.test.intelligent-payments.com/token
POST data:
merchantId=160001&action=TOKENIZE&password=*to_be_provided*&allowOriginUrl=www.myurl.com&timestamp=1466691987144

Session Token Response
{"result":"success","token":"0728339b-7149-4e0d-9ffb-6bdadf4c86ec","merchantId":"160001","additionalDetails":{},"resultId":"f9659f32-a335-4af9-a60d-967f3a0b2506"}

Card Token Request
POST https://apiuat.test.intelligent-payments.com/payments
POST data:
token=2609cb3e-6a70-4fc2-860a-33822aa3bfbc&merchantId=160001&number=4111111111111111&nameOnCard=Jean+Carl+Grech&expiryMonth=12&expiryYear=2018

Card Token Response
{"result":"success","cardToken":"6727777487171111","customerId":"HXSmIMf7qUYp76Gllwe5","cardIssuer":null,"merchantId":"160001","additionalDetails":{},"resultId":"d9904160-c567-458e-9336-2939abfbab34","cardType":"400","country":null}

Sample 2: Authorisation Card Payment Test

Session Token Request
POST https://apiuat.test.intelligent-payments.com/token
POST data:
merchantId=160001&action=AUTH&password=*to_be_provided*&allowOriginUrl=www.myurl.com&timestamp=1466691988431&channel=ECOM&userDevice=DESKTOP&amount=20.00&currency=EUR&country=GB&paymentSolutionId=500&specinCreditCardToken=6727777487171111&customerId=GZPvv9JQ3cxEAxNqdqIX

Session Token Response

{"result":"success","token":"8ac61a97-f884-4ccc-a1e7-d11a3ce4051c","merchantId":"160001","additionalDetails":{},"resultId":"e9ca2017-6549-4ccc-b20d-fd5fe924f190"}

Authorisation Payment Request

POST https://apiuat.test.intelligent-payments.com/payments
POST data:
token=8ac61a97-f884-4ccc-a1e7-d11a3ce4051c&merchantId=160001&specinCreditCardCVV=111

Authorisation Payment Response

{"result":"success","txId":"11259136","paymentSolutionId":"500","acquirerTxId":"200BE2FDF2A94348BD3A917994DD7A0E","freeText":null,"resultId":"633d5ca6-b69b-4a30-89cd-a27176d4e79f","brandId":"1","currency":"EUR","pan":"3674506345311111","amount":"20.00","acquirerCurrency":null,"customerId":"GZPvv9JQ3cxEAxNqdqIX","errors":null,"paymentSolutionDetails":null,"acquirerAmount":null,"action":"AUTH","merchantTxId":"kmQQWpG5NkAAvQYXvWXW","merchantId":"160001","additionalDetails":{},"language":null}

Note:

As one can notice above, for every action, a different session token needs to be generated and obtained, for security purposes. A session token can only be used once and has a maximum lifetime of a few minutes.

9 Frequently Asked Questions

Question	Answer
What is a credit card token?	A credit card token (parameter name ***specinCreditCardToken***) is a numeric value which represents a real credit card number. This is used across the payments platform to avoid using the real card number across the applications. The credit card token usually consists of a set of randomly generated numbers followed by the last (real) four digits of the credit card. A credit card token is unique for each card. The token also requires to be identified by the customerId for a transaction to take place.
What is a session token? How do I use it to perform a transaction?	A session token is a set of characters generated by the payment platform which is used to identify an operation. A different session token is required for each and every operation/action that is performed on the payments platform. Therefore, for example, to perform a transaction with a new credit card, the following steps are required: To tokenize the new credit card: ● Session token is requested for the TOKENIZE action using the required parameters ● A session token is generated for the above request ● A request is then performed using the same session token and the required card details to tokenize the new card. ● This will return the card details (including the card token and the customerId + other info) To perform an AUTH transaction: ● Session token is requested for the AUTH action using the required parameters (including the payment details together with the card token and its respective customerId returned in the previous action) ● The transaction is then performed using the same session token together with the merchantId and the credit card CVV (specinCreditCardCVV) ● The transaction response is received.
Why do I keep getting "internal error" when I try	Please ensure that for the test environment the currency EUR is used. If you try to use different currencies, the transactions will fail. Of course this won't be the case in production.

to process a transaction?	
How do I test 3D Secure?	On the test environment, the easiest way to force a 3Dsecure payment is: add the parameters brandId = 2 and forceSecurePayment = true to the session token request. Also merchantNotificationUrl and merchantLandingPageUrl need to be specified.

Appendix A - Payment Page Customisation Policy

To keep our payment security standards and PCI compliance, the following regulations apply for payment page customisations:

- Customisation must be limited to:
  - Font used in texts
  - Background colours
  - Text colours
  - Element colours (boxes, shapes etc…)
  - Element styling (e.g. box style, round/sharp edges)
- The following items are **not** permitted:
  - Elements outside of the existing elements in the payment page;
  - External links
  - Hidden elements
  - Element layout changes (e.g. box positioning in the iframe);

END OF DOCUMENT