E-Commerce Payment Gateway (EPG)
Merchant Integration Guide

Contents

1.Preface

The SmartVista E-Commerce Payment Gateway: Specification – API: WEB Interface describes how to connect to the SmartVista E-Commerce Payment Gateway (EPG) using the Web API interface.

Audience

This guide is intended for users of merchants, and acquirers that work with the SmartVista E-Commerce Payment Gateway (EPG) system remotely through API calls.

2. Introduction

Internet acquiring is used to sell goods and services via the public Internet, allowing customers to use their regular bank cards to make purchases. A merchant planning to sell goods or services via the Internet must ensure the security of the payments being processed by taking the following measures:

- Key data (such as personal data and bank cards details) must be transferred via a secure (SSL or TLS) connection.
- Information about the current payment (amount, currency, or description of the order) and the result of the payment must be protected from and not be accessed by intruders.
- Before the payment is made, the customer's card parameters (for example, the expiration date, cardholder name and so on) should be checked.

The 3-D Secure protocol is used to provide an additional security layer. The major payments networks have introduced their own services based on the 3-D Secure protocol, for example, Mastercard has "Mastercard Identity Check" and Visa has "Visa Secure".

The SmartVista E-Commerce Payment Gateway is used as a technology platform. It enables a merchant to perform the necessary safety procedures without the need to

significantly restructure their Internet store site and existing business processes. EPG supports the following operations:

- P2P transactions
- Recurring payments

2.1 Data types used

The characters described in the table below are used to designate the data types used in API methods.

Character	Description
A	Alphabetic characters.
N	Numeric characters.
S	Special characters.
AN	Alphanumeric characters.
AS	Alphabetic and special characters.
NS	Numeric and special characters.
ANS	Alphanumeric and special characters.
4	Fixed length of N characters (four in the example).

...16	Variable length up to a maximum of N characters (16 in the example).

2.2 Mandatory fields

The designations described in the table below are used to indicate whether fields transferred in API methods are mandatory.

Designation	Description
Yes	Field is required.
No	Field is optional.
Conditional	Field may be required or optional depending on a condition.

3. Request structure

The examples of requests provided in this document include the query and JSON body. The JSON body parameters are in braces ({), for example:

```
https://<host:port>/epg/rest/p2p/register.do?userName=test_api&password=1
 {
    "amount": 1,
    "currency": "840",
    "returnUrl": "http://127.0.0.1/cart.html",
```

```
        "orderNumber": "88878890887535756777"
}
```

where the following is the query:

```
https://<host:port>/epg/rest/p2p/register.do?userName=test_api&password=1
```

and the following is the JSON body:

```
{
    "amount": 1,
    "currency": "840",
    "returnUrl": "http://127.0.0.1/cart.html",
    "orderNumber": "88878890887535756777"
}
```

4. Web API

Interactions between the merchant and the SmartVista E-Commerce Payment Gateway are implemented as HTTP requests with **GET** or **POST** methods to specific URLs. These are separate for each individual request type.

IMPORTANT: It is highly recommended that you use the **POST** method and send parameters of a request in the request body with the content type *application/x-www-form-URL-encoded* if the other content type is not explicitly required by a specific API according to its description. If the **GET** method is used, sensitive request parameters may be undesirably stored in logs due to the HTTP protocol peculiarities and the logging settings of the network infrastructure

components. However, all request examples are provided with the **GET** structure for the sake of clearer visual illustration.

The parameters used in the interactions are sent as parameters of **GET** or **POST** requests; their values must be URL-encoded.

The result of processing a request returns as a JSON object, for example:

```
{"errorCode":"12","errorMessage":"Empty amount"}
```

For authorization purposes, each request must include the login and password of the user who manages merchants and were generated during the user's registration. The values are sent as the following parameters:

Name	**Type**	**Mandatory**
userName	AN 1..100	Yes
Login of the user generated during registration.		
password	String	Yes
Password of the user generated during registration.		

EPG - Merchant Integration Guide

Note: The **browserAcceptHeader** in all requests to web API should contain the following:
application/json, text/javascript, */*;

Either one-phase or two-phase payment schemes may be used:

- **One-phase payment** — a payment for goods or services executed through the Internet with the use of a bank card. It is executed as one action that does not require additional confirmation.
- **Two-phase payment** — a payment for goods or services executed through the Internet with the use of a bank card and requiring additional confirmation. The two-phase mechanism splits the process into checking whether the card is capable of paying (authorization) and debiting the money from the account (financial confirmation). During the first step of a two-phase payment, the bank card paying capacity is checked and money on the customer's account are put on hold. The second step is a confirmation of the funds transfer.

Requests for each of these schemes, which may differ, are described below. One-phase and two-phase payment web API requests include the following:

- Creating bindings without payments
- Creating bindings without payments anonymously
- Processing payments with bindings
- Modifying the card expiration date in a binding using 3DS
- Modifying the card expiration date in a binding
- Processing the updated card expiration date in a binding
- Deactivating a binding
- Reactivating a binding
- Checking activation of a binding
- Checking the order amount
- Getting the list of bindings
- Getting the list of bindings by PAN or bindingId
- Request for a zero amount binding

· **P2P requests:**

- Request for P2P order registration
- Request for P2P funds transfer
- Request for P2P payment status
- Request for P2P transfer verification
- Request for P2P transfer verification without registration
- Request for information about P2P transaction
- Request for money transfer status

All text fields must use UTF-8 encoding. In web API requests, special characters must be encoded using URL encoding. The table of characters can be viewed at the following URL: https://www.w3schools.com/tags/ref_urlencode.asp. For example, a password qwe?rt%y is transferred as qwe%0Frt%25y.

Note: The error code does not display the order status. To view the status of an order, use the getOrderStatus or getOrderStatusExtended requests.

4.1 Order registration request

Depending on whether one-phase or two-phase payments are used, one of the following methods is used to register an order:

- **register.do** — for one-phase payments
- **registerPreAuth.do** — for two-phase payments

These requests are designed to register orders in the SmartVista E-Commerce Payment Gateway and have the same set of parameters.

Request parameters

Name	Type	Mandatory
userName	AN 1..100	Yes
Login of the API user on whose behalf requests are processed for a specific merchant.		
password	String	Yes
User's password.		
orderNumber	AN 1..32	No
Number (identifier) of the order in the merchant's online store system. It is unique for every store in the system and is generated on the order registration. If the **Require system to generate order numbers** permission is enabled for the merchant, the order number is generated automatically. Otherwise, the order number data is sent in the API.		

amount	N 1..12	Yes
Order amount in the minor denomination (for example, cents).		
currency	N3	No
Payment currency code in the ISO 4217 format. If the **currency** parameter in the request is not specified, the default currency code is used.		
returnUrl	ANS 1..2000	No
URL to which the customer is redirected after a successful payment.		
description	ANS 1..600	No
Free form description of the order.		
language	A2	No
Language code in the ISO 639-1 format. If it is unspecified, SmartVista E-Commerce Payment Gateway uses the default language from the merchant settings. Error messages are also returned in this language.		

clientId	ANS 1..255	No
Customer identifier in the merchant system. This parameter is mandatory for bindings.		
sendPaymentLink	A3..10	No
Specifies the channel used to send the payment link: · SMS · EMAIL Either one or both these values can be specified using a comma and a space as delimiters, for example: **sendPaymentLink = EMAIL** or **sendPaymentLink = SMS** or **sendPaymentLink = EMAIL,SMS** or **sendPaymentLink = SMS, EMAIL**		
email	ANS..*	Conditional

Customer's email address to which the payment link is sent. It is required if **sendPaymentLink = EMAIL**.		
phone	AN..255	Conditional
Customer's phone number to which the payment link is sent. It is required if **sendPaymentLink = SMS**.		
name	AN 1..30	No
Customer's full name.		
jsonParams[]	AN..1024	No
Fields used to store additional information. The type is as follows: {"param":"value","param2":"value2"} Note: The parameter name length is 255 characters or less and the value length is 1024 characters or less. See JSON parameter list for information about which parameters are passed.		
sessionTimeoutSecs	N..9	No

Lifespan of the order, in seconds.

The order lifespan duration can be taken from the following parameters (from the highest priority to the lowest):

- **sessionTimeoutSecs** transferred in a request. It overrides all other order timeout settings.
- If the **sessionTimeoutSecs** parameter is not specified, the value from the merchant's settings is used. It is configured by the **Alternative session timeout** option that must be enabled and the additional **Session duration** parameter that must be specified.
- If none of the above mentioned settings is specified (neither **sessionTimeoutSecs** nor merchant's timeout), the default value set on the **Administration > System settings** page by the **default.session.timeout.milliseconds** system setting is used. The default value is 1200 seconds.

If the request contains the **expirationDate** parameter, the **sessionTimeoutSecs** parameter is ignored.

expirationDate	ANS	No

Date and time when the order is terminated, in the following format: *yyyy-MM-ddTHH:mm:ss*.

If this parameter is not specified, the **sessionTimeoutSecs** parameter is used to determine the date and time when the order is terminated.

bindingId	AN..255	No

Identifier of the binding that was created earlier (see Managing bindings). It can only be used if the merchant has the permission to work with bindings. If this parameter is sent in the **registerOrder** request, the following is executed: · This order can only be paid by binding. · The payer is redirected to a payment page where the CVC must be entered.		
features	String	No
Specifies whether 3-D Secure check is enabled for a merchant: · FORCE_SSL — transaction is processed as SSL. A merchant must have a permission to process SSL transactions. · FORCE_TDS — transaction is processed as 3DS2. A merchant must have a permission to process 3DS2 transactions.		
paymentWay	AN..32	No

Payment method used for the payment:

- CARD — payment made by entering the card details
- CARD_BINDING — payment made using a binding
- CARD_MOTO — payment made using the call center
- UPOP_MOTO (CUP UPOP MOTO) — payment made China UnionPay Express Pay
- UPOP — payment made China UnionPay Secure Pay
- FILE_BINDING — payment made using a binding uploaded in a file
- P2P — payment made when transferring funds from one card to another
- APPLE_PAY — payment made using the Apple Pay service
- MASTERPASS — payment made using a Masterpass e-wallet
- OTHER — payment used for orders processed outside EPG
- GOOGLE_PAY — payment made using the Google Pay service
- SAMSUNG_PAY — payment made using the Samsung Pay service
- NSPK_E_CERT — payment made using the NSPK Electronic Certificate
- FASTPAYMENT_QR — payment made using the FastPayment QR code
- MPU — payment via Myanmar Payment Union (MPU)
- WAVE — payment via WavePay

Note: If you want to use custom payment methods, they can be defined through **jsonParams[]**. See Customizing payment methods and JSON parameter list in “Reference information”.

recurrenceType	String	Yes for recurring payments

Way in which subsequent recurring payments are processed:

- AUTO — for automatic handling of payments scheduled on the EPG side.
- MANUAL — for manual processing of recurring payments scheduled on the merchant side.

For recurring payments, this parameter is used to schedule the next payment (if AUTO is selected).

For installment payments, the parameter is used to schedule the next payment (if AUTO is selected) and check if the merchant is sending the request at the correct time (if MANUAL is selected).

When the next payment date comes, the system sends an AUTO payment request according to the schedule, and the merchant sends a MANUAL payment request.

EPG checks whether the payment date is correct. If it is not, the request is rejected and the merchant is warned that that it is not time for the payment (either too early or too late). For the late payments, an updated installment payment date (**recurrenceEndDate** and **recurrenceFrequency**) must be provided before a new payment can be made. If **recurrenceType = MANUAL**, the merchant can initiate a **recurrentPayment.do** request any time regardless of the date specified by the **recurrenceFrequency** parameter. The system does not validate whether the payment date matches the scheduled installment plan.

The **recurrenceType** parameter is mandatory to register a recurring or installment payment.

The **recurrenceType** parameter is not used in the **registerPreAuth.do** method.

recurrenceFrequency	String	For MANUAL: No For AUTO: Yes

This parameter is mandatory for AUTO installments and recurring payments and is ignored for MANUAL installments and recurring payments. For information about configuring this parameter, see Specifying reccurenceFrequency. This parameter is not used in the **registerPreAuth.do** method.		
recurrenceStartDate	N8	Yes
Date when the recurring/installment payment starts, in the following format: *YYYYMMDD*. *YYYYMMDD* 00:00:00 time is used for the start date. This parameter is mandatory for installment payments and AUTO recurring payments, and ignored for MANUAL recurring payments. The minimum **recurrenceStartDate** parameter value is tomorrow for AUTO recurring payments and installments. If the **recurrenceStartDate** is not specified, it will be tomorrow. This parameter can be specified for both recurring and installment payments. This parameter is not used in the **registerPreAuth.do** method.		
recurrenceEndDate	N8	Yes for installment payments

Date when the recurrence/installment payment ends, in the following format: *YYYYMMDD*. *YYYYMMDD* 23:59:59 time is used for the end date. This parameter is used for installment payments and AUTO recurring payments, and ignored for MANUAL recurring payments. This parameter is not used in the **registerPreAuth.do** method. This parameter is ignored for MANUAL recurring payments. If the Terminate Subscription code was received from SVFE, the **recurrenceEndDate** is set as the current date and time for recurring and installment payments. Future scheduled payments are canceled.		
threeDS2Params[]		Conditional
Parameters of the 3-D Secure 2 protocol authentication. The **threeDS2Params** parameter is a JSON-based structure. For more information, see 3-D Secure 2 parameter list. Depending on the type of channel interface that is used to initiate the transaction (**deviceChannel**), the parameter is mandatory or optional: · Optional for browser-based authentication · Mandatory for application-based authentication		
externalFee		Conditional
Fee amount in the minor denomination of the transaction currency. This parameter is used only if the **EXTERNAL_FEE_ALLOWED** option in the merchant configuration is enabled. Otherwise, this value is ignored.		

The fee is not calculated in SVFE if the **externalFee** value is provided for the transaction.		
amountECert	N 1..19	Conditional
Transaction amount paid using the NSPK electronic certificate. This field displays the **totalCertAmount** value of the NSPK Electronic Certificate Front Office system. The amount is specified in the minor currency units. This field is present if the **NSPK E-Certificate** payment method is allowed for the acquirer and the merchant can use NSPK E-Certificates for payments.		
orderBundle	JSON array	Conditional
List of ordered items. This parameter is available in the request if the **Include Cart Items** parameter in the payment page settings is enabled. The following data is provided for each item added to the chart: name, position ID, quantity, value, measure unit, ordered amount, and item code.		
pageView	ANS..20	No

Format of the customer's payment interface:

- DESKTOP — pages displayed on the PC (pages with names of the payment_<*locale*>.html and errors_<*locale*>.html format). It is also used if the **pageView** parameter is not specified or its format is invalid. This is the default value.
- MOBILE — pages displayed on the mobile device (pages with names of the mobile_payment_<*locale*>.html and mobile_errors_<*locale*>.html format).

<*locale*> is the page language code according to ISO 639-1. For example, ru for Russian or en for English.

If the page file name contains a custom prefix, this prefix must be specified by the **pageView** parameter. For example, if **pageView = iphone**, the system searches for pages with the following names: iphone_payment_<*locale*>.html and iphone_error_<*locale*>.html.

installmentTotalAmount	N 1..12	Conditional
Total amount of the order. This parameter is mandatory if any of the other installment parameters is present.		
installmentSingleAmount	N 1..12	Conditional
Amount of the single installment payment (excluding the first payment; for the first installment payment, the **amount** value is used). This parameter is mandatory if any of the other installment parameters is present.		
installmentNumber	N 1..2	Conditional

Number of installments planned (including the first payment). This parameter is mandatory if any of the other installment parameters is present.		
airlineData[]		No
Airline data object. For the description of the parameters, see airlineData parameters. The airline data can be transferred if the **Airline data allowed** permission is enabled for the merchant.		
tipAmount	N 1..12	No
Amount of the tip. If this amount is present in the request, it is added to the main order amount during payment and transferred to SVFE separately in the **DE54** field. **Notes**: The **tipAmount** field is supported only for the **register.do** request. To perform payments with tips.		
tipRecipientCardNumber	N13..19	Conditional
Card number of the tip recipient. The amount specified in the **tipAmount** field is then transferred to the specified card using the P2P credit request. The tip recipient card number is transferred to SVFE in the **DE34** field. Notes: The **tipRecipientCardNumber** field is supported only for the **register.do** request. This field must be present in the request if **tipAmount** is present, otherwise an error is displayed.		
bindingManagedByMerchant		No

Entity that manages bindings: · true — bindings are stored and managed by merchants. · false — bindings are stored and managed by acquirer or EPG. This is the default value.		
bindingUsage		Conditional
Type of the Card-on-File (CoF) transaction: · saveCard — card is stored within a transaction. · paymentBySavedCard — card stored during the previous transactions is used in the next transaction. Note: This parameter can only be used if **bindingManagedByMerchant =** true.		
originalTransactionData		No
See the description of the Transaction Data block below.		
transactionInitiator		No
Originator of the transaction: · CIT — cardholder-initiated transaction · MIT — merchant-initiated transaction		
recurringPaymentManagedByMerchant		No

Entity that manages recurring payments:

- true — recurring payments are stored and managed by merchants.
- false — recurring payments are stored and managed by acquirer or EPG. This is the default value.

recurringPaymentForm		No

The type of the recurring payment:

- recurring
- installment

This parameter is mandatory if **recurringPaymentManagedByMerchant =** true.

recurringExpiry	N8	No

Recurring payment expiration date in the following format: *YYYYMMDD*.

Typically, this parameter is used for installment transactions. However, it can also be available in recurring transactions. If the value is empty, a default value of 99991231 is used for the Visa Areq A000000802-004 message extension.

originalTransactionData	JWS	No

See the description of the Transaction Data block.

This parameter is present in the request if **recurringPaymentManagedByMerchant =** true.

originalDsTransId	JWS	No
Directory Server Transaction ID, UUID (only for 3DS2 transactions). This parameter is present in the request if **recurringPaymentManagedByMerchant = true**.		
originalAcsTransId	JWS	No
Original transaction identifier in ACS (only for 3DS2 transactions). This parameter is present in the request if **recurringPaymentManagedByMerchant = true**.		
originalInitTransactionDate	N8	No
Initiating transaction date (only for 3DS2 transactions). This parameter is present in the request if **recurringPaymentManagedByMerchant = true**.		
originalInstallmentNumber	N1..99	No
Initiating installment payment number. This parameter is present in the request if **recurringPaymentManagedByMerchant = true** and **recurringPaymentForm = installment**.		

Note: Either all three parameters are required or none of them: **installmentTotalAmount**, **installmentSingleAmount**, and **installmentNumber**. The **recurrenceEndDate** and **recurrenceFequency** parameters are required for installments.

Transaction Data block

A single signed data block is used to transfer all original transaction data fields to and from a PCI DSS merchant, instead of multiple separate fields.

An RSA key pair with **CertificateType.TRANSACTION_DATA_SIGNATURE** must be generated for the merchant.

The block includes the following:

Header

Field	Description
alg	Algorithm (RS256).
kid	ID of the corresponding record in the key_mgmt_certificate table.
typ	Header type (JOSE).
merchantLogin	Login of the merchant.

Payload

Field	Description

mdOrder	This field is reserved for future use.
transDate	This field is reserved for future use.
networkReferenceData	Network reference data returned from the authorization host.
transactionDataElements	Data elements of the authorization request.
isSsl	Specifies whether 3DS2 authentication was not performed for the transaction.
dsTransId	Directory Server Transaction ID, UUID (only for 3DS2 transactions).
acsTransId	Transaction ID in ACS (only for 3DS2 transactions).
initTransactionDate	Initiating transaction date (only for 3DS2 transactions).
version	Constant value of 1. This field is reserved for future use.
panSalt	Base64-encoded 16 bytes random salt (the resulting string will be approximately 24 character long; from **SecureRandom**).

panHash	Base64 SHA-256 hash of byte[] = PAN.getBytes + salt.bytes.

The block validation fails if:

- The field is missing for MIT requests with **recurringPaymentManagedByMerchant=**true or **bindingManagedByMerchant=**true.
- A different **merchantLogin** is used.
- The key provided in the JWS header cannot be found for the current merchant.
- Signature validation fails (both technical and not errors).

Response parameters

Name	Type	Mandatory
orderId	ANS36	No
Unique order number generated by EPG after the registration of the order. It is absent if the order registration has failed (the error is described in the **ErrorCode** field).		
formUrl	AN..512	No
URL of the payment page to which the customer should be redirected. If the order registration has failed, this parameter is absent (the error is described in the **ErrorCode** field).		
errorCode	N 1..3	No

Response code: · 0 — a successful transaction · Any other number — an error occurred when processing the request		
errorMessage	AN 1..512	No
Description of the error in the language that was sent in the **language** parameter of the request.		
recurrenceId	N6	Yes for recurring payments
Identifier that is used for a series of all subsequent payments that will be made until the recurrence period ends.		

Error codes (errorCode) and messages (errorMessage)

Code	Message	Description
0	No system error.	No errors.
1	Order with given order number has already been processed or the **childId** is incorrect.	The specified order number is already associated with a completed order, or the **childId** value in the request is not valid. This typically happens when a system error occurs, duplicate

		order numbers are used, or incorrect data is transferred.
	Order with this number was registered, but was not paid.	A payment processing issue has occurred.
	Submerchant is blocked or deleted.	The submerchant's account is either blocked or does not exist as it has been deleted.
3	Unknown currency.	The specified currency is not registered in the system.
4	Order number is not specified.	The **orderNumber** parameter value was not specified in the request.
	Merchant user name is not specified.	The **clientId** parameter value was not specified in the request.
	Amount is not specified.	The **amount** parameter value was not specified in the request.
	Return URL cannot be empty.	The **returnUrl** parameter value was not specified in the request.
	Password cannot be empty.	The **password** parameter value was not specified in the request.

5	Incorrect value of a request parameter.	The value of some request parameter was specified incorrectly.
	Incorrect value in the **Language** parameter.	The system does not recognize the language code provided for this parameter.
	Access is denied.	The user does not have the necessary permissions to access this resource.
	Merchant must change the password.	A merchant needs to update their password for security reasons.
	Invalid **jsonParams[]**.	Some parameter in the **jsonParams** structure has an incorrect value or a character in the JSON string does not conform to the JSON syntax rules.
7	System error.	Software or hardware issue or malfunction.
14	**PaymentWay** is invalid.	The specified payment method is not accepted by the system or the merchant.

Request example

```
http://<host:port>/payment/rest/register.do?userName=apiuser&password=
 apiuserpassword&orderNumber=26122017090900&amount=888&currency=978
 &returnUrl=http://www.return.url.com
```

Example of the orderBundle parameter

```
"orderBundle" : {
  "cartItems" : {
   "items" : [
   {
            "name" : "item1",
    "positionId" : 1,
    "quantity" : {
            "value" : 2.0,
     "measure": "quantity"
            },
    "itemAmount" : 3025,
    "itemCode" : "code1"
   },
   {
            "name" : "item2",
    "positionId" : 2,
    "quantity" : {
            "value" : 2.0,
     "measure": "quantity"
            },
    "itemAmount" : 3025,
    "itemCode" : "code2"
   }
```

Response example

{"orderId":"5f9adf5a-4796-4668-9285-07adf3c9a1aa","formUrl":
"http://10.7.32.60/payment/merchants/Merchant/payment_en.html?mdOrder
=5f9adf5a-4796-4668-9285-07adf3c9a1aa"}

4.2 Order completion request (for two-phase payments)

The **deposit.do** request is used to complete a preauthorized payment in the case of a two-phase payment.

Users can perform this request if they have the relevant permissions in the system.

Request parameters

Name	Type	Mandatory
userName	AN 1..100	Yes
Login of the API user on whose behalf requests are processed for a specific merchant.		
password	String	Yes
User’s password.		
orderId	ANS36	Yes

Unique order number generated by EPG after the registration of the order.		
amount	N 1..12	Yes
Order amount in the minor denomination (for example, cents).		

IMPORTANT: If you specify 0 in the **DepositAmount** parameter, completion occurs for the entire preauthorized amount.

Response parameters

Name	Type	Mandatory
errorCode	N 1..3	No
Response code: · 0 — a successful transaction · Any other number — an error occurred when processing the request		
errorMessage	AN 1..512	No
Information message about the transaction result: a success message or the description of an error.		

Error codes (errorCode) and messages (errorMessage)

Code	Message	Description
0	No system error.	No errors.
5	Deposited amount is exceeding approved amount	The transaction amount exceeds the approved limit.
	Deposited amount is exceeding tolerance	A payment or deposit amount is greater than the acceptable difference or allowance set by a system or process.
	Deposit amount is invalid	The specified amount is not accepted for the deposit. This could be due to a few reasons, including incorrect input, exceeding limits, or issues with the bank or payment method.
	Deposit amount must be at least 1 or equal to zero	The **amount** parameter value is not specified or specified incorrectly in the request.
6	**orderId** is empty.	The **orderId** parameter value is not specified in the request.
	No such order.	The order meeting the specified criteria was not found.

7	System error.	Software or hardware issue or malfunction.
	Transaction is processing now. Please, repeat later.	The payment or transfer is being processed but may take some time to complete.
	Payment must be in approved state.	The payment must be authorized by the payment processing system or the issuer.
	Deposit is impossible chargeback flag is present.	The payment is blocked due to a previous chargeback history. This flag suggests the payment processor suspects the transaction may be fraudulent or that the cardholder is likely to dispute the charge.
9	Payment must be processed according to Multiple completions flow.	A system should have an ability to divide a single transaction into multiple smaller payments that are processed separately.
	Tolerance parameter for this payment system is not set.	The system cannot process the payment because a tolerance level has not been configured.

Request example

```
http://<host:port>/payment/rest/deposit.do?userName=apiuser&password=
 apiuserpassword&&amount=100&orderId=a2565183-fa21-4b8b-84dc-15b34933424d
```

Response example

```
{"errorCode":"0","errorMessage":"Success"}
```

4.3 Payment request

The **paymentorder.do** method is used for a payment request. The payment card data is validated in accordance with the table below:

Name	Description	Validation
PAN	Full card number	Luhn validation (checking if the card number is real) — the number of digits in the card number ranges from 13 to 19.
CVC	Card Verification Code (CVC)	Three or four digits.
YYYY, MM	Year, month of card expiration	Date (the current or a future one). If the card expires in the current month, a payment is possible until the end of the month.

TEXT	Cardholder name	This parameter is verified according to the following criteria. · Acceptable characters are: Latin letters, 0-9, $,), (, -, . , a space · Cardholder name must start with a letter · Minimum length: one Latin letter · Maximum length: 26 characters · Null is valid · Uppercase and lowercase are acceptable

Only **POST** requests are supported.

Request parameters

Name	Type	Mandatory
userName	AN 1..100	Yes
Login of the API user on whose behalf requests are processed for a specific merchant.		
password	String	Yes
User's password.		
MDORDER	ANS36	Yes

Order number generated by EPG after the registration of the order.		
$PAN	N 13...19	Yes
Card number.		
$CVC	N3..4	Conditional
Card security code: CVV2 (Card Verification Value 2), CVC2 (Card Verification Code 2), and analogous codes. Whether this parameter is mandatory for payments that use the respective payment method, depends on the **Can pay by binding without CVV2/CVC2** and **Can pay by card without CVV2/CVC2** options: · If the option is selected for a merchant, this parameter is optional for this merchant. · If the option is not selected, this parameter is mandatory for the merchant. The options are configured through the administrative portal UI.		
YYYY	N..4	Yes
Year when the card expires.		
MM	N..2	Yes
Month when the card expires.		

TEXT	A..512	No
Cardholder name.		
language	A..2	Yes
Language of the information (or error) message that is to be sent in a response to this request.		
ip	AN..19	No
IP address of the customer.		
email	ANS..*	No
Customer’s email address to which notifications are sent, if notifying customers is enabled.		
bindingNotNeeded	Boolean	No
Binding settings: · true — disables the creation of a binding after a successful payment (the client identifier sent with the order registration request is deleted from the order details after the payment). · false — a successful payment may result in the creation of a binding (under the relevant conditions). This is the default value.		

jsonParams[]	AN..1024	No
Fields used to store additional information. The type is as follows: {"param":"value","param2":"value2"} **Note**: The parameter name length is 255 characters or less and the value length is 1024 characters or less. See JSON parameter list for information about which parameters are passed.		
threeDS2Params[]		Conditional
Parameters of the 3-D Secure 2 protocol authentication. The **threeDS2Params** parameter is a JSON-based structure. Depending on the type of channel interface that is used to initiate the transaction (**deviceChannel**), the parameter is mandatory or optional: · Optional for browser-based authentication · Mandatory for application-based authentication		
clientBrowserInfo		No
JSON structure containing the client browser information for 3DS2. See the description of the clientBrowserInfo contents.		
threeDSCompInd	String	No

Specifies whether the 3DS Method Completion notification was received by the merchant:

- Y — 3DS Method Completion notification from the issuer ACS was received by the merchant using the address specified by the **threeDSMethodNotificationURL** parameter of the **checkPreliminary** method (see Request to check card eligibility for 3DS2 before the main request).
- N — 3DS Method Completion notification from the issuer ACS was not received by the merchant.

clientId	ANS 1..255	No

Customer identifier in the merchant system. This parameter is mandatory for bindings.

bindingId	AN..255	No

Identifier of the binding that was created earlier (see Managing bindings). It can only be used if the merchant has the permission to work with bindings.

If this parameter is sent in the **registerOrder** request, the following is executed:

- This order can only be paid by binding.
- The payer is redirected to a payment page where the CVC must be entered.

dccState	AN..32	No

Dynamic Currency Conversion state:

- USED — the purchase transaction is initiated with the updated currency amount.

· DECLINED — the purchase transaction is initiated with the original amount.		
dccUuid	ANS36	No
UUID of dynamic currency conversion data.		
threeDs2ReturnUrl	ANS 1..512	No
Return URL for user redirect from the issuer ACS after 3DS2 authentication is completed. If it is not specified, the user will be redirected to the specific EPG URL.		
aReqFieldsOverride		No
JSON structure containing AReq override data for the specific transaction. See the description of the aReqFieldsOverride structure contents.		
originalDsTransId	JWS	No
Directory Server Transaction ID, UUID (only for 3DS2 transactions). This parameter is present in the request if **recurringPaymentManagedByMerchant =** true.		
originalAcsTransId	JWS	No
Original transaction identifier in ACS (only for 3DS2 transactions).		

originalInitTransactionDate	N8	No
Initiating transaction date (only for 3DS2 transactions).		
originalInstallmentNumber	N1..99	No
Initiating installment payment number.		

The 3DS authentication result is also passed in the **jsonParams** block of **paymentorder.do**.

Response parameters

Name	Type	Mandatory
errorCode	N 1..3	Yes
Code of the error that occurred during the payment process.		
errorMessage	AN 1..512	No
Description of the error in the language that was sent in the **language** parameter of the request.		
info	AN..512	Yes

Result of the payment attempt: · Your order is proceeded, redirecting... · Operation declined. Please check the data and available balance of the card. Redirecting... · Sorry, payment cannot be completed. Redirecting... · Payment declined. Please, contact the merchant. Redirecting... · Payment declined. Please, contact your bank. Redirecting... · Cannot connect to your bank. Please, contact your bank. Redirecting... · Processing timeout. Please, try again later. Redirecting...		
redirect	AN..512	No
URL to which the customer is redirected after executing the payment, depending on the payment result.		
termUrl	AN..512	No
Return address from ACS for the customer to complete the payment. This parameter is used in payments that require additional authentication on the issuing bank's ACS.		
acsUrl	AN..512	No
URL of the ACS server. This parameter is used in payments that require additional authentication on the issuing bank's ACS.		
paReq	AN..512	Conditional

Payer Authentication Request is a message sent from the MPI to ACS via the cardholder device. PAReq requests the issuer to authenticate its cardholder and contains the cardholder, merchant, and transaction-specific information necessary to perform authentication. It is used in 3-D Secure 1. This parameter is not used in payments that do not require additional authentication on the issuing bank's ACS.		
cReq	AN..512	Conditional
Challenge Request message. EMV 3-D Secure message sent by the 3DS SDK or 3DS Server where additional information is sent from the cardholder to the ACS to support the authentication process. It must be present for 3-D Secure 2 if a cardholder challenge is required.		
cardholderInfo	ANS..128	No
Optional information displayed to the cardholder by the merchant on a Frictionless transaction authentication. The information is received from ACS or the issuer. It provides the details on the authentication, for example, a request to call the issuer if additional authentication is needed. This field is passed for 3-D Secure 2 transactions.		
additionalResponseCodes	AN..1024	No
Result of the cardholder billing address check. This parameter is transferred in case the AVS check is used (the **AVS enabled** option must be enabled for the merchant).		

The additionalResponseCodes block contents are described below.		
orderStatusCode	N2	No
A numeric code that specifies the order status in SmartVista E-Commerce Payment Gateway. It is absent if a matching order was not found. The possible values of the field are listed in the **orderStatusCode values** table below.		
orderStatusName	AN 1..100	No
Order processing status: · started — the order was created in EPG. This is the initial order status. · payment_approved — the payment was authorized. · payment_declined — the payment was declined. · payment_void — the payment was reversed. · payment_deposited — the payment was deposited. · refunded — the payment refund was performed. · card_added — the binding was created (for binding register orders only). · card_modified — the binding was modified (for binding modify orders only). · card_verified — the card was verified (for card verification orders only). · recurring_template_added — the recurring payment template was added (for Create recurring payment template orders only).		
transactionData	JWS	No
See the description of the Transaction Data block below.		

Parameters of the additionalResponseCodes block

Name	Type	Mandatory
type	AN..3	No
Type of additional response code. The only available value is AVS.		
responseCode	A1	No
AVS response code.		
responseMessage	AN..512	No
AVS response message.		

Transaction Data block

A single signed data block is used to transfer all original transaction data fields to and from a PCI DSS merchant, instead of multiple separate fields.

An RSA key pair with **CertificateType.TRANSACTION_DATA_SIGNATURE** must be generated for the merchant.

The block includes the following:

Header

Field	Description
alg	Algorithm (RS256).
kid	ID of the corresponding record in the key_mgmt_certificate table.
typ	Header type (JOSE).
merchantLogin	Login of the merchant.

Payload

Field	Description
mdOrder	This field is reserved for future use.
transDate	This field is reserved for future use.
networkReferenceData	Network reference data returned from the authorization host.
transactionDataElements	Data elements of the authorization request.

isSsl	Specifies whether 3DS2 authentication was not performed for the transaction.
dsTransId	Directory Server Transaction ID, UUID (only for 3DS2 transactions).
acsTransId	Transaction ID in ACS (only for 3DS2 transactions).
initTransactionDate	Initiating transaction date (only for 3DS2 transactions).
version	Constant value of 1. This field is reserved for future use.
panSalt	Base64-encoded 16 bytes random salt (the resulting string will be approximately 24 character long; from **SecureRandom**).
panHash	Base64 SHA-256 hash of byte[] = PAN.getBytes + salt.bytes.

The block validation fails if:

- The field is missing for MIT requests with **recurringPaymentManagedByMerchant=**true or **bindingManagedByMerchant=**true.
- A different **merchantLogin** is used.
- The key provided in the JWS header cannot be found for the current merchant.
- Signature validation fails (both technical and not errors).

Error codes (errorCode) and messages (errorMessage)

Code	Message	Description
0	No system error.	No errors.
5	There are no more payment attempts.	
5	System or internal error.	

OrderStatusCode values

The **OrderStatusCode** field may have the following values:

Code	Description
0	Order registered, but not paid. This is not the final status of the transaction. The order can be paid for again. The intermediary statuses and the history of each attempt can be observed in the **actionCode** and **actionCodeDescription** fields by processing the **getOrderStatusExtended.do** request.
-1	This status replaces the decline status which is returned if none of the specified statuses is suitable for the transaction.

1	Transaction has been approved (for a one-phase payment).
1	Preauthorization amount was put on hold (for a two-phase payment).
2	Amount was deposited successfully.
3	Transaction has been reversed. This represents the transaction final status that cannot be modified.
4	Transaction has been refunded.
6	Transaction was declined. This represents the transaction final status that cannot be modified.
7	Card was added.
8	Card was updated.
9	Card was verified.
10	Recurring template was added.

Request example

```
http://<host:port>/payment/rest/paymentorder.do?userName=apiuser&password=
 apiuserpassword&MDORDER=aca0f48d-9e99-4437-9335-a50a42c363da
 &$PAN=4000010000000050&$CVC=123&YYYY=2020&MM=06&TEXT=Card Holder&language=EN
```

Response example in case of a payment that does not require additional authentication on the issuing bank's ACS

```
{"redirect":"https://msk-ecom-wls02.bpc.in:443/epg/merchants/root/finish.html?orderId=
  ffd85ad3-1851-430f-99da-a78ecd228d4d&lang=en&status=payment_deposited",
  "info":"Your order is proceeded, redirecting...","errorCode":0,"orderStatusCode":2,
  "orderStatusName":"payment_deposited"}
```

Response example in case of a 3DS-payment that requires additional authentication on the issuing bank's ACS

```
{"info":"Your order is proceeded, redirecting...","acsUrl":
 "http://10.7.32.60:6001/acs/pareq","paReq":"eJxVUdFygjAQ/BWG9xKioNQ54tCi
 rQ+oo/QDKNwAVYIGENuvbyJQ60Nmdu8um80ezK/FUbugqPKS\nuzo1TF1DHpdJzlNX/wiXT4
 4+ZxBmAtHfY9wIZBBgVUUpanni6kWVGlRnsPV2eGbQKzEpZIyADFRe\nEXEW8ZpBFJ9fVmtm
 jaYT0wTSUyhQrHw2BdIB4FGBbI8X5Is8zep1zhHIrQhx2fBafLOJNQYyEGjE\nkWV1fZoR0r
 atkZZlekRDNEBUB8jdwbZRqJJK1zxhge+13Vn8bMLD9yZM7eDLs4Lw4AJRE5BENbKR\nSS3T
 Nm2NOjPTmVnS+60OUaEsMGrKrvpQR+GkXvEee/9rILMUMurhHwMDvJ5KjnJC5veHgdxdv76r
 \nFONaxuSveRIvnyfeWxjsNk4QHXJaFZ/V+NIuVba3IaWYy4ioTTtJRYAoGdKvjfQrlehh1b
 8el7SS\n","termUrl":"http://10.7.32.60/payment/rest/finish3ds.do",
 "errorCode":0}
```

If 3-D Secure is required to perform a payment, then after the response to the payment request has been received, the merchant must redirect the customer to ACS.

To redirect the customer to ACS, the merchant must perform a redirection to the address specified in the **acsUrl** parameter with the body of the request. The request must be in the **POST** format.

Depending on the integration scheme, after passing the authentication on ACS, the customer is redirected to the merchant or to EPG. The processes of payment completion for each of these cases are listed below.

Return to the payment gate from ACS

In the classic scheme, the issuing bank's ACS authenticates the cardholder and redirects them to the SmartVista E-Commerce Payment Gateway sending the Payer Authentication Response (PARes). If the cardholder was authorized by ACS successfully, EPG authorizes the order. Having received PARes, EPG processes the authentication results and sends the authorization request to the acquirer's system.

Return to the merchant from ACS using Finish 3DS

The scheme uses an additional method, Finish 3DS. In this scheme, the issuing bank's ACS authenticates a cardholder and redirects the cardholder to the merchant. The PARes from ACS is sent to the merchant. The merchant then transfers it to EPG via the **finish3dsPayment.do** method.

4.4 Request to check card eligibility for 3DS2

The **/api/v1/3ds2/check** request is used to check whether the entered PAN is eligible for 3DS2 transactions. This method is called in parallel with **paymentorder.do**.

Only **POST** requests are supported.

Request parameters

Name	Type	Mandatory
userName	AN 1..100	Yes
Login of the API user on whose behalf requests are processed for a specific merchant.		

password	String	Yes
User's password.		
mdOrder	ANS36	Yes
Unique order number generated by EPG after the registration of the order.		

Response parameters

Name	Type	Mandatory
completed	Boolean	Yes
Completion indicator. It specifies whether the card eligibility check has been completed: · true — card eligibility check has been completed. · false — card eligibility check has not been completed. If a value of false is received, the API call should be repeated after some time.		
is3Ds2Eligible	Boolean	Conditional
Specifies whether the card is eligible for 3DS2. This parameter value is set if the card eligibility check has been completed (**completed** = true). Possible values: · true — card is eligible for 3DS2.		

· false — card is not eligible for 3DS2.		
threeDSServerTransID	ANS36	Conditional
Transaction identifier (UUID) in terms of the 3DS Server. This parameter value is set if the card eligibility check has been completed (**completed = true**).		
threeDSMethodURL	ANS..256	Conditional
URL of the issuer's ACS ThreeDS Method (a method used by ACS to collect cardholder browser information on its own). This parameter value is set if the card eligibility check has been completed (**completed = true**). If the issuer's ACS does not support the ThreeDS Method, the field is blank or missing.		
threeDSMethodDataPacked	ANS..512	Conditional
If issuer's ACS does support ThreeDS Method, this field will contain the so called ThreeDS Method Data. Otherwise the field is blank or missing. This parameter value is set if the card eligibility check has been completed (**completed = true**).		
threeDSMethodURLServer	ANS..256	Conditional
URL of 3DS Server API which provides a default means of collecting cardholder browser information. This parameter value is set if the card eligibility check has been completed (**completed = true**).		

4.5 Request to check card eligibility for 3DS2 before the main request

The **/api/v1/3ds2/checkPreliminary** method is used to check the card eligibility for 3DS2 before executing the **paymentorder.do** or **/p2p/perform** request.

Request parameters

Name	Type	Mandatory
userName	AN 1..100	Conditional
Login of the API user on whose behalf requests are processed for a specific merchant.		
password	String	Conditional
User's password.		
mdOrder	ANS36	Yes
Order number generated by EPG after the registration of the order.		
pan	N 13...19	Yes

Primary account number.		
threeDSMethodNotificationURL	AN 1..255	No
URL used by the merchant to receive a notification about completion of the 3DS Method ACS from the issuing bank's ACS. If this URL is not specified, 3DSS populates this field with its own URL for notification from the issuing bank's ACS.		

Response parameters

Name	Type	Mandatory
threeDSProtocolVersion	ANS	Yes
3DS protocol version: · 3DSv1 — this version is received if the card belongs to the card range eligible for 3DS1. · 3DSv2 — this version is received if the card belongs to the card range eligible for 3DS2. · None — this version is received if the card belongs to none of the card ranges (neither eligible for 3DS1 nor eligible for 3DS2).		
threeDSServerTransID	ANS36	Conditional
Transaction identifier (UUID) in terms of the 3DS Server. This parameter value is set if the 3DS protocol version 2 is used (**threeDSProtocolVersion = 3DSv2**).		

threeDSMethodURL	ANS..256	Conditional
URL of the issuer's ACS ThreeDS Method (a method used by ACS to collect cardholder browser information on its own). This parameter value is set if the 3DS protocol version 2 is used (**threeDSProtocolVersion** = 3DSv2). If the issuer's ACS does not support the ThreeDS Method, the field is blank or missing.		
threeDSMethodDataPacked	ANS..512	Conditional
ThreeDS Method Data. This parameter value is set if the 3DS protocol version 2 is used (**threeDSProtocolVersion = 3DSv2**) and if ACS of the issuing bank supports the 3DS method. Otherwise the field is blank or missing.		
threeDSMethodURLServer	ANS..256	Conditional
URL of 3DS Server API which provides a default mean of collecting cardholder browser information. This parameter value is set if the 3DS protocol version 2 is used (**threeDSProtocolVersion = 3DSv2**).		

4.6 Recurring payments

This section contains the description of API methods used for recurring payments. A *recurring payment* is a payment model where the customers authorize the merchant to pull funds from their accounts automatically at regular intervals for the goods and services provided to them on an ongoing basis.

Recurring payments are performed as follows:

1. The first recurring payment is registered by the **`register.do`** request sent by a merchant.

This request contains necessary recurring parameters such as **recurrenceType**, **recurrenceFrequency**, **recurrenceStartDate**, **recurrenceEndDate**.

2. EPG registers the order and sends a response with **recurrenceId**.
3. The merchant sends the **paymentorder.do** request with card data to process the first recurring transaction.
4. The second and subsequent recurring payments are made using the **recurringPayment.do** method. If **recurenceType=MANUAL**, the merchant may initiate a **recurrentPayment.do** request any time regardless of the date specified by the **recurrenceFrequency** parameter. The system does not validate whether the payment date matches the scheduled installment plan.

EPG also enables users to perform the following actions:

- Create templates for recurring payments using the **createRecurringTemplateNoPayment.do** method
- Get a list of transactions processed for a specific recurring payment using the **getRecurrentReport** method
- Update information about a recurring payment the using the **updateRecurrent.do** method
- Get information about a specific recurring payment using the **getRecurrentDetails** method

4.6.1 Recurring payment request

The **recurringPayment.do** method processes subsequent recurring payments based on a respective **recurrenceId** generated on the first payment registration (see the description of **register.do**).

> **Note**: This operation is only used for manual recurring payments.

To be able to execute this method, a merchant must have the **Recurrent payments allowed** permission enabled. A merchant must have at least one user with the **Use Merchant API** and **Recurrent payment allowed** permissions.

Note: A recurring payment must be associated with an active acquirer. If a recurring payment is associated with an acquirer that has been deleted or deactivated, the default acquirer is set for the next recurring payment.

Request parameters

Name	Type	Mandatory
userName	AN 1..100	Yes
Login of the API user on whose behalf requests are processed for a specific merchant.		
password	String	Yes
User's password.		
recurrenceId	N6	Yes
Identifier that is used for a series of all subsequent payments that will be made until the recurrence period ends.		
updateAccount	AN..32	No

Specifies whether account information for bindings and/or recurring payments is updated: · YES — all bindings and recurring payments are updated. · NO — account data is not updated. · FORCE — account data is updated by force in the online mode via ABU (Account Billing Updater).		
amount	N 1..12	No
Transaction amount. The amount is updated if the **Recurrent payment with fluctuated amount allowed** option is enabled for the merchant.		

The amount of the current recurring payment can be changed if the following conditions are met:

- The **Recurrent payment with fluctuated amount allowed** option is enabled for the merchant.
- The **amount** parameter with the new amount specified is added to the current **recurringPayment.do** request.

The amount will be changed only for this specific instance of the recurring payment, all subsequent payments by default will have the amount specified on the first payment instance.

Response parameters

Name	Type	Mandatory

errorCode	N 1..3	No
Response code: · 0 — a successful transaction · Any other number — an error occurred when processing the request		
errorMessage	AN 1..512	No
Information message about the transaction result: a success message or the description of an error.		
orderId	ANS36	No
Unique order number generated by EPG after the registration of the order. It is absent if the order registration has failed (the error is described in the **ErrorCode** field).		

Error codes (errorCode) and messages (errorMessage)

Code	Message	Description
0	No system error.	No errors.

5	Access denied.	The user does not have the necessary permissions to access this resource.
5	The user must change their password.	The user needs to update their password for security reasons.
5	Recurring payment canceled by the cardholder	Cancellation of the recurring payment by the cardholder.

Request example

```
http://<host:port>/payment/rest/recurringPayment.do?userName
 =apiuser&password=apiuserpassword&recurrenceId=46
```

Response examples

Successful payment

```
{"error":"Successful recurring payment","errorCode":0,"orderId":
  "4fb3b8d1-b19f-4bf7-9c59-a49ec30bafd3","errorMessage":"Successful recurring payment"}
```

Recurring payment canceled by the cardholder

```
{
        "error": "Terminate Subscription",
    "errorCode": 5,
    "orderId": "4d302502-2c34-446a-8dbe-1ebacfb7d0fa",
```

```
    "errorMessage": "Terminate Subscription"
}
```

4.6.2 Recurring payment update request

The **updateRecurrent.do** service is used to update information about a recurring payment.

To be able to execute this method, a merchant must have the **Recurrent payments allowed** permission enabled. A merchant must have at least one user with the **Use Merchant API** and **Recurrent payment allowed** permissions.

> **Note:** A recurring payment must be associated with an active acquirer. If a recurring payment is associated with an acquirer that has been deleted or deactivated, the default acquirer is set for the next recurring payment.

Request parameters

Name	Type	Mandatory
userName	AN 1..100	Yes
Login of the API user on whose behalf requests are processed for a specific merchant.		
password	String	Yes
User’s password.		

recurrenceId	N6	Yes
Identifier that is used for a series of all subsequent payments that will be made until the recurrence period ends.		
recurrenceType	String	Yes
Way in which subsequent recurring payments are processed: · AUTO — for automatic handling of payments scheduled on the EPG side. · MANUAL — for manual processing of recurring payments scheduled on the merchant side.		
recurrenceFrequency	String	For MANUAL: No For AUTO: Yes
Specifies how often recurring payments are made. This parameter is used only for automatic recurring payments and not required for manual payments. **Note**: This parameter is ignored for MANUAL recurring payments.		
recurrenceStartDate	N8	No

Date when the recurring payment starts in the following format: *YYYYMMDD*. *YYYYMMDD* 00:00:00 time is used for the start date. This parameter can be used for installment payments and AUTO recurring payments. This parameter is ignored for MANUAL recurring payments. The minimum **recurrenceStartDate** parameter value is tomorrow for AUTO recurring payments and installments.		
recurrenceEndDate	N8	No
Date when the recurring payment ends in the following format: *YYYYMMDD*. *YYYYMMDD* 23:59:59 time is used for end date. Possible values: · Date in the following format: <*yyyymmdd*>. This value is later used as the end of the day (*yyyy-mm-dd* 23:59:59) · Now — cancel the recurring payment immediately. If the Terminate Subscription code was received from SVFE, the **recurrenceEndDate** is set as the current date and time for recurring and installment payments. Future scheduled payments are canceled. **Note**: This parameter is ignored for MANUAL recurring payments.		

Response parameters

Name	Type	Mandatory

errorCode	N 1..3	No
Response code: · 0 — a successful transaction · Any other number — an error occurred when processing the request		
errorMessage	AN 1..512	No
Information message about the transaction result: a success message or the description of an error.		

Request example

```
http://<host:port>/payment/rest/updateRecurrent.do?userName=apiuser&password
 =apiuserpassword&recurrenceId=61&recurrenceType=AUTO&recurrenceFrequency=0 0 * * * ?
```

Response example

```
{"errorCode":0,"recurrenceId":61}
```

4.6.3 Recurring payment details request

The **getRecurrentDetails** service is used to obtain information about a specific recurring payment,

To be able to execute this method, a merchant must have the **Recurrent payments allowed** permission enabled. A merchant must have at least one user with the **Use Merchant API** and **Recurrent payment allowed** permissions.

Request parameters

Name	Type	Mandatory
userName	AN 1..100	Yes
Login of the API user on whose behalf requests are processed for a specific merchant.		
password	String	Yes
User's password.		
recurrenceId	N6	Yes
Identifier that is used for a series of all subsequent payments that will be made until the recurrence period ends.		

Response parameters

Name	Type	Mandatory
errorCode	N 1..3	No

Response code: · 0 — a successful transaction · Any other number — an error occurred when processing the request		
errorMessage	AN 1..512	No
Description of the error in the language that was sent in the **language** parameter of the request.		
recurrenceId	N6	Yes
Identifier that is used for a series of all subsequent payments that will be made until the recurrence period ends. It is returned for the purpose of validation.		
recurrenceType	String	Yes
Way in which subsequent recurring payments are processed: · AUTO — for automatic handling of payments scheduled on the EPG side. · MANUAL — for manual processing of recurring payments scheduled on the merchant side.		
recurrenceFrequency	String	Yes
Specifies how often recurring payments are made. This parameter is used only for automatic recurring payments and not required for manual payments.		

recurrenceStartDate	N8	Yes
Date when the recurring payment starts in the following format: *YYYYMMDD*. *YYYYMMDD* 00:00:00 time is used for the start date.		
recurrenceEndDate	N8	No
Date when the recurring payment ends. This parameter is mandatory for installments in the following format: *YYYYMMDD*. The default value (if empty) is infinite. *YYYYMMDD* 23:59:59 time is used for the end date. If the Terminate Subscription code was received from SVFE, the **recurrenceEndDate** is set as the current date and time for recurring and installment payments. Future scheduled payments are canceled.		
amount	N 1..12	No
Amount of the recurring payment.		
currency	N3	No
Currency of the recurring payment.		
paymentsHistory	Not applicable	Conditional

List of transactions associated with the recurring payment (see the description of paymentsHistory block parameters below). This field is available if **recurring.payment.history.display.limit** > 0 in the system settings. The number of records in this list is limited by the **recurring.payment.history.display.limit** parameter value.		
scheduledPayments	Not applicable	Conditional
List of remaining payments (see the description of scheduledPayments block parameters below). This field is available if **recurring.payment.scheduled.display.limit** > 0 in the system settings. Only the nearest scheduled payments according to **recurring.payment.scheduled.display.limit** value will be displayed in the SmartVista E-Commerce Payment Gateway web interface.		
paymentsNumber	Number	No
Total number of recurring payments that the customer has paid.		
nextPaymentDate	ANS	No
Next payment date in the following format: *yyyy.MM.dd HH:mm:ss*.		

Parameters of the paymentsHistory block

Name	Type	Mandatory
orderNumber	AN 1..32	No
Number (identifier) of the order in the merchant's online store system.		
mdOrderId	ANS 36	No
Unique order number generated by EPG after the registration of the order.		
state	N..9	No
Payment status.		
paymentDate	ANS	No
Payment date in the following format: *yyyy.MM.dd HH:mm:ss*.		
formattedAmount	N 1..19	No
Formatted payment amount (with a dot separating the minor currency units).		
currency	N3	No

Numeric currency code.

Parameters of the scheduledPayments block

Name	Type	Mandatory
number	N3	No
Payment order number. A value of 1 means the nearest payment.		
paymentDate	ANS23	No
Payment date.		
formattedAmount	N 1..19	No
Formatted payment amount (with a dot separating the minor currency units).		
currency	N3	No
Payment currency code in the ISO 4217 format.		

Request example

```
http://<host:port>/payment/rest/getRecurrentDetails.do?
 userName=apiuser&password=apiuserpassword&recurrenceId=61
```

Response example


```
{
    "errorCode": 0,
    "frequency": "0 0/2 * * * ?",
    "recurrenceId": 1622,
    "startDate": "20250218",
    "endDate": "20251124",
            "type": "AUTO",
    "paymentsNumber": 1008,
    "nextPaymentDate": "2025.03.02 12:52:00",
            "amount": "60.50",
    "currency": "840",
    "paymentsHistory": [{"orderNumber": "1740755160473",
    "mdOrderId": "3bac8cca-0e3f-4b0f-9baf-c0f8d7aeab6a",
            "state": "DECLINED", "paymentDate": "2025.03.02 12:40:02",
    "formattedAmount": "60.50", "currency": "840"},...]}
```


4.6.4 Recurring payment report request

The service **getRecurrentReport** is used to obtain a list of transactions processed for a recurring payment.

To be able to execute this method, a merchant must have the **Recurrent payments allowed** permission enabled. A merchant must have at least one user with the **Use Merchant API** and **Recurrent payment allowed** permissions.

Request parameters

Name	Type	Mandatory
userName	AN 1..100	Yes
Login of the API user on whose behalf requests are processed for a specific merchant.		
password	String	Yes
User’s password.		
recurrenceId	N6	Yes
Identifier that is used for a series of all subsequent payments that will be made until the recurrence period ends.		

Response parameters

Name	Type	Mandatory
errorCode	N 1..3	No
Response code: · 0 — a successful transaction · Any other number — an error occurred when processing the request		

errorMessage	AN 1..512	No
Description of the error in the language that was sent in the **language** parameter of the request.		
report[]	Not applicable	Yes
block containing the list of transactions and their details for the specified recurring payment. The report block contents are detailed below.		

The report block

Name	Type	Mandatory
recurrenceId	N6	Yes
Identifier that is used for a series of all subsequent payments that will be made until the recurrence period ends. It is returned for the purpose of validation.		
transactionId	N8	Yes
Identifier of the currently processed recurring payment.		

mdOrder	ANS36	Yes
Order number generated by EPG after the registration of the order.		
status	String	Yes
Status of the current payment.		
date	ANS	Yes
Date of the payment in the following format: *<yyyymmdd hh:mm:ss>*.		

Request example

```
https://<host:port>/payment/rest/getRecurrentReport.do?
 userName=apiuser&password=apiuserpassword&recurrenceId=61
```

Response example

```
{"errorCode":0,"report":[{"recurrenceId":61,"tranasctionId":
 11220039,"status":"DEPOSITED","date":"2018-09-14 07:44:53"},
 {"recurrenceId":61,"tranasctionId":11220040,"status":"DEPOSITED",
 "date":"2018-09-14 07:45:40"}]}
```

4.6.5 Request to register a template for recurring payments

The **registerRecurringTemplateNoPayment.do** method is used to register a request to create a template for recurring payments. This request is then completed by the **createRecurringTemplateNoPayment.do** request with the **mdOrder** parameter.

Request parameters

Name	Type	Mandatory
userName	AN 1..100	Yes
Login of the API user on whose behalf requests are processed for a specific merchant.		
password	String	Yes
API user password.		
orderNumber	AN 1..32	No
Number (identifier) of the order in the merchant's online store system. It is unique for every store in the system and is generated on the order registration. If the **Require system to generate order numbers** permission is enabled for the merchant, the order number is generated automatically. Otherwise, the order number data is sent in the API.		
language	A2	No
Language in the ISO 639-1 format. Error messages are sent in this language. If it is not specified, the default language specified in the merchant settings is used.		

Response parameters

Name	Type	Mandatory
errorCode	N 1..3	No
Response code: ● `0` — a successful transaction ● Any other number — an error occurred when processing the request		
errorMessage	AN 1..512	No

Name	Type	Mandatory
Description of the error in the language that was sent in the **language** parameter of the request.		
orderId	ANS36	No
Unique order number generated by EPG after the registration of the order. It is absent if the order registration has failed (the error is described in the **ErrorCode** field).		

Error codes (errorCode) and messages (errorMessage)

Code	Message	Description
0	No system error.	No errors.
1	A required field is absent or invalid.	A mandatory field is not available in the request or its value is incorrect.
5	User must change the password.	The user needs to update their password for security reasons.
7	System error.	Software or hardware issue or malfunction.

Request example

```
http://{{URL}}/epg/rest/public/registerRecurringTemplateNoPayment.do?
userName={{user}}&password={{password}}&language=RU
```

Response example

```
{"errorCode":0,"errorMessage":"Success",
 "orderId":"cc46dc72-d1b4-4c4a-b4ba-5389ce228920"}
```

4.6.6 Request to cancel a recurring payment

The **cancelRecurring.do** method is used to cancel a recurring payment. Only recurring payments which **recurrenceType** = **AUTO** can be canceled.

Request parameters

Name	Type	Mandatory
userName	AN 1..100	Yes
Login of the API user on whose behalf requests are processed for a specific merchant.		
password	String	Yes
API user password.		
recurrenceId	N6	Yes
ID of the recurring payment to be canceled.		

Response parameters

Name	Type	Mandatory
errorCode	N 1..3	No
Response code: ● `0` — a successful transaction ● Any other number — an error occurred when processing the request		
errorMessage	AN 1..512	No
Description of the error in the language that was sent in the **language** parameter of the request.		
recurrenceId	N6	Yes
ID of the canceled recurring payment.		

Error codes (errorCode) and messages (errorMessage)

Code	Message	Description
0	No system error.	No errors.
1	A required field is absent or invalid.	A mandatory field is not available in the request or its value is incorrect.
5	User must change the password.	The user needs to update their password for security reasons.
7	System error.	Software or hardware issue or malfunction.

Request example

```
http://127.0.0.1/epg/rest/public/cancelRecurring.do?userName=user1&password=
user1&recurrenceId=1622
```

Response examples

```
{"errorCode": 0, "recurrenceId": 1622 }
{"errorCode": 5,"errorMessage": "Can not get payment by recurrenceId [1]"}
```

4.7 Order reversal request

The **reverse.do** request is used to cancel a payment for an order. This functionality is available within a limited period (specified by the bank) after a payment has been executed.

The reversal transaction may be performed only once for a payment. If a reversal request results in an error, the next try will fail.

Reversals are available to a merchant only if the merchant has an agreement with its bank to perform reversals. Users must have the relevant permissions to perform the reversal request.

Request parameters

Name	Type	Mandatory
userName	AN 1..100	Yes
Login of the API user on whose behalf requests are processed for a specific merchant.		
password	String	Yes
User's password.		
orderId	ANS36	Yes
Unique order number generated by EPG after the registration of the order.		
language	A2	No
Language code in the ISO 639-1 format. If it is unspecified, SmartVista E-Commerce Payment Gateway uses the default language from the merchant settings. Error messages are also returned in this language.		
email	ANS..*	No

Customer's email address to which notifications are sent, if notifying customers is enabled.		
phone	AN..255	No
Customer's phone number to which SMS notifications are sent, if notifying customers is enabled.		
jsonParams[]	AN..1024	No
Fields used to store additional information. The type is as follows: {"param":"value","param2":"value2"} **Note**: The parameter name length is 255 characters or less and the value length is 1024 characters or less. See JSON parameter list for information about which parameters are passed.		

Response parameters

Name	Type	Mandatory
errorCode	N 1..3	No
Response code: · 0 — a successful transaction · Any other number — an error occurred when processing the request		

errorMessage	AN 1..512	No
Information message about the transaction result: a success message or the description of an error.		

Error codes (errorCode) and messages (errorMessage)

Code	Message	Description
0	No system error.	No errors.
5	Access is denied.	The user does not have the necessary permissions to access this resource.
	The user must change their password.	The user needs to update their password for security reasons.
	Invalid amount.	Incorrect order amount.
	Deposit amount must be zero, or more than 1 currency unit (for example, 1 euro).	The amount must be positive or more that 1 currency unit.
6	Unregistered **OrderId**.	The order with the specified **OrderId** is not registered in the system.

7	System error.	Software or hardware issue or malfunction.
	Incorrect internal values, check the hold and deposited amounts.	Check your internal accounting records.
	Payment must be in a correct state.	An error occurred during a payment process, often related to incorrect billing information.

Request example

```
http://<host:port>/payment/rest/reverse.do?userName=apiuser&password=
 apiuserpassword&orderId=aca0f48d-9e99-4437-9335-a50a42c363da
```

Response example

```
{"errorCode":"0","errorMessage":"Success"}
```

4.8 Refund request

6	Unregistered **OrderId**.	The order with the specified **OrderId** is not registered in the system.

7	System error.	Software or hardware issue or malfunction.
	Payment must be in a correct state.	An error occurred during a payment process, often related to incorrect billing information.

Request examples

```
http://<host:port>/payment/rest/refund.do?userName=apiuser&password=
 apiuserpassword&amount=333&orderId=a2565183-fa21-4b8b-84dc-15b34933424d

http://<host:port>/epg/rest/refund.do?userName={{user}}&password=
 {{password}}&orderId=c2fd92fd-3999-4062-b245-3b8db47cc76a&language=
 en&amount=11&externalRefundId=224c429d-1620-44a0-b87c-0a1cb0161b36
```

Response examples

```
{"errorCode":"0","errorMessage":"Success"}

{"errorCode":"5","errorMessage":
  "Refund with this externalRefundId already exists for merchant"}
```

4.9 Order status request

To obtain the current state of a registered order, send data with **getOrderStatus.do** method (**GET** or **POST**) to the corresponding URL.

The order status is determined by the value of the **OrderStatus** parameter.

Note: The **authCode** field is deprecated.

Request parameters

Name	Type	Mandatory
userName	AN 1..100	Yes
Login of the API user on whose behalf requests are processed for a specific merchant.		
password	String	Yes
User's password.		
orderId	ANS36	Yes
Unique order number generated by EPG after the registration of the order.		
language	A2	No
Language code in the ISO 639-1 format. If it is unspecified, SmartVista E-Commerce Payment Gateway uses the default language from the merchant settings. Error messages are also returned in this language.		

Response parameters

Name	Type	Mandatory
depositAmount	N...5	Yes
Amount to be debited in the order currency. The amount can be less or equal to the preauthorization amount. **Note:** If the amount parameter is not specified, the operation will be completed for the whole preauthorization amount.		
ErrorCode	N 1..3	Yes
Error code.		
ErrorMessage	AN 1..512	Yes
Description of the error in the language that was sent in the **language** parameter of the request.		
currency	N3	Yes
Payment currency code in the ISO 4217 format.		
OrderStatus	N1	Yes

Current order status in SmartVista E-Commerce Payment Gateway. The possible values are listed in the OrderStatus values table below. This parameter is absent if an order with the specified ID is not found.		
OrderNumber	ANS 1..32	No
Number (identifier) of the order in the merchant's online store system. It is unique for every store in the system and is generated on the order registration. If the **Require system to generate order numbers** permission is enabled for the merchant, the order number is generated automatically. Otherwise, the order number data is sent in the API.		
Pan	N 13...19	No
Masked number of the card that was used in payment. It is only specified for paid orders.		
expiration	N6	No
Card expiration date in the following format: *YYYYMM*. It is only specified for paid orders.		
cardholderName	ANS 2..26	No

Name of the cardholder. This parameter is verified according to the following criteria: · Acceptable characters are: Latin letters, 0-9, $,), (, -, . , a space · Cardholder name must start with a letter · Minimum length: 2 characters · Maximum length: 26 characters · Null is valid · Uppercase and lowercase are acceptable		
Amount	N 1..19	No
Order amount in the minor denomination (for example, cents).		
authorizationResponseId (the deprecated name is **approvalCode**)	AN6	No
Payments network authorization code. The field has a fixed length of six characters; it can contain both numbers and letters.		
Ip	NS 15..39	No
IP address of the customer who paid for the order.		
clientId	ANS 1..255	No

Customer identifier in the merchant system. This parameter is mandatory for bindings.		
bindingId	ANS..255	No
Identifier of the binding created when paying this order or created earlier and used to pay for this order. It is present only if the merchant has the permission to create bindings.		
paymentAccountReference	ANS...999	No
Payment account data. This data can be sent if it is available and this field transfer is configured on the host.		
Description	ANS..512	No
Free form description of the order. If the **orderDescription** field in the transaction is empty, the **Description** field is not included in the response.		
responseParameters		No

Result of mapping of SVFE ISO parameters to the transaction fields. The **responseParameters** structure contains parameters selected for the response. These parameters can be selected either on the merchant or at the system level: · For the merchant, the **Response** check box on the **OLTP ISO fields (SVFE)** tab in the merchant's configuration must be selected for the parameter. · At the system level, **response** must be set to true in the **svfe.iso.parameters.mapping** parameter in the system settings. An example of the **responseParameters** structure is provided below: ... "responseParameters":{"respCode_desc":"00 code descriprion","terminal":"TERMINAL1","respCode":"00"} The **responseParameters** parameter is absent in the response if mapping rules are not configured for the parameters.		
transactionData	JWS	No
See the description of the Transaction Data block below.		

OrderStatus values

The **OrderStatus** field may have the following values:

Code	Description
0	Order registered, but not paid. This is not the final status of the transaction. The order can be paid for again. The intermediary statuses and the history of each attempt can be observed in the **actionCode** and

	actionCodeDescription fields by processing the **getOrderStatusExtended.do** request.
-1	This status replaces the decline status which is returned if none of the specified statuses is suitable for the transaction.
1	Transaction has been approved (for a one-phase payment).
1	Preauthorization amount was put on hold (for a two-phase payment).
2	Amount was deposited successfully.
3	Transaction has been reversed. This represents the transaction final status that cannot be modified.
4	Transaction has been refunded.
6	Transaction was declined. This represents the transaction final status that cannot be modified.
7	Card was added.
8	Card was updated.

9	Card was verified.
10	Recurring template was added.

Transaction Data block

A single signed data block is used to transfer all original transaction data fields to and from a PCI DSS merchant, instead of multiple separate fields.

An RSA key pair with **CertificateType.TRANSACTION_DATA_SIGNATURE** must be generated for the merchant.

The block includes the following:

Header

Field	Description
alg	Algorithm (RS256).
kid	ID of the corresponding record in the key_mgmt_certificate table.
typ	Header type (JOSE).
merchantLogin	Login of the merchant.

Payload

Field	Description
mdOrder	This field is reserved for future use.
transDate	This field is reserved for future use.
networkReferenceData	Network reference data returned from the authorization host.
transactionDataElements	Data elements of the authorization request.
isSsl	Specifies whether 3DS2 authentication was not performed for the transaction.
dsTransId	Directory Server Transaction ID, UUID (only for 3DS2 transactions).
acsTransId	Transaction ID in ACS (only for 3DS2 transactions).
initTransactionDate	Initiating transaction date (only for 3DS2 transactions).
version	Constant value of 1. This field is reserved for future use.

panSalt	Base64-encoded 16 bytes random salt (the resulting string will be approximately 24 character long; from **SecureRandom**).
panHash	Base64 SHA-256 hash of byte[] = PAN.getBytes + salt.bytes.

The block validation fails if:

- The field is missing for MIT requests with **recurringPaymentManagedByMerchant=**true or **bindingManagedByMerchant=**true.
- A different **merchantLogin** is used.
- The key provided in the JWS header cannot be found for the current merchant.
- Signature validation fails (both technical and not errors).

Error codes (errorCode) and messages (errorMessage)

Code	Message	Description
0	Success.	The request was processed successfully.
2	The order is declined because of an error in the payment credentials.	The bank or payment processor is refusing to authorize the transaction.
5	Access is denied.	The user does not have the necessary permissions to access this resource.

5	The user must change the password.	The user needs to update their password for security reasons.
5	**orderId** is empty.	The **orderId** parameter value is not specified in the request.
6	Unregistered **order Id**.	The specified order ID is not registered in the system.
7	System error.	Software or hardware issue or malfunction.

Request example

```
https://<host:port>/epg/rest/getOrderStatus.do?orderId=
 a2565183-fa21-4b8b-84dc-15b34933424d&language=en&password=
 apiuserpassword&userName=apiuser
```

Response examples

Success


```
{"expiration":"202006","cardholderName":"Card Holder","depositAmount":100,
 "currency":"978","authorizationResponseId":"077807","ErrorCode":"0",
 "ErrorMessage":"Success","OrderStatus":2,"OrderNumber":"26122017090903",
 "Pan":"400001**0050","Amount":888},"paymentAccountReference": "1234567890"}
```


Declined payment

{"ErrorCode": "2","ErrorMessage": "Payment is declined"}

4.10 Refund status request

To obtain the current status of a refund, send data with the **getRefundStatus.do** method (**GET** or **POST**) to the corresponding URL.

Request parameters

Name	Type	Mandatory
userName	AN 1..100	Yes
Login of the API user on whose behalf requests are processed for a specific merchant.		
password	String	Yes
User's password.		
mdOrder	ANS36	No
Unique order number generated by EPG after the registration of the order.		
externalRefundId	AN 1..30	No

Unique refund identifier in the merchant's system.		
language	A2	No
Language code in the ISO 639-1 format. If it is unspecified, SmartVista E-Commerce Payment Gateway uses the default language from the merchant settings. Error messages are also returned in this language.		

Response parameters

Name	Type	Mandatory
ErrorCode	N 1..3	Yes
Error code.		
ErrorMessage	AN 1..512	Yes
Description of the error in the language that was sent in the **language** parameter of the request.		
mdOrder	ANS36	No
Unique order number generated by EPG after the registration of the order.		

refunds[]	Not applicable	Yes
Information about refunds (see Parameters of the refunds block		

Parameters of the refunds block

Name	Type	Mandatory
actionCode	N3	Yes
Processing system authorization code.		
refNum	AN..24	No
Authorization reference number.		
authorizationResponseId	AN6	No
Payments network authorization code. The field has a fixed length of six characters; it can contain both numbers and letters.		
refundDate	ANS	No

Date when the refund was performed.		
amount	N 1..12	No
Order amount in the minor denomination (for example, cents).		
reversed	Boolean	No
Reversal indicator: · true · false		
externalRefundId	AN 1..30	No
Identifier of the refund in the merchant's system.		

Error codes (errorCode) and messages (errorMessage)

Code	Message	Description
0	Success.	The request has been processed successfully.
2	The order is declined because of an error in the payment credentials.	The bank or payment processor is refusing to authorize the transaction.

5	Access is denied.	The user does not have the necessary permissions to access this resource.
5	The user must change the password.	The user needs to update their password for security reasons.
5	**externalRefundId** is invalid	The **externalRefundId** parameter value in the request is incorrect.
7	System error.	Software or hardware issue or malfunction.

Request example

```
http://<host:port>/epg/rest/public/getRefundStatus.do?userName={{user}}&password=
 {{password}}&language=en&externalRefundId=224c429d-1620-44a0-b87c-0a1cb0161b33&mdOrder=
 c2fd92fd-3999-4062-b245-3b8db47cc76a
```

Response examples


```
{"errorCode":0,"mdOrder":"c2fd92fd-3999-4062-b245-3b8db47cc76a","refunds":
 [{"actionCode":"000","refNum":"702887093561","authorizationResponseId":
 "164396","refundDate":1702887164000,"amount":1,"reversed":false,
 "externalRefundId":"224c429d-1620-44a0-b87c-0a1cb0161b33"}],"error":false}
 {"errorCode":0,"mdOrder":"c2fd92fd-3999-4062-b245-3b8db47cc76a","refunds":
 [{"actionCode":"000","refNum":"702887093561","authorizationResponseId":
 "164396","refundDate":1702887164000,"amount":1,"reversed":false,
 "externalRefundId":"224c429d-1620-44a0-b87c-0a1cb0161b33"},{"actionCode":
 "000","refNum":"702887093561","authorizationResponseId":"042386",
 "refundDate":1702888042000,"amount":11,"reversed":false,"externalRefundId":
 "224c429d-1620-44a0-b87c-0a1cb0161b36"},{"actionCode":"000","refNum":
```

"702887093561","authorizationResponseId":"656894","refundDate":
1702888656000,"amount":11,"reversed":false,"externalRefundId":
"224c429d-1620-44a0-b87c-0a1cb0161b37"}],"error":false}

{"errorCode":5,"errorMessage":"[externalRefundId] is invalid","error":true}

4.11 Extended order status request

The **getOrderStatusExtended.do** is used to obtain the status of a registered order.

Request parameters

Name	Type	Mandatory
userName	AN 1..100	Yes
Login of the API user on whose behalf requests are processed for a specific merchant.		
password	String	Yes
User’s password.		
orderId	ANS36	Yes*
Unique order number generated by EPG after the registration of the order.		

Note: Either **orderId** or **orderNumber** can be specified is in the request. If the request contains both parameters, **orderId** has priority.		
orderNumber	AN 1..32	Yes*
Number (identifier) of the order in the merchant's online store system. It is unique for every store in the system and is generated on the order registration. If the **Require system to generate order numbers** permission is enabled for the merchant, the order number is generated automatically. Otherwise, the order number data is sent in the API.		
language	A2	No
Language code in the ISO 639-1 format. If it is unspecified, SmartVista E-Commerce Payment Gateway uses the default language from the merchant settings. Error messages are also returned in this language.		
fraudLevel	INT	No
Transaction's level of suspicion. Depending on this value, the transaction can appear to be at risk.		

Response parameters

The set of the response parameters depends on the **getOrderStatusExtended** version that is specified by the **Version of getOrderStatusExtended** parameter on the **Additional information** tab in the merchant settings.

The following parameters are returned in a response for all **getOrderStatusExtended** versions:

Name	Type	Mandatory
orderNumber	AN 1..32	Yes
Number (identifier) of the order in the merchant's online store system. It is unique for every store in the system and is generated on the order registration. If the **Require system to generate order numbers** permission is enabled for the merchant, the order number is generated automatically. Otherwise, the order number data is sent in the API.		
orderStatus	N2	No
Order status in SmartVista E-Commerce Payment Gateway. The possible values of the field are listed in the orderStatus values table below. It is absent if a matching order was not found.		
actionCode	N3	Yes
Processing system authorization code.		
actionCodeDescription	ANS..512	Yes
Action code description in the language specified in the **Language** parameter.		

errorCode	N 1..3	No
Error code.		
errorMessage	AN 1..512	No
Description of the error in the language that was sent in the **language** parameter of the request.		
amount	N 1..12	Yes
Order amount in the minor denomination (for example, cents).		
currency	N3	No
Payment currency code in the ISO 4217 format. If it is not specified, the default value is used.		
date	ANS	Yes
Date of order registration.		
orderDescription	ANS..512	No

Free form description of the order.		
ip	N..	Yes
IP address of the customer who paid for the order.		
merchantOrderParams[]		No
Tag with attributes that contain additional merchant parameters. See the merchantOrderParams block.		
attributes[]		Yes
Attributes of the order in the payments system (order number). See the attributes block.		
cardAuthInfo[]		No
Tag containing the payment attributes. The parameters of the block are detailed in the cardAuthInfo block parameters table below.		
fraudLevel	N..20	No

Assessed fraud level of the payment based on the fraud checks and the payment attributes.		
paymentAccountReference	ANS...999	No
Payment account data. This data can be sent if it is available and this field transfer is configured on the host.		

The following parameters are returned additionally to the parameters described above depending on the request version:

Name	Type	Mandatory	Version
authDateTime	ANS	No	02, 03, 05, 06, 07, 08, 09, 11
Authorization date and time.			
authRefNum	AN..24	No	02, 03, 05, 06, 07, 08, 09, 11
Reference number.			
terminalId	ANS..8	Yes	02, 03, 05, 06, 07, 08, 09, 11

Card Acceptor Terminal Identification. It is a unique ID of the merchant's terminal.			
paymentAmountInfo[]		No	03, 05, 06, 07, 08, 09, 11
Tag containing information about the confirmation amount, debit amount, and refund amount. See the paymentAmountInfo block parameters table below.			
bankInfo[]		No	03, 05,03, 05, 06, 07, 08, 09, 11
Tag containing information about the issuing bank. See the bankInfo block parameters table below.			
chargeback	Boolean	No	06, 07, 08, 09, 11
Specifies whether the transaction is chargeback: · true · false			
paymentWay	AN..32	No	09, 11

· CARD — payment made by entering the card details · CARD_BINDING — payment made using a binding · CARD_MOTO — payment made using the call center · UPOP_MOTO (CUP UPOP MOTO) — payment made China UnionPay Express Pay · UPOP — payment made China UnionPay Secure Pay · FILE_BINDING — payment made using a binding uploaded in a file · P2P — payment made when transferring funds from one card to another · APPLE_PAY — payment made using the Apple Pay service · MASTERPASS — payment made using a Masterpass e-wallet · OTHER — payment used for orders processed outside EPG · GOOGLE_PAY — payment made using the Google Pay service · SAMSUNG_PAY — payment made using the Samsung Pay service · NSPK_E_CERT — payment made using the NSPK Electronic Certificate · FASTPAYMENT_QR — payment made using the FastPayment QR code · MPU — payment via Myanmar Payment Union (MPU) · WAVE — payment via WavePay			
feeAmount	N 1..19	No	11
Fee amount in the minor denomination of the currency.			
transactionData	JWS	No	
See the description of the Transaction Data block below.			

orderStatus values

The **orderStatus** field may have the following statuses:

Code	Description
0	Order registered, but not paid. This is not the final status of the transaction. The order can be paid for again. The intermediary statuses and the history of each attempt can be observed in the **actionCode** and **actionCodeDescription** fields by processing the **getOrderStatusExtended.do** request.
-1	This status replaces the decline status which is returned if none of the specified statuses is suitable for the transaction.
1	Transaction has been approved (for a one-phase payment).
1	Preauthorization amount was put on hold (for a two-phase payment).
2	Amount was deposited successfully.
3	Transaction has been reversed. This represents the transaction final status that cannot be modified.
4	Transaction has been refunded.

6	Transaction was declined. This represents the transaction final status that cannot be modified.
7	Card was added.
8	Card was updated.
9	Card was verified.
10	Recurring template was added.

cardAuthInfo block parameters

Name	Type	Mandatory
pan	N 13...19	No
Masked number of the card that has been used for the payment. It is only specified for paid orders.		
expiration	N6	No
Card expiration date in the following format: *YYYYMM*. It is only specified for paid orders.		

cardholderName	ANS 2..26	No
Name of the cardholder. This parameter is verified according to the following criteria: · Acceptable characters are: Latin letters, 0-9, $,), (, -, . , a space · Cardholder name must start with a letter · Minimum length: 2 characters · Maximum length: 26 characters · Null is valid · Uppercase and lowercase are acceptable		
authorizationResponseId (the deprecated name is **approvalCode**)	AN6	No
Payments network authorization code. The field has a fixed length of six characters; it can contain both numbers and letters.		
secureAuthInfo[]	Not applicable	No
Tag containing information about secure authentication. The parameters of the secureAuthInfo[] block are detailed below.		
authenticationIndicator	String	No

3DS authentication indicator that specifies the type of 3DS authentication used for the transaction.

Possible values:

- THREEDS_1_Y — SCA Cardholder authentication with 3DS 1.x
- THREEDS_1_A — Cardholder authentication attempt with 3DS 1.x
- THREEDS_2_Y — SCA Cardholder authentication with 3DS 2.x
- THREEDS_2_F — RBA Cardholder authentication with 3DS 2.x
- THREEDS_2_A — Cardholder authentication attempt with 3DS 2.x

Note: This parameter is only applicable for successfully 3DS authenticated transactions, and is used for **Merchant.OrderStatusExtendedVersion.VERSION_13** or a higher version.

secureAuthInfo[] block parameters

Note: To include the **secureAuthInfo** block in the method response, enable the **Receive 3DS requisites of transactions** option for the merchant.

Name	Type	Mandatory
eci	AN2	No
Electronic Commerce Indicator.		
cavv	ANS..200	No
Cardholder Authentication Verification Value.		

xid	ANS..80	No
Electronic Commerce Transaction Identifier.		
transStatus	String	No
Value of **ARes.transStatus** (for frictionless authentication) or **RReq.transStatus** (for challenge authentication). For possible values, see the EMVCo specification. **Note:** This parameter is used for **Merchant.OrderStatusExtendedVersion.VERSION_13** or a higher version.		
transStatusReason	String	No
Value of **ARes.transStatusReason** (for frictionless authentication) or **RReq.transStatusReason** (for challenge authentication). For possible values, see the EMVCo specification. **Note:** This parameter is used for **Merchant.OrderStatusExtendedVersion.VERSION_13** or higher version.		

The bindingInfo block parameters

Name	Type	Mandatory
clientId	ANS 1..255	No

Identifier of the customer in the merchant's system. It is used to implement the binding functional. It may be present, if the merchant has the permissions to create and manage bindings (the **Merchant is allowed to use bindings** merchant option and other.		
bindingId	ANS..255	No
Identifier of the binding created when paying this order or created earlier and used to pay for this order. It is present only if the merchant has the permission to create bindings.		

The paymentAmountInfo block parameters

Name	Type	Mandatory
paymentState	N..9	No
Payment status.		
approvedAmount	N 1..19	No
Amount confirmed to be debited.		
depositedAmount	N 1..19	No
Amount confirmed debited from the card.		

refundedAmount	N 1..19	No
Refund amount.		

The bankInfo block parameters

Name	Type	Mandatory
bankName	AN..200	No
Name of the issuing bank.		
bankCountryCode	AN..4	No
Code of the issuing bank country.		
bankCountryName	AN..160	No
Name of the country of the issuing bank passed in the language parameter of the request or in the language of the user who has called the method if the language has not been specified in the request.		

Parameters of the merchantOrderParams block

Name	Type	Mandatory

name	AN..20	Yes
Name of the additional merchant parameter.		
value	AN..1024	Yes
Value of the additional merchant parameter.		

Parameters of the attributes block

Name	Type	Mandatory
name	AN7	Yes
Attribute name is **mdOrder**.		
value	ANS36	Yes
Order number in the payments system (it is unique in the system).		

Transaction Data block

A single signed data block is used to transfer all original transaction data fields to and from a PCI DSS merchant, instead of multiple separate fields.

An RSA key pair with **CertificateType.TRANSACTION_DATA_SIGNATURE** must be generated for the merchant.

The block includes the following:

Header

Field	Description
alg	Algorithm (RS256).
kid	ID of the corresponding record in the key_mgmt_certificate table.
typ	Header type (JOSE).
merchantLogin	Login of the merchant.

Payload

Field	Description
mdOrder	This field is reserved for future use.
transDate	This field is reserved for future use.
networkReferenceData	Network reference data returned from the authorization host.

transactionDataElements	Data elements of the authorization request.
isSsl	Specifies whether 3DS2 authentication was not performed for the transaction.
dsTransId	Directory Server Transaction ID, UUID (only for 3DS2 transactions).
acsTransId	Transaction ID in ACS (only for 3DS2 transactions).
initTransactionDate	Initiating transaction date (only for 3DS2 transactions).
version	Constant value of 1. This field is reserved for future use.
panSalt	Base64-encoded 16 bytes random salt (the resulting string will be approximately 24 character long; from **SecureRandom**).
panHash	Base64 SHA-256 hash of byte[] = PAN.getBytes + salt.bytes.

The block validation fails if:

- The field is missing for MIT requests with **recurringPaymentManagedByMerchant=**true or **bindingManagedByMerchant=**true.
- A different **merchantLogin** is used.
- The key provided in the JWS header cannot be found for the current merchant.
- Signature validation fails (both technical and not errors).

Error codes (errorCode) and messages (errorMessage)

Code	Message	Description
0	No system error.	No errors.
1	Expected **orderId** or **orderNumber**	Neither order ID nor order number was specified in the request.
2	Order is declined due to an error in the payment credential.	The bank or payment processor is refusing to authorize the transaction.
5	Access is denied.	The user does not have the necessary permissions to access this resource.
5	The user who processes payments for the merchant must change their password.	The user needs to update their password for security reasons.
6	Unregistered **orderId**.	The order with the specified **OrderId** is not registered in the system.
7	System error.	Software or hardware issue or malfunction.

Request example

```
http://<host:port>/payment/rest/getOrderStatusExtended.do?
 userName=apiuser&password=apiuserpassword&orderId=
 9249a590-f528-4299-af01-7c4c92319213
```

Response examples

Response example v.1


```
{
    "errorCode": "0",
    "errorMessage": "Success",
    "orderNumber": "20220429115021730",
    "orderStatus": 6,
    "actionCode": -2018,
    "actionCodeDescription": "Declined. DS connection timeout",
            "amount": 6050,
    "currency": "840",
            "date": 1651222231076,
            "ip": "0:0:0:0:0:0:0:1",
    "merchantOrderParams": [
            {
           "name": "suppressShippingAddress",
           "value": "true"
            },
            {
           "name": "disablePhone",
           "value": "true"
            },
            {
           "name": "cartId",
           "value": "20220429115021730"
            },
            {
           "name": "disableEmail",
```

```
            "value": "true"
             }
             ],
    "attributes": [
             {
                 "name": "mdOrder",
            "value": "ae8b70f5-4e27-4777-9718-e79d1bae9006"
             }
             ],
    "cardAuthInfo": {
        "expiration": "202404",
        "cardholderName": "FEFWE EWEFEW",
        "secureAuthInfo": {
            "eci": 7
             },
        "pan": "400001**0001"
             },
    "fraudLevel": 0
}
```


Response example v.4


```
{
    "errorCode": "0",
    "errorMessage": "Success",
    "orderNumber": "2022042511542217",
    "orderStatus": 2,
    "actionCode": 0,
    "actionCodeDescription": "Request processed successfully",
             "amount": 6050,
    "currency": "840",
             "date": 1650876845158,
             "ip": "0:0:0:0:0:0:0:1",
    "merchantOrderParams": [
             {
```

```
            "name": "suppressShippingAddress",
            "value": "true"
             },
              {
            "name": "disablePhone",
            "value": "true"
             },
             {
            "name": "cartId",
            "value": "2022042511542217"
             },
             {
            "name": "disableEmail",
            "value": "true"
             }
    ],
    "attributes": [
             {
            "name": "mdOrder",
            "value": "61fe654a-f2ff-45f1-8f58-ec15d2f97137"
             }
             ],
    "cardAuthInfo": {
        "expiration": "202404",
        "cardholderName": "SDQW QWDQWDQW",
             "authorizationResponseId": "867186",
        "secureAuthInfo": {
            "eci": 5,
            "threeDSInfo": {
                "xid": "NjU4NDIwNTE2NTA4NzY4NjE2NDk=",
                "cavv": "AAABASMABAAAAAAAAAAEAAAAAAA="
             }
             },
        "pan": "400001**0001"
             },
    "fraudLevel": 0,
    "paymentAccountReference": "1234567890"
}
```

Response example v.5


```
{
    "errorCode": "0",
    "errorMessage": "Success",
    "orderNumber": "2022042511542217",
    "orderStatus": 2,
    "actionCode": 0,
            "actionCodeDescription": "Request processed successfully",
            "amount": 6050,
    "currency": "840",
            "date": 1650876845158,
            "ip": "0:0:0:0:0:0:0:1",
    "merchantOrderParams": [
            {
           "name": "suppressShippingAddress",
            "value": "true"
            },
            {
           "name": "disablePhone",
           "value": "true"
            },
            {
           "name": "cartId",
           "value": "2022042511542217"
            },
            {
           "name": "disableEmail",
             "value": "true"
            }
            ],
    "attributes": [
            {
           "name": "mdOrder",
           "value": "61fe654a-f2ff-45f1-8f58-ec15d2f97137"
            }
            ],
    "cardAuthInfo": {
```

```
        "expiration": "202404",
        "cardholderName": "SDQW QWDQWDQW",
        "authorizationResponseId": "867186",
        "secureAuthInfo": {
           "eci": 5,
           "threeDSInfo": {
              "xid": "NjU4NDIwNTE2NTA4NzY4NjE2NDk=",
              "cavv": "AAABASMABAAAAAAAAAAEAAAAAAA="
              }
            },
        "pan": "400001**0001"
            },
    "authDateTime": 1650876867059,
    "terminalId": "20000001",
    "authRefNum": "650876867186",
    "paymentAmountInfo": {
        "paymentState": "DEPOSITED",
        "approvedAmount": 6050,
             "depositedAmount": 6050,
        "refundedAmount": 0
            },
    "bankInfo": {
        "bankCountryName": "<Unknown>"
            },
    "fraudLevel": 0,
    "paymentAccountReference": "1234567890"
}
```


Response example v.8


```
{
    "errorCode": "0",
    "errorMessage": "Success",
    "orderNumber": "2022042511542217",
    "orderStatus": 2,
```

"actionCode": 0,
"actionCodeDescription": "Request processed successfully",
"amount": 6050,
"currency": "840",
"date": 1650876845158,
"ip": "0:0:0:0:0:0:0:1",
"merchantOrderParams": [
{
"name": "suppressShippingAddress",
"value": "true"
},
{
"name": "disablePhone",
"value": "true"
},
{
"name": "cartId",
"value": "2022042511542217"
},
{
"name": "disableEmail",
"value": "true"
}
],
"attributes": [
{
"name": "mdOrder",
"value": "61fe654a-f2ff-45f1-8f58-ec15d2f97137"
}
],
"cardAuthInfo": {
"expiration": "202404",
"cardholderName": "SDQW QWDQWDQW",
"authorizationResponseId": "867186",
"paymentSystem": "VISA",
"secureAuthInfo": {
"eci": 5,
"threeDSInfo": {

```
            "xid": "NjU4NDIwNTE2NTA4NzY4NjE2NDk=",
            "cavv": "AAABASMABAAAAAAAAAAEAAAAAAA="
          }
          },
      "pan": "400001**0001"
          },
   "authDateTime": 1650876867059,
   "terminalId": "20000001",
          "authRefNum": "650876867186",
   "paymentAmountInfo": {
      "paymentState": "DEPOSITED",
      "approvedAmount": 6050,
      "depositedAmount": 6050,
      "refundedAmount": 0
          },
   "bankInfo": {
      "bankCountryName": "<Unknown>"
          },
   "chargeback": false,
   "fraudLevel": 0,
   "paymentAccountReference": "1234567890"
}
```


Response example v.9


```
{
   "errorCode": "0",
   "errorMessage": "Success",
   "orderNumber": "2022042511542217",
   "orderStatus": 2,
   "actionCode": 0,
          "actionCodeDescription": "Request processed successfully",
          "amount": 6050,
   "currency": "840",
          "date": 1650876845158,
          "ip": "0:0:0:0:0:0:0:1",
```

```
"merchantOrderParams": [
        {
       "name": "suppressShippingAddress",
        "value": "true"
        },
        {
       "name": "disablePhone",
       "value": "true"
        },
        {
       "name": "cartId",
       "value": "2022042511542217"
        },
        {
       "name": "disableEmail",
            "value": "true"
        }
        ],
"attributes": [
        {
       "name": "mdOrder",
       "value": "61fe654a-f2ff-45f1-8f58-ec15d2f97137"
        }
        ],
"cardAuthInfo": {
   "expiration": "202404",
   "cardholderName": "SDQW QWDQWDQW",
   "authorizationResponseId": "867186",
   "paymentSystem": "VISA",
   "secureAuthInfo": {
      "eci": 5,
      "threeDSInfo": {
         "xid": "NjU4NDIwNTE2NTA4NzY4NjE2NDk=",
         "cavv": "AAABASMABAAAAAAAAAAEAAAAAAA="
        }
        },
   "pan": "400001**0001"
        },
```

```
    "authDateTime": 1650876867059,
    "terminalId": "20000001",
    "authRefNum": "650876867186",
    "paymentAmountInfo": {
        "paymentState": "DEPOSITED",
        "approvedAmount": 6050,
        "depositedAmount": 6050,
        "refundedAmount": 0
            },
    "bankInfo": {
        "bankCountryName": "<Unknown>"
            },
    "chargeback": false,
    "paymentWay": "CARD",
    "fraudLevel": 0,
    "paymentAccountReference": "1234567890"
}
```


Response example v.11


```
{
    "errorCode": "0",
    "errorMessage": "Success",
    "orderNumber": "2022042511542217",
    "orderStatus": 2,
    "actionCode": 0,
    "actionCodeDescription": "Request processed successfully",
            "amount": 6050,
    "currency": "840",
            "date": 1650876845158,
            "ip": "0:0:0:0:0:0:0:1",
    "merchantOrderParams": [
            {
           "name": "suppressShippingAddress",
           "value": "true"
```

```
        },
        {
       "name": "disablePhone",
       "value": "true"
        },
        {
       "name": "cartId",
       "value": "2022042511542217"
        },
        {
       "name": "disableEmail",
       "value": "true"
        }
        ],
"attributes": [
        {
       "name": "mdOrder",
       "value": "61fe654a-f2ff-45f1-8f58-ec15d2f97137"
        }
        ],
"cardAuthInfo": {
   "expiration": "202404",
   "cardholderName": "SDQW QWDQWDQW",
   "authorizationResponseId": "867186",
       "paymentSystem": "VISA",
   "secureAuthInfo": {
      "eci": 5,
      "threeDSInfo": {
         "xid": "NjU4NDIwNTE2NTA4NzY4NjE2NDk=",
         "cavv": "AAABASMABAAAAAAAAAAEAAAAAAA="
        }
        },
   "pan": "400001**0001"
        },
"authDateTime": 1650876867059,
"terminalId": "20000001",
"authRefNum": "650876867186",
"paymentAmountInfo": {
```

```
        "paymentState": "DEPOSITED",
        "approvedAmount": 6050,
        "depositedAmount": 6050,
            "refundedAmount": 0
            },
    "bankInfo": {
        "bankCountryName": "<Unknown>"
            },
    "chargeback": false,
    "paymentWay": "CARD",
    "feeAmount": 0,
    "fraudLevel": 0,
    "paymentAccountReference": "1234567890"
}
```


Response v.13 – Full 3DS2 response


```
{
    "errorCode": "0",
    "errorMessage": "Success",
    "orderNumber": "postman__1691072312072",
    "orderStatus": 2,
    "actionCode": 0,
    "actionCodeDescription": "Request processed successfully",
            ...
    "cardAuthInfo": {
            "expiration": "204912",
        "paymentSystem": "MIR",
        "secureAuthInfo": {
            "eci": "02",
            "threeDSInfo": {
                "xid": "299e1978-2464-46e6-b768-e998454dfb83",
                "cavv": "AAIAAAAAAAAAAAAAADKFIiIiIiI=",
                "transStatus": "Y"
            },
            "authenticationIndicator": "THREEDS_2_Y"
```

```
            },
        "approvalCode": "334674",
        "pan": "220100**0011"
            },
...
            }
```

Response v.13 – Authentication failed in ACS (max challenges exceeded)

```
{

    "errorCode": "0",
    "errorMessage": "Success",
    "orderNumber": "postman__1691072221558",
    "orderStatus": 6,
    "actionCode": 341024,
    "actionCodeDescription": "processing.error.341024",

...

    "cardAuthInfo": {
            "expiration": "204912",
        "paymentSystem": "MIR",
        "secureAuthInfo": {
            "threeDSInfo": {
                "xid": "092cac46-721c-4bab-aff3-2328f19e5f34",
                "transStatus": "N",
                "transStatusReason": "19"
              }
            },

        "pan": "220100**0011"
            },
...

}
```

Declined payment


```
{
   "errorCode": "0",
   "errorMessage": "Success",
   "actionCode": 100239,
   "actionCodeDescription": "Terminate Subscription",
     "paymentAmountInfo": {"paymentState": "DECLINED"...

 }
```


4.12 Request for payments statistics for a period

The **getLastOrdersForMerchants.do** method is used to get the payment statistics for a specified period. The method returns a list of order descriptions, where each description represents an extended order status that can be received by executing the **getOrderStatusExtended.do** request (see Extended order status request) for a specific order.

Request parameters

Name	Type	Mandatory
userName	AN 1..100	Yes
Login of the API user on whose behalf requests are processed for a specific merchant.		
password	String	Yes

User’s password.		
language	A2	No
Language code in the ISO 639-1 format. If it is unspecified, SmartVista E-Commerce Payment Gateway uses the default language from the merchant settings. Error messages are also returned in this language.		
page	N	No
When processing the request, a list is formed that is broken down into pages (with the number of records on each page of the same size). A page with the number that was specified in the page parameter is returned in the response. The pages numbering starts from 0. If the parameter is not specified, a page with the number 0 is returned.		
size	N..3	Yes
Number of elements on a page. The maximum value is 200.		
from		Yes
Date and time when the period of collecting orders starts, in the following format *YYYYMMDDHHmm*.		

to		Yes
Date and time when the period of collecting orders ends, in the following format *YYYYMMDDHHmmss*.		
transactionStates[]	A..9	Yes
Required order statuses. Only orders in one of the specified statuses are included in the report. If there are several values, they are separated by commas. The following values are available: · CREATED · APPROVED · DEPOSITED · DECLINED · REVERSED · REFUNDED		
merchants[]	ANS	Yes
List of logins of the merchant whose transactions are included in the report. If there are several values, they are separated by commas. Leave this field empty to get the list of reports on all the available merchants (child merchants and merchants specified in the settings of the user).		
searchByCreatedDate	Boolean	No

Specifies whether the creation date is used to search for orders:

- true — a search for orders that have the creation date that falls into the specified period.
- false — a search for orders that have the payment date that falls into the specified period (thus, orders in the CREATED or DECLINED status cannot be present in the report). This is the default value.

Response parameters

Name	Type	Mandatory
errorCode	N 1..3	No
Error code.		
errorMessage	AN 1..512	No
Description of the error in the language that was sent in the **language** parameter of the request.		
orderStatuses[]	Not applicable	
block that contains information about the orders included in the report. See the orderStatuses block parameters table below.		

totalCount	N	Yes
Total number of elements in the report (on all pages).		
page	N	Yes
Number of the current page (it is the same as the page number passed in the request).		
pageSize	N..3	Yes
Maximum number of records on a page (it is the same as the page number passed in the request).		

Parameters of the orderStatuses block

Name	Type	Mandatory
orderNumber	AN 1..32	Yes
Number (identifier) of the order in the merchant's online store system. It is unique for every store in the system and is generated on the order registration. If the **Require system to generate order numbers** permission is enabled for the merchant, the order number is generated automatically. Otherwise, the order number data is sent in the API.		

orderStatus	N..2	Yes
Status of the order in the payments system. The available values are provided in the table with the orderStatus values.		
actionCode	N..3	Yes
Response code.		
actionCodeDescription	ANS..512	Yes
Meaning of the response code.		
amount	N 1..12	Yes
Payment amount in the minor denomination of the currency.		
currency	N3	Yes
ISO 4217 code of the payment currency. If it is not specified, it is considered to be the same as the default currency value.		
date	ANS	Yes

Order registration date.		
orderDescription	AN..512	No
Order description passed on its registration.		
ip	AN..20	No
IP address of the buyer. It is specified only after a payment.		
errorCode	N 1..3	Yes
Error code.		
merchantOrderParams[]	Not applicable	No
Tag containing attributes that contain additional merchant parameters. See the merchantOrderParams block parameters table below.		
attributes[]	Not applicable	Yes
Attributes of the order in the payments system (order number). See the attributes block parameters table below.		

cardAuthInfo[]	Not applicable	No
Tag containing the payment attributes. See the cardAuthInfo block parameters table below.		
p2pData[]	Not applicable	No
Details about the source and recipient cards, in the case of a card-to-card transfer. See the p2pData block parameters table below.		
bindingInfo[]	Not applicable	No
Tag containing information about the binding with which the payment is performed. See the bindingInfo block parameters table below.		
authDateTime	ANS	No
Authorization date and time.		
terminalId	ANS..8	Yes
Card Acceptor Terminal Identification. It is a unique ID of the merchant's terminal.		

authRefNum	AN..24	No
Reference number.		
paymentAmountInfo[]	Not applicable	No
Tag containing information about the confirmation amount, debit amount, and refund amount. See the paymentAmountInfo block parameters table below.		
bankInfo[]	Not applicable	No
Tag containing information about the issuing bank. See the bankInfo block parameters table below.		

Parameters of the merchantOrderParams block

Name	Type	Mandatory
name	AN..20	Yes
Name of the additional merchant parameter.		
value	AN..1024	Yes
Value of the additional merchant parameter.		

Parameters of the attributes block

Name	Type	Mandatory
name	AN7	Yes
Attribute name is **mdOrder**.		
value	ANS36	Yes
Order number in the payments system (it is unique in the system).		

Parameters of the cardAuthInfo block

Name	Type	Mandatory
pan	N 13...19	No
Masked number of the card that has been used for the payment. It is only specified for paid orders.		
expiration	N6	No
Card expiration date in the following format: *YYYYMM*. It is only specified for paid orders.		

cardholderName	ANS 2..26	No
Name of the cardholder. This parameter is verified according to the following criteria: · Acceptable characters are: Latin letters, 0-9, $,), (, -, . , a space · Cardholder name must start with a letter · Minimum length: 2 characters · Maximum length: 26 characters · Null is valid · Uppercase and lowercase are acceptable		
authorizationResponseId (the deprecated name is **approvalCode**)	AN6	No
Payments network authorization code. The field has a fixed length of six characters; it can contain both numbers and letters.		
secureAuthInfo[]	Not applicable	No
Tag containing information about secure authentication. The parameters of the secureAuthInfo[] block are detailed below.		
authenticationIndicator	String	No

3DS authentication indicator that specifies the type of 3DS authentication used for the transaction.

Possible values:

- THREEDS_1_Y — SCA Cardholder authentication with 3DS 1.x
- THREEDS_1_A — Cardholder authentication attempt with 3DS 1.x
- THREEDS_2_Y — SCA Cardholder authentication with 3DS 2.x
- THREEDS_2_F — RBA Cardholder authentication with 3DS 2.x
- THREEDS_2_A — Cardholder authentication attempt with 3DS 2.x

Note: This parameter is only applicable for successfully 3DS authenticated transactions, and is used for **Merchant.OrderStatusExtendedVersion.VERSION_13** or a higher version.

secureAuthInfo[] block parameters

Note: To include the **secureAuthInfo** block in the method response, enable the **Receive 3DS requisites of transactions** option for the merchant.

Name	Type	Mandatory
eci	AN2	No
Electronic Commerce Indicator.		
cavv	ANS..200	No
Cardholder Authentication Verification Value.		

xid	ANS..80	No
Electronic Commerce Transaction Identifier.		
transStatus	String	No
Value of **ARes.transStatus** (for frictionless authentication) or **RReq.transStatus** (for challenge authentication). For possible values, see the EMVCo specification.		
transStatusReason	String	No
Value of **ARes.transStatusReason** (for frictionless authentication) or **RReq.transStatusReason** (for challenge authentication). For possible values, see the EMVCo specification.		

Parameters of the p2pData block

Name	Type	Mandatory
debitPan	N 13...19	No
Masked number of a card that is the source of the funds to transfer.		
creditPan	N 13...19	No

Masked number of a card that is the recipient of the funds.

Parameters of the bindingInfo block

Name	Type	Mandatory
clientId	ANS 1..255	No
Identifier of the customer in the merchant's system. It is used to implement the binding functional. It may be present, if the merchant has the permissions to create and manage bindings (the **Merchant is allowed to use bindings** merchant option and other.		
bindingId	ANS..255	No
Identifier of the binding created when paying this order or created earlier and used to pay for this order. It is present only if the merchant has the permission to create bindings.		

Parameters of the paymentAmountInfo block

Name	Type	Mandatory
paymentState	N..9	No
Payment status.		

approvedAmount	N 1..19	No
Amount confirmed to be debited.		
depositedAmount	N 1..19	No
Amount confirmed debited from the card.		
refundedAmount	N 1..19	No
Refund amount.		

Parameters of the bankInfo block

Name	Type	Mandatory
bankName	AN..200	No
Name of the issuing bank.		
bankCountryCode	AN..4	No
Code of the issuing bank country.		

bankCountryName	AN..160	No
Name of the country of the issuing bank passed in the language parameter of the request or in the language of the user who has called the method if the language has not been specified in the request.		

Values of the orderStatus field

Value	Description
0	The order has been registered but not paid.
1	The preauthorized amount has been put on hold (for two-phase payments).
2	Full authorization of the order amount has been performed.
3	Authorization is canceled.
4	A refund operation has been processed for the transaction.
5	Authorization through ACS of the issuing bank has been initiated.
6	Authorization is declined.

Error codes (errorCode) and messages (errorMessage)

Code	Message	Description
0	Success	The request has been processed successfully.
5	One of the mandatory fields is not filled in.	The value of one of the mandatory parameters is not specified.
	Incorrect format of the **transactionStates** parameter.	The format of the transactionStates parameter value is incorrect.
	Access is denied.	The user does not have the necessary permissions to access this resource.
7	System error.	Software or hardware issue or malfunction.
10	Value of the size parameter exceeds the maximum allowed value.	The **size** parameter value exceeds 200.
	Insufficient permissions to view transactions for the specified merchant.	The user does not have the necessary access to see the specified transactions.

Request example

```
http://<host:port>/payment /rest/getLastOrdersForMerchants.do?
 userName=apiuser&password=apiuserpassword&page=0&size=100&from=
 20180809160000&to=20180909160000&transactionStates=DEPOSITED,
 REVERSED&merchants=Merchant&searchByCreatedDate=false
```

Response example


```
{"errorCode":0,"orderStatuses":[{"errorCode":"0","orderNumber":"20180810183020704",
 "orderStatus":2,"actionCode":0,"actionCodeDescription":"","amount":5670,
 "currency":"840","date":1533915047250,"ip":"10.0.1.183","merchantOrderParams":[],
 "attributes":[{"name":"mdOrder","value":"c1c5d65d-212b-4f5f-a53f-fc2fd9fc3d52"}],
 "cardAuthInfo":{"expiration":"202006","cardholderName":"IVAN","approvalCode":
 "123456","pan":"400001****0050"},"bindingInfo":{"clientId":"888","bindingId":
 ...
 "value":"7a835003-c8b8-487c-8450-28824990dce6"}],"cardAuthInfo":
 {"expiration":"202008","cardholderName":"IVAN","approvalCode":
 "123456","pan":"500001****0032"},"bindingInfo":{"clientId":"888","bindingId":
 "7481ac1c-3074-46e8-8f61-dffb802f5227"}}],"totalCount":6,"page":0,"pageSize":100}
```


4.13 Fee calculation request

The **fee/calculate.do** and **fee/public/calculate.do** methods are used to calculate a fee for a transaction and to return the fee.

System settings

On the **Administration > System settings** page of the EPG administration portal the following setting must be specified:

fee.service.enabled	
Description	Enables the fee service that is used to calculate a fee.

Merchant options

To be able to execute this method, the merchant must have the Fee calculation allowed permission enabled.

Request parameters

Name	Type	Mandatory
mdOrder	ANS36	No
Order number generated by EPG after the registration of the order.		
$PAN	N 13...19	Yes
Card number.		
expiry	N6	Yes

Expiration date of the card that is used for fee calculation. It is required for transactions if the binding (COF) is not used.		
bindingId	AN..255	No
Identifier of the binding that was created earlier (see Managing bindings). It can only be used if the merchant has the permission to work with bindings. If this parameter is sent in the **registerOrder** request, the following is executed: · This order can only be paid by binding. · The payer is redirected to a payment page where the CVC must be entered. It is required for transactions if the binding (COF) is not used.		
userName	AN 1..100	Conditional
Login of the API user on whose behalf requests are processed for a specific merchant. **Note:** It is specified only for **fee/public/calculate.do**.		
password	String	Conditional
User's password. **Note:** It is specified only for **fee/public/calculate.do**.		
language	A2	Conditional

Language in the ISO 639-1 format. Error messages are sent in this language. If it is not specified, the default language specified in the merchant settings is used. **Note:** It is specified only for **fee/public/calculate.do**.

Response parameters

Name	Type	Mandatory
status	AN..32	Yes

Fee calculation status:
 	
· SUCCESS — fee is calculated.
· ERROR — error occurred.
· UNAVAILABLE — fee cannot be calculated for the transaction. This status is returned if:
  ○ Fee calculation is not enabled in the system settings (**fee.service.enabled = false**).
  ○ The **FEE_CALCULATION_ALLOWED** option is not enabled for the merchant.
  ○ The external fee is transferred in **register.do** and the External fee allowed permission is enabled.
 	
Note: If the ERROR or UNAVAILABLE status is returned, other fields are not present in the response.		

feeAmountFormatted	NS1..13	Yes
Formatted fee amount.		

totalAmountFormatted	NS1..13	Yes
Formatted total transaction amount.		

Request example

fee/calculate.do

http://<host:port>/epg/fee/calculate.do?

```
{
   "$CVC": "472",
   "$PAN": "5000030000000337"
   "expiry": "202512",
   "MDORDER": "3da48fb9-37ce-4458-8766-954dd18c287a",
}
```

fee/public/calculate.do

http://<host:port>/epg/fee/public/calculate.do?

```
{
   "$PAN": "5000030000000337"
   "expiry": "202512",
   "MDORDER": "3da48fb9-37ce-4458-8766-954dd18c287a",
   "language": "en",
   "userName": "apiuser",
   "password": "apiuserpassword",
}
```

Response example

```
{"status": "SUCCESS",
   "feeAmountFormatted": "6.05",  "totalAmountFormatted": "66.55"}
```

4.14 Processing payments with bindings

The **paymentOrderBinding.do** method is used to perform a payment using a binding. For more information about bindings, see Managing binding.

Request parameters

Name	Type	Mandatory
userName	AN 1..100	Yes
API user login.		
password	String	Yes
API user password.		
mdOrder	ANS36	Yes
Unique order number generated by EPG after the registration of the order.		
bindingId	AN..255	Yes
Identifier of the binding that was created earlier (see Managing bindings). It can only be used if the merchant has the permission to work with bindings. If this parameter is sent in the **registerOrder** request, the following is executed: · This order can only be paid by binding.		

· The payer is redirected to a payment page where the CVC must be entered.		
cvc	N3..4	Conditional
Card security code: CVV2 (Card Verification Value 2), CVC2 (Card Verification Code 2) or a similar code. The **cvc** parameter is optional for the merchant if the **Can pay by binding without CVV2/CVC2** permission is enabled for this merchant. If the **Can pay by binding without CVV2/CVC2** permission is not enabled, the **cvc** parameter is mandatory for the merchant.		
language	A2	No
Language in the ISO 639-1 format. Error messages are sent in this language. If it is not specified, the default language specified in the merchant settings is used.		
clientBrowserInfo	Not applicable	No
JSON structure containing the client browser information for 3DS2. See the description of the clientBrowserInfo contents.		
noShowPayment	Boolean	No
No-show is a fee billed by a merchant in accordance with the merchant's service policy when the cardholder fails to cancel a reservation within the time frame disclosed at the time of the booking. If the parameter value is true, the payment is marked as No Show. It is used for one-phase payments.		

Response parameters

Name	Type	Mandatory
redirect	AN..512	No
URL to which the customer is redirected after executing the payment, depending on the payment result.		
info	AN..512	Yes
Result of the payment attempt: · Your order is proceeded, redirecting... · Operation declined. Please check the data and available balance of the card. Redirecting... · Sorry, payment cannot be completed. Redirecting... · Payment declined. Please, contact the merchant. Redirecting... · Payment declined. Please, contact your bank. Redirecting... · Cannot connect to your bank. Please, contact your bank. Redirecting... · Processing timeout. Please, try again later. Redirecting...		
errorCode	N 1..3	No
Response code: · 0 — a successful transaction · Any other number — an error occurred when processing the request		

orderStatusCode	N2	No
Numeric code that specifies the order status in SmartVista E-Commerce Payment Gateway. It is absent if a matching order was not found. The possible values of the field are listed in the **orderStatusCode values** table below.		
orderStatusName	AN 1..100	No
Order processing status: · started — the order was created in EPG. This is the initial order status. · payment_approved — the payment was authorized. · payment_declined — the payment was declined. · payment_void — the payment was reversed. · payment_deposited — the payment was deposited. · refunded — the payment refund was performed. · card_added — the binding was created (for binding register orders only). · card_modified — the binding was modified (for binding modify orders only). · card_verified — the card was verified (for card verification orders only). · recurring_template_added — the recurring payment template was added (for Create recurring payment template orders only).		

Error codes (errorCode) and messages (errorMessage)

Code	Message	Description

0	No system error.	No errors.
1	A required field is absent or invalid.	A mandatory parameter is not available in the request or its value is incorrect.
2	No order found.	The order meeting the specified criteria was not found.
	Binding not found.	The binding matching the specified criteria was not found.
5	Access is denied.	The user does not have the necessary permissions to access this resource.
	User must change the password.	The user needs to update their password for security reasons.
	ClientId is empty or the merchant does not have a permission to execute the method.	The customer ID in the merchant system is not specified or the **Merchant is allowed to use bindings** merchant option is not enabled.
7	System error.	Software or hardware issue or malfunction.

OrderStatusCode values

The **OrderStatusCode** field may have the following values:

Code	Description
0	Order registered, but not paid. This is not the final status of the transaction. The order can be paid for again. The intermediary statuses and the history of each attempt can be observed in the **actionCode** and **actionCodeDescription** fields by processing the **getOrderStatusExtended.do** request.
-1	This status replaces the decline status which is returned if none of the specified statuses is suitable for the transaction.
1	Transaction has been approved (for a one-phase payment).
1	Preauthorization amount was put on hold (for a two-phase payment).
2	Amount was deposited successfully.
3	Transaction has been reversed. This represents the transaction final status that cannot be modified.

4	Transaction has been refunded.
6	Transaction was declined. This represents the transaction final status that cannot be modified.
7	Card was added.
8	Card was updated.
9	Card was verified.
10	Recurring template was added.

Request example

```
http://<host:port>/payment/rest/paymentOrderBinding.do?userName=
 apiuser&password=apiuserpassword&mdOrder=
 7911c014-6ad3-478c-b8cf-b5d5bce3eacd&bindingId=
 7ced41cd-521f-41a7-9d1e-8b6ffebec9e4
```

Response example


```
{"redirect":"https://msk-ecom-wls02.bpc.in:443/epg/merchants/root/finish.html?orderId=
  ffd85ad3-1851-430f-99da-a78ecd228d4d&lang=en&status=payment_deposited",
  "info":"Your order is proceeded, redirecting...","errorCode":0,"orderStatusCode":2,
  "orderStatusName":"payment_deposited"}
```

4.15 Managing bindings

In the context of the SmartVista EPG system, the term *binding* refers to a proprietary Card-on-File (CoF) functionality that enables merchants to securely store and reuse card credentials for future transactions.

Note: For more information about binding processing by PCI DSS certified merchants that, see Card-on-File (CoF) transactions for PCI DSS merchants.

A binding links together in a merchant system the identifier of a customer of the merchant (**clientId**) and the number of the customer card (**PAN**). A binding is created on a successful payment for goods or services (a payment without errors, with the response containing `"errorCode":"0","errorMessage":"Success"`).

If a binding is deleted by the customer, it becomes inactive in the EPG system.

The following EPG database tables are used for the binding:

- BINDING — stores bindings
- BINDING_HISTORY — stores all binding operations (create and update)

Bindings can be used only if a merchant has the permission to create and manage bindings.

The **bindingId** parameter can be passed to EPG during registration of an order. In that case, the order can be paid only with this binding.

Bindings is a proprietary CoF functionality.

Merchant options

The following permissions are mandatory for a merchant to manage bindings:

- Merchant is allowed to use bindings
- Can create and update bindings without payment

To create links for bindings, merchants must have at least one of the following permissions enabled:

- Can send a link to the payment/binding form via e-mail
- Can send a link to the payment/binding form via sms

For more information about merchant options.

User permissions

The user must have the following permission enabled on the **Administration > Users** page:

Permission	Action type	Level
Can create order for binding/update card by the link	API	Regular user
Create a link for the card update. This permission enables users to create an order and send the link to the binding web page to cardholders.		

4.15.1 Creating bindings without payments

The **createBindingNoPayment.do** method is used to create a binding without a payment. It deactivates other bindings for the same merchant, clientID, and PAN.

Note: To be able to execute this method, a merchant must have the **Can create bindings without payment** permission enabled.

Request parameters

Name	Type	Mandatory
userName	AN 1..100	Yes
API user login.		
password	String	Yes
API user password.		
pan	N13..19	Yes
Card number.		
expiryDate	N6	Yes
Card expiration date in the following format: *YYYYMM*.		
cardholderName	ANS 2..26	No

Name of the cardholder. This parameter is verified according to the following criteria: · Acceptable characters are: Latin letters, 0-9, $,), (, -, . , a space · Cardholder name must start with a letter · Minimum length: 2 characters · Maximum length: 26 characters · Null is valid · Uppercase and lowercase are acceptable		
Clientid	ANS 1..255	Yes
Customer identifier in the merchant system. This parameter is mandatory for bindings.		

Response parameters

Name	Type	Mandatory
maskedPan	N13..19	Yes
Masked card number.		
expiryDate	N6	Yes
Card expiration date in the following format: *YYYYMM*.		

cardholderName	ANS 2..26	No
Name of the cardholder. This parameter is verified according to the following criteria: · Acceptable characters are: Latin letters, 0-9, $,), (, -, . , a space · Cardholder name must start with a letter · Minimum length: 2 characters · Maximum length: 26 characters · Null is valid · Uppercase and lowercase are acceptable		
clientid	ANS 1..255	Yes
Customer identifier in the merchant system. This parameter is mandatory for bindings.		
bindingId	AN..255	Yes
Identifier of the binding created. It may be used only if the merchant has the permission to work with bindings. If this parameter is sent in the **registerOrder** request, the following actions are applied: 1. This order can be paid only by binding. 2. The payer is redirected to a payment page where the CVC must be entered.		
errorCode	N 1..3	No

Response code: · 0 — a successful transaction · Any other number — an error occurred when processing the request		
error	AN..512	No
Description of the error.		

Error codes (errorCode) and messages (errorMessage)

Code	Message	Description
0	No system error.	No errors.
1	Access is denied.	The user does not have the necessary permissions to access this resource.
1	Some field is invalid.	A form input or a data field does not meet the required format or validation rules.
1	System error.	Software or hardware issue or malfunction.

Request example

```
http://<host:port>/payment/rest/createBindingNoPayment.do?
 userName=apiuser&password=apiuserpassword&pan=
 2200000022940500&expiryDate=202107&cardholderName=IVAN&ClientId=888
```

Response example

```
{"maskedPan":"220000****0500","expiryDate":"202107","cardholderName":
 "IVAN","clientId":"888","bindingId":"e3ada8ba-2615-452c-a3f7-cf7e2f400c8b",
 "errorCode":0,"error":false}
```

4.15.2 Creating bindings without payments anonymously

The **createBindingNoPaymentAnonymous.do** method is used to create a binding without a payment anonymously. The method deactivates other bindings for the same merchant and **clientID**.

Note: To be able to execute this method, a merchant must have the **Can create bindings without payment** permission enabled.

Request parameters

Name	Type	Mandatory
merchantLogin	AN..255	Yes
API user login.		

pan	N13..19	Yes
Card number.		
expiryDate	N6	Yes
Card expiration date in the following format: *YYYYMM*.		
cardholderName	ANS 2..26	No
Name of the cardholder. This parameter is verified according to the following criteria: · Acceptable characters are: Latin letters, 0-9, $,), (, -, . , a space · Cardholder name must start with a letter · Minimum length: 2 characters · Maximum length: 26 characters · Null is valid · Uppercase and lowercase are acceptable		
Clientid	ANS 1..255	Yes
Customer identifier in the merchant system. This parameter is mandatory for bindings.		

Response parameters

Name	Type	Mandatory
maskedPan	N13..19	Yes
Masked card number.		
expiryDate	N6	Yes
Card expiration date in the following format: *YYYYMM*.		
cardholderName	ANS 2..26	No
Name of the cardholder. This parameter is verified according to the following criteria: · Acceptable characters are: Latin letters, 0-9, $,), (, -, . , a space · Cardholder name must start with a letter · Minimum length: 2 characters · Maximum length: 26 characters · Null is valid · Uppercase and lowercase are acceptable		
clientid	ANS 1..255	Yes
Customer identifier in the merchant system. This parameter is mandatory for bindings.		

bindingId	AN..255	Yes
Identifier of the binding created. It may be used only if the merchant has the permission to work with bindings. If this parameter is sent in the **registerOrder** request, the following actions apply: 1. This order can be paid only by binding. 2. The payer is redirected to a payment page where the CVC must be entered.		
errorCode	N 1..3	No
Response code: · 0 — a successful transaction · Any other number — an error occurred when processing the request		
error	AN..512	No
Description of the error.		

Error codes (errorCode) and messages (errorMessage)

Code	Message	Description
0	No system error.	No errors.

1	Access is denied.	The user does not have the necessary permissions to access this resource.
1	Some field is invalid.	A form input or a data field does not meet the required format or validation rules.
1	System error.	Software or hardware issue or malfunction.

Request example

```
http://<host:port>/payment/rest/createBindingNoPaymentAnonymous.do?
 merchantLogin=Merchant888&pan=2200000022940500&expiryDate=
 202107&cardholderName=IVAN&ClientId=888
```

Response example

```
{"maskedPan":"220000****0500","expiryDate":"202107","cardholderName":
 "IVAN","clientId":"888","bindingId":"7ced41cd-521f-41a7-9d1e-8b6ffebec9e4"
 ,"errorCode":0,"error":false}
```

4.15.3 Processing payments with bindings

The **paymentOrderBinding.do** method is used to perform a payment using a binding.

Request parameters

Name	Type	Mandatory
userName	AN 1..100	Yes
API user login.		
password	String	Yes
API user password.		
mdOrder	ANS36	Yes
Unique order number generated by EPG after the registration of the order.		
bindingId	AN..255	Yes
Identifier of the binding that was created earlier (see Managing bindings). It can only be used if the merchant has the permission to work with bindings. If this parameter is sent in the **registerOrder** request, the following is executed: · This order can only be paid by binding. · The payer is redirected to a payment page where the CVC must be entered.		

cvc	N3..4	Conditional
Card security code: CVV2 (Card Verification Value 2), CVC2 (Card Verification Code 2) and analogous codes. If the **Can pay by binding without CVV2/CVC2** option is selected, this parameter is optional for this merchant, Otherwise, the parameter is mandatory.		
language	A2	No
Language in the ISO 639-1 format. Error messages are sent in this language. If it is not specified, the default language specified in the merchant settings is used.		
clientBrowserInfo		No
JSON structure containing the client browser information for 3DS2. See the description of the clientBrowserInfo contents.		

Response parameters

Name	Type	Mandatory
redirect	AN..512	No
URL to which the customer is redirected after executing the payment, depending on the payment result.		

info	AN..512	Yes
Result of the payment attempt: · Your order is proceeded, redirecting... · Operation declined. Please check the data and available balance of the card. Redirecting... · Sorry, payment cannot be completed. Redirecting... · Payment declined. Please, contact the merchant. Redirecting... · Payment declined. Please, contact your bank. Redirecting... · Cannot connect to your bank. Please, contact your bank. Redirecting... · Processing timeout. Please, try again later. Redirecting...		
errorCode	N 1..3	No
Response code: · 0 — a successful transaction · Any other number — an error occurred when processing the request		

Error codes (errorCode) and messages (errorMessage)

Code	Message	Description
0	No system error.	No errors.
1	A required field is absent or invalid.	One of mandatory fields is either not available in the request or its value is incorrect.

2	No order found.	The order matching the specified criteria was not found.
2	Binding not found.	The binding matching the specified criteria was not found.
5	Access is denied.	The user does not have the necessary permissions to access this resource.
5	User must change the password.	The user needs to update their password for security reasons.
5	**ClientId** is empty or the merchant does not have a permission to execute the method.	The customer ID in the merchant system is not specified or the **Merchant is allowed to use bindings** merchant option is not enabled.
7	System error.	Software or hardware issue or malfunction.

Request example

```
http://<host:port>/payment/rest/paymentOrderBinding.do?userName=
 apiuser&password=apiuserpassword&mdOrder=
 7911c014-6ad3-478c-b8cf-b5d5bce3eacd&bindingId=
 7ced41cd-521f-41a7-9d1e-8b6ffebec9e4
```

Response example

{"redirect":"https://msk-ecom-wls02.bpc.in:443/epg/merchants/root/finish.html?orderId=
ffd85ad3-1851-430f-99da-a78ecd228d4d&lang=en&status=payment_deposited",
"info":"Your order is proceeded, redirecting...","errorCode":0,"orderStatusCode":2,
"orderStatusName":"payment_deposited"}

4.15.4 Modifying the card expiration date in a binding using 3DS

The **registerModifyExpirationOrder.do** method is used to register and update a binding using the 3-D Secure technology. When the cardholder is redirected to the payment page and fills in the card data, the **processModifyExpiration.do** method is called to complete the binding transaction (see Processing the updated card expiration date in a binding).

Merchant options

The following options must be enabled at the merchant level:

Permission	Description
Merchant is allowed to use binding	Enables a merchant to use bindings.
Can create and update bindings without payment	Enables a merchant to create and update bindings without payments.

The following options must be enabled either both or at least one of them:

Permission	Description
Can send a link to the payment/binding form via sms	Enables a merchant to send payment and binding pages in SMS messages.
Can send a link to the payment/binding form via email	Enables a merchant to send payment and binding pages via email.

User privileges

A user who processes payments for the merchant must have this permission enabled:

Permission	Action type	Level
Can create order for binding/update card by the link	API	Regular user
Create a link for the card update. This permission enables users to create an order and send the link to the binding web page for cardholders.		

Request parameters

Name	Type	Mandatory
userName	AN 1..100	Yes

API user login.		
password	String	Yes
API user password.		
sessionExpiredDate	N6	Yes
Session expiration date in the following format: *YYYY-MM-DDTHH:MM:ss*.		
orderNumber	ANS32	No
Unique order number generated by EPG after the registration of the order.		
email	ANS..*	Conditional
Customer's email address to send a link for creation of a binding. The **Can send a link to the payment/binding form via email** and **Can send a link to the payment/binding form via SMS** options in the merchant configuration specify whether the **email** parameter is mandatory for payments that use the respective payment method: · If both these options are enabled for the merchant, the **email** parameter is optional for this merchant. However, in this case either **phone** or **email** must be specified. · If only **Can send a link to the payment/binding form via email** option is enabled, the **email** parameter is mandatory for the merchant.		

phone	AN..255	Conditional
Customer's phone number to which a link for binding creation is sent. The **Can send a link to the payment/binding form via email** and **Can send a link to the payment/binding form via SMS** options in the merchant configuration specify whether the **phone** parameter is mandatory for payments that use the respective payment method: · If both these options are enabled for the merchant, the **email** parameter is optional for this merchant. However, in this case either **phone** or **email** must be specified. · If only **Can send a link to the payment/binding form via SMS** option is enabled, the **phone** parameter is mandatory for the merchant.		
merchantId	N..22	No
Merchant identifier in EPG.		
language	A2	No
Language in the ISO 639-1 format. Error messages are sent in this language. If it is not specified, the default language specified in the merchant settings is used.		
bindingId	AN..255	Yes

Identifier of the binding that was created earlier (see Managing bindings). It can only be used if the merchant has the permission to work with bindings. If this parameter is sent in the **registerOrder** request, the following is executed: · This order can only be paid by binding. · The payer is redirected to a payment page where the CVC must be entered.		
clientBrowserInfo		No
JSON structure containing the client browser information for 3DS2. See the description of the clientBrowserInfo contents.		
returnUrl	ANS 1..2000	No
URL to which the customer is redirected after the transaction is successfully completed.		
features	String	No
Specifies whether 3-D Secure check is enabled for a merchant: · FORCE_SSL — transaction is processed as SSL. A merchant must have a permission to process SSL transactions. · FORCE_TDS — transaction is processed as 3DS2. A merchant must have a permission to process 3DS2 transactions.		

Response parameters

Name	Type	Mandatory
orderId	ANS36	No
Unique order number generated by EPG after the registration of the order.		
formUrl	AN..512	No
URL of the card data collection page to which the customer's web browser is redirected. This parameter is not returned if the registration of the order was not successful (the error is described in the **ErrorCode** field).		
errorCode	N 1..3	No
Response code: · 0 — a successful transaction · Any other number — an error occurred when processing the request		
errorMessage	AN 1..512	No
Information about the transaction result: either a success message or the error description.		

Error codes (errorCode) and messages (errorMessage)

Code	Message	Description
0	No system error.	No errors.
1	Access is denied	The user does not have the necessary permissions to access this resource.
1	Some field is invalid.	A value in some field is incorrect.
1	System error.	Software or hardware issue or malfunction.
1	Binding not found.	The binding matching the specified criteria was not found.

Request example

```
http://<host:port>/epg/rest/public/registerModifyExpirationOrder.do?userName=
 apiuser&password=apiuserpassword&bindingId=7ced41cd-521f-41a7-9d1e-
 8b6ffebec9e4&sessionExpiredDate=2023-08-16T22:08:23&phone=73807777777&orderNumber=11111111
```

Response example

```
{"errorCode":0,"orderId":"41f226e3-62a8-4922-802a-5c02e0990319",
  "formUrl":"https://msk-ecom-wls02.bpc.in/p/QfIm42Ko"}
```

4.15.5 Modifying the card expiration date in a binding

The **extendBinding.do** method is used to modify the card expiration date in a binding.

Request parameters

Name	Type	Mandatory
userName	AN 1..100	Yes
API user login.		
password	String	Yes
API user password.		
newExpiry	N6	Yes
New expiration date (year and month) of the card in the following format: *YYYYMM*		
bindingId	AN..255	Yes

Identifier of the binding that was created earlier (see Managing bindings). It can only be used if the merchant has the permission to work with bindings. If this parameter is sent in the **registerOrder** request, the following is executed: · This order can only be paid by binding. · The payer is redirected to a payment page where the CVC must be entered.		
language	A2	No
Language in the ISO 639-1 format. Error messages are sent in this language. If it is not specified, the default language specified in the merchant settings is used.		

Response parameters

Name	Type	Mandatory
errorCode	N 1..3	No
Response code: · 0 — a successful transaction · Any other number — an error occurred when processing the request		
errorMessage	AN 1..512	No
Information message about the transaction result: a success message or the description of an error.		

Error codes (errorCode) and messages (errorMessage)

Code	Message	Description
0	No system error.	No errors.
1	A required field is absent or invalid.	One of mandatory fields is either not available in the request or its value is incorrect.
2	Binding not found.	The binding matching the specified criteria was not found.
5	Access is denied.	The user does not have the necessary permissions to access this resource.
5	User must change the password.	The user needs to update their password for security reasons.
7	System error.	Software or hardware issue or malfunction.

Request example

```
http://<host:port>/payment/rest/extendBinding.do?
 userName=apiuser&password=apiuserpassword&newExpiry=
 203107&bindingId=7ced41cd-521f-41a7-9d1e-8b6ffebec9e4
```

Response example

```
{"errorCode":"0","errorMessage":"Success"}
```

4.15.6 Processing the updated card expiration date in a binding

The **processModifyExpiration.do** method is used to complete the transaction created by the **registerModifyExpirationOrder.do** method and update the card credentials (see Modifying the card expiration date in a binding using 3DS).

Merchant options

The following options must be enabled at the merchant level:

Permission	Description
Merchant is allowed to use binding	Enables a merchant to use bindings.
Can create and update bindings without payment	Enables a merchant to create and update bindings without payments.

The following options must be enabled either both or at least one of them:

Permission	Description

Can send a link to the payment/binding form via sms	Enables a merchant to send payment and binding pages in SMS messages.
Can send a link to the payment/binding form via email	Enables a merchant to send payment and binding pages via email.

User privileges

A user who processes payments for the merchant must have this permission enabled:

Permission	Action type	Level
Can create order for binding/update card by the link	API	Regular user
Create a link for the card update. This permission enables users to create an order and send the link to the binding web page for cardholders.		

Request parameters

Name	Type	Mandatory
userName	AN 1..100	Yes
API user login.		

password	String	Yes
API user password.		
expiryDate	N6	Yes
Card expiration date in the following format: *YYYYMM*.		
CVC	N3..4	Conditional
Card security code: CVV2 (Card Verification Value 2), CVC2 (Card Verification Code 2), or similar codes. The **Can make zero-amount bindings without CVV2/CVC2** option in the merchant configuration specifies whether the **CVC** parameter is mandatory for payments that use the respective payment method: · If this option is enabled for the merchant, the **CVC** parameter is optional for this merchant. · If this option is not enabled, the **CVC** parameter is mandatory for the merchant.		
bindingId	AN..255	Yes
Identifier of the binding that was created earlier (see Managing bindings). It can only be used if the merchant has the permission to work with bindings. If this parameter is sent in the **registerOrder** request, the following is executed: · This order can only be paid by binding.		

· The payer is redirected to a payment page where the CVC must be entered.		
mdOrder	ANS36	Yes
Order number generated by EPG after the registration of the order.		
language	A2	No
Language in the ISO 639-1 format. Error messages are sent in this language. If it is not specified, the default language specified in the merchant settings is used.		
threeDs2ReturnUrl	ANS 1..512	No
Return URL for user redirect from the issuer ACS after 3DS2 authentication is completed. If it is not specified, the user will be redirected to the specific EPG URL.		
threeDSCompInd	String	No
Specifies whether the 3DS Method Completion notification was received by the merchant: · Y — the 3DS Method Completion notification from the issuer ACS was received by the merchant using the address specified by the **threeDSMethodNotificationURL** parameter of the **checkPreliminary** method (see Request to check card eligibility for 3DS2 before the main request). · N — the 3DS Method Completion notification from the issuer ACS was not received by the merchant.		

checkCardValidity	Boolean	No
Enables the account validation request (**debitAccountVerification**) to be sent after the main request: · true — the request to verify card credentials is sent automatically after the **processModifyExpiration** request only if **features =** FORCE_SSL. · false — the request to verify card credentials is not sent. This is the default value. **ATTENTION:** If **features =** FORCE_TDS, the card verification request is *always* sent regardless of the **checkCardValidity** parameter value (true or false).		

Response parameters

Name	Type	Mandatory
redirect	AN 512	No
URL to which the customer is redirected after executing the payment depending on the payment result. This parameter is returned in the response if **features=**FORSE_SSL in the **registerModifyExpiration.do** request. For NPA orders registered via the EPG web interface (using the **Create a link for binding** and **Create a link for card update** functions), the final successful and unsuccessful redirect URLs specified by the return URL after payment/binding and return URL on error parameters are used. For NPA orders registered using API, the final successful and unsuccessful redirect URLs are the same as for payment transactions.		

errorCode	N 1..3	No
Response code: · 0 — a successful transaction · Any other number — an error occurred when processing the request		
errorMessage	AN 1..512	No
Information about the transaction result: either a success message or the error description.		
info	AN..512	No
Result of the payment attempt: · Your order is proceeded, redirecting... · Operation declined. Please check the data and available balance of the card. Redirecting... · Sorry, payment cannot be completed. Redirecting... · Payment declined. Please, contact the merchant. Redirecting... · Payment declined. Please, contact your bank. Redirecting... · Cannot connect to your bank. Please, contact your bank. Redirecting... · Processing timeout. Please, try again later. Redirecting...		
cReq	AN..512	Conditional
Challenge Request message. EMV 3-D Secure message sent by the 3DS SDK or 3DS Server where additional information is sent from the cardholder to the ACS to		

support the authentication process. It must be present for 3-D Secure 2 if a cardholder challenge is required.		
acsUrl	AN..512	Conditional
URL of the ACS server. This parameter is used in payments that require additional authentication on the issuing bank's ACS.		
handle3ds2MethodInEpg	Boolean	Conditional
Specifies whether 3DS2 method is used in EPG when executing the **processModifyExpiration.do** request: · true · false		

Error codes (errorCode) and messages (errorMessage)

Code	Message	Description
0	No system error.	No errors.
1	A required field is absent or invalid.	One of mandatory fields is either not available in the request or its value is incorrect.
2	Binding not found.	The binding matching the specified criteria was not found.

5	Access is denied.	The user does not have the necessary permissions to access this resource.
7	System error.	Software or hardware issue or malfunction.

Request example

```
http://<host:port>/epg/rest/internal/processModifyExpiration.do?
 expiryDate=203107&bindingId=7ced41cd-521f-41a7-9d1e-8b6ffebec9e4&
 mdOrder=f318e477-5d19-4d65-b2ab-673493cdfb1a&cvc=443
```

Response example

```
{
    "redirect": "https://msk-ecom-wls02.bpc.in:443/epg/merchants/
     anna_testapi/finish.html?orderId=
     79cd7db7-e49c-4d67-bac9-ffdcfc20acfc&lang=en&status=card_modified",
    "errorCode": 0
 }
```

4.15.7 Deactivating a binding

The **unBindCard.do** method is used to deactivate a binding.

Note: To be able to execute this method, a merchant must have the **Bindings deactivation on a payment page is allowed** permission enabled.

Request parameters

Name	Type	Mandatory
userName	AN 1..100	Yes
API user login.		
password	String	Yes
API user password.		
bindingId	AN..255	Yes
Identifier of the binding that was created earlier (see Managing bindings). It can only be used if the merchant has the permission to work with bindings. If this parameter is sent in the **registerOrder** request, the following is executed: · This order can only be paid by binding. · The payer is redirected to a payment page where the CVC must be entered.		
language	A2	No
Language in the ISO 639-1 format. Error messages are sent in this language. If it is not specified, the default language specified in the merchant settings is used.		

Response parameters

Name	Type	Mandatory
errorCode	N 1..3	No
Response code: · 0 — a successful transaction · Any other number — an error occurred when processing the request		
errorMessage	AN 1..512	No
Information message about the transaction result: a success message or the description of an error.		

Error codes (errorCode) and messages (errorMessage)

Code	Message	Description
0	No system error.	No errors.
1	Invalid state.	Incorrect binding status (when attempting to deactivate an inactive binding or active a binding which is active).
2	Binding not found.	The binding matching the specified criteria was not found.

5	Access is denied.	The user does not have the necessary permissions to access this resource.
5	User must change the password.	The user needs to update their password for security reasons.
7	System error.	Software or hardware issue or malfunction.

Request example

```
http://<host:port>/payment/rest/unBindCard.do?
 userName=apiuser&password=apiuserpassword&bindingId=
 7ced41cd-521f-41a7-9d1e-8b6ffebec9e4
```

Response example

```
{"errorCode":"0","errorMessage":"Success"}
```

4.15.8 Reactivating a binding

The **bindCard.do** method is used to activate a binding that was deactivated.

Request parameters

Name	Type	Mandatory
userName	AN 1..100	Yes
API user login.		
password	String	Yes
API user password.		
bindingId	AN..255	Yes
Identifier of the binding that was created earlier (see Managing bindings). It can only be used if the merchant has the permission to work with bindings. If this parameter is sent in the **registerOrder** request, the following is executed: · This order can only be paid by binding. · The payer is redirected to a payment page where the CVC must be entered.		
language	A2	No
Language in the ISO 639-1 format. Error messages are sent in this language. If it is not specified, the default language specified in the merchant settings is used.		

Response parameters

Name	Type	Mandatory
errorCode	N 1..3	No
Response code: · 0 — a successful transaction · Any other number — an error occurred when processing the request		
errorMessage	AN 1..512	No
Information message about the transaction result: a success message or the description of an error.		

Error codes (errorCode) and messages (errorMessage)

Code	Message	Description
0	No system error.	No errors.
2	Invalid state.	Incorrect binding status (when attempting to deactivate an inactive binding or active a binding which is active).
2	Binding not found.	The binding matching the specified criteria was not found.

5	Access is denied.	The user does not have the necessary permissions to access this resource.
5	User must change the password.	The user needs to update their password for security reasons.
7	System error.	Software or hardware issue or malfunction.

Request example

```
http://<host:port>/payment/rest/bindCard.do?userName=
 apiuser&password=apiuserpassword&bindingId=
 e3ada8ba-2615-452c-a3f7-cf7e2f400c8b
```

Response example

```
{"errorCode":0,"errorMessage":"Success"}
```

4.15.9 Checking activation of a binding

The **isBindingEnabled.do** method is used to check whether a binding is active.

Request parameters

Name	Type	Mandatory

merchantLogin	AN 1..100	Yes
Merchant login to access EPG.		
bindingId	AN..255	Yes
Identifier of the binding that was created earlier (see Managing bindings). It can only be used if the merchant has the permission to work with bindings. If this parameter is sent in the **registerOrder** request, the following is executed: · This order can only be paid by binding. · The payer is redirected to a payment page where the CVC must be entered.		

Response parameter

Name	Type	Mandatory
result	Boolean	Yes
Result of the binding activation check: · true — the binding is active. · false — the binding is deactivated.		

Error codes (errorCode) and messages (errorMessage)

Code	Message	Description

0	No system error.	No errors.
2	Binding not found.	The binding matching the specified criteria was not found.
5	Access is denied.	The user does not have the necessary permissions to access this resource.
5	User must change the password.	The user needs to update their password for security reasons.
7	System error.	Software or hardware issue or malfunction.

Request example

```
http://<host:port>/payment/rest/isBindingEnabled.do?
 merchantLogin=Merchant888&bindingId=
 e3ada8ba-2615-452c-a3f7-cf7e2f400c8b
```

Response example

```
{"result":true}
```

4.15.10 Checking the order amount

The **matchSum.do** method is used to compare the amount of an order paid using a binding with a specified amount.

Request parameters

Name	Type	Mandatory
merchantLogin	AN 1..100	Yes
Merchant login to access EPG.		
bindingId	AN..255	Yes
Identifier of the binding that was created earlier (see Managing bindings). It can only be used if the merchant has the permission to work with bindings. If this parameter is sent in the **registerOrder** request, the following is executed: · This order can only be paid by binding. · The payer is redirected to a payment page where the CVC must be entered.		
mdOrder	ANS36	Yes
Unique order number generated by EPG after the registration of the order.		
amount	N 1..12	Yes
Amount of the order.		

Response parameters

The response returns one of the following values:

- True if the amount matches
- False if the amount does not matches

4.15.11 Getting the list of bindings

The **getBindings.do** method is used to obtain the list of existing bindings for the customer via **clientId** or **bindingId**.

Request parameters

Name	Type	Mandatory
userName	AN 1..100	Yes
API user login.		
password	String	Yes
API user password.		
clientId	ANS 1..255	Yes
Customer identifier in the merchant system. This parameter is mandatory for bindings.		

bindingId	AN..255	No
Identifier of the binding that was created earlier (see Managing bindings). It can only be used if the merchant has the permission to work with bindings. If this parameter is sent in the **registerOrder** request, the following is executed: · This order can only be paid by binding. · The payer is redirected to a payment page where the CVC must be entered.		
showExpired	B	No
Flag that specifies whether to include binding of expired cards: · TRUE · FALSE (this is the default value).		

Response parameters

Name	Type	Mandatory
errorCode	N 1..3	No
Response code: · 0 — a successful transaction · Any other number — an error occurred when processing the request		
error	AN..512	No

Description of the error.		
bindings[]	AN..1024	No
Array of parameters for each binding. The bindings block parameters are described below.		

The binding block parameters

Name	Type	Mandatory
bindingId	AN..255	Yes
Identifier of the binding created. It may be used only if the merchant has the permission to work with bindings. If this parameter is sent in the **registerOrder** request, the following actions apply: 1. This order can be paid only by binding. 2. The payer is redirected to a payment page where the CVC must be entered.		
maskedPan	N13..19	Yes
Masked card number.		
expiryDate	N6	Yes

Card expiration date in the following format: *YYYYMM*.

Error codes (errorCode) and messages (errorMessage)

Code	Message	Description
0	No system error.	No errors.
1	**clientId** is missing	The client ID is not properly configured or is missing from the request.
2	Invalid state.	Incorrect binding status (when attempting to deactivate an inactive binding or active a binding which is active).
2	Data is not found.	One of the parameter values was entered incorrectly.
5	Access is denied.	The user does not have the necessary permissions to access this resource.
5	User must change the password.	The user needs to update their password for security reasons.

7	System error.	Software or hardware issue or malfunction.

Request example

```
http://<host:port>/payment/rest/getBindings.do?
 userName=apiuser&password=apiuserpassword&clientId=888
```

Response example

```
{"errorCode":"0","errorMessage":"Success","bindings":
 [{"bindingId":"e3ada8ba-2615-452c-a3f7-cf7e2f400c8b",
 "maskedPan":"220000****0500","expiryDate":"202107"},
 {"bindingId":"3752a0cd-f6be-4eb3-8309-d2d6a40d4add",
 "maskedPan":"440000****0600","expiryDate":"202107"}]}
```

4.15.12 Getting the list of bindings by PAN or bindingId

The **getBindingsByCardOrId.do** method is used to obtain the list of existing bindings for the customer via **PAN** or **bindingId**.

Note: To be able to execute this method, a merchant must have the **Can get bindings by card number** permission enabled.

Request parameters

Name	Type	Mandatory

userName	AN 1..100	Yes
API user login.		
password	String	Yes
API user password.		
pan	N 13...19	No
Card number. This parameter is required if **bindingId** is not specified.		
bindingId	AN..255	No
Identifier of the binding that was created earlier (see Managing bindings). It can only be used if the merchant has the permission to work with bindings. If this parameter is sent in the **registerOrder** request, the following is executed: · This order can only be paid by binding. · The payer is redirected to a payment page where the CVC must be entered. This parameter is required if **PAN** is not specified.		
showExpired	B	No

Specifies whether expired card bindings are included in the search result: · TRUE · FALSE (this is the default value)

Response parameters

Name	Type	Mandatory
errorCode	N 1..3	No
Response code: · 0 — a successful transaction · Any other number — an error occurred when processing the request		
error	AN..512	No
Description of the error.		
bindings[]	AN..1024	No
Array of parameters for each binding that are described below.		

Binding block parameters

Name	Type	Mandatory

bindingId	AN..255	Yes
Identifier of the binding created. It may be used only if the merchant has the permission to work with bindings. If this parameter is sent in the **registerOrder** request, the following actions apply: 1. This order can be paid only by binding. 2. The payer is redirected to a payment page where the CVC must be entered.		
maskedPan	N13..19	Yes
Masked card number.		
expiryDate	N6	Yes
Card expiration date in the following format: *YYYYMM*.		
clientId	ANS 1..255	Yes
Customer identifier in the merchant system. This parameter is mandatory for bindings.		

Error codes (errorCode) and messages (errorMessage)

Code	Message	Description

0	No system error.	No errors.
1	Both **PAN** and **bindingId** are missing	The card number (PAN) and binding ID are not properly configured or missing from the request.
2	Data is not found.	One of the parameter values was entered incorrectly.
5	Access is denied.	The user does not have the necessary permissions to access this resource.
5	User must change the password.	The user needs to update their password for security reasons.
7	System error.	Software or hardware issue or malfunction.

Request example

```
http://<host:port>/payment/rest/getBindingsByCardOrId.do?
 userName=apiuser&password=apiuserpassword&pan=4400000041520600
```

Response example

{"errorCode":"0","errorMessage":"Success","bindings":
[{"bindingId":"3752a0cd-f6be-4eb3-8309-d2d6a40d4add","maskedPan":
"440000****0600","expiryDate":"202107","clientId":"888"}]}

4.15.13 Request for a zero amount binding

The **registerBindingOrder.do** method is used to initially register a zero amount binding. When the cardholder is redirected to the payment page and fills in the card data, the **processBindingOrder.do** method is triggered to complete the binding transaction (see Request for processing a zero amount binding).

Merchant options

The following options are configured at the merchant level:

Permission	Description
Merchant is allowed to use binding	Enables a merchant to use bindings.
Can create and update bindings without payment	Enables a merchant to create and update bindings without payments.
Can send a link to the payment/binding form via sms	Enables a merchant to send payment and binding pages in SMS messages.

Can send a link to the payment/binding form via email	Enables a merchant to send payment and binding pages via email.

User privileges

The following permission is configured for users who processes payments for the merchant.

Permission	Action type	Level
Can create order for binding/update card by the link	API	Regular user
Create a link for the card update. This permission enables users to create an order and send the link to the binding web page for cardholders.		

Request parameters

Name	Type	Mandatory
userName	AN 1..100	Yes
API user login.		
password	String	Yes

API user password.		
sessionExpiredDate	N6	Yes
Session expiration date in the following format: *YYYY-MM-DDTHH:MM:ss*.		
orderNumber	ANS32	No
Unique order number generated by EPG after the registration of the order.		
name		No
Customer name.		
clientId	ANS 1..255	Yes
Customer identifier in the merchant system. This parameter is mandatory for bindings.		
features[]	String	Yes
Specifies whether 3-D Secure check is enabled for a merchant: · FORCE_SSL — transaction is processed as SSL. A merchant must have a permission to process SSL transactions.		

· FORCE_TDS — transaction is processed as 3DS2. A merchant must have a permission to process 3DS2 transactions.		
email	ANS..*	No
Customer's email address to send a link for creation of a binding. If the email is specified, the following permissions are required: · Can send a link to the payment/binding form via email — for the merchant · Can create order for binding/update card by the link — for the user		
phone	AN..255	No
Customer's phone number to which a link for binding creation is sent. If the phone number is specified, the following permissions are required: · Can send a link to the payment/binding form via sms — for the merchant · Can create order for binding/update card by the link — for the user		
merchantId	N..22	No
Merchant identifier in EPG.		
description	ANS..512	No
Order description.		
language	A2	No

Language in the ISO 639-1 format. Error messages are sent in this language. If it is not specified, the default language specified in the merchant settings is used.		
returnUrl	ANS 1..2000	No
URL to which the customer is redirected after the transaction is successfully completed.		
bindingManagedByMerchant		No
Entity that manages bindings: · true — bindings are stored and managed by merchants. · false — bindings are stored and managed by acquirer or EPG. This is the default value.		
bindingUsage		Conditional
Type of the Card-on-File (CoF) transaction: · saveCard — card is stored within a transaction. · paymentBySavedCard — card stored during the previous transactions is used in the next transaction. **Note**: This parameter can only be used if **bindingManagedByMerchant =** true.		
originalTransactionData	Not applicable	No

See the description of the Transaction Data block below.		
transactionInitiator	String	No
Originator of the transaction: · CIT — cardholder-initiated transaction · MIT — merchant-initiated transaction		

Transaction Data block

A single signed data block is used to transfer all original transaction data fields to and from a PCI DSS merchant, instead of multiple separate fields.

An RSA key pair with **CertificateType.TRANSACTION_DATA_SIGNATURE** must be generated for the merchant.

The block includes the following:

Header

Field	Description
alg	Algorithm (RS256).
kid	ID of the corresponding record in the key_mgmt_certificate table.
typ	Header type (JOSE).

merchantLogin	Login of the merchant.

Payload

Field	Description
mdOrder	This field is reserved for future use.
transDate	This field is reserved for future use.
networkReferenceData	Network reference data returned from the authorization host.
transactionDataElements	Data elements of the authorization request.
isSsl	Specifies whether 3DS2 authentication was not performed for the transaction.
dsTransId	Directory Server Transaction ID, UUID (only for 3DS2 transactions).
acsTransId	Transaction ID in ACS (only for 3DS2 transactions).
initTransactionDate	Initiating transaction date (only for 3DS2 transactions).

version	Constant value of 1. This field is reserved for future use.
panSalt	Base64-encoded 16 bytes random salt (the resulting string will be approximately 24 character long; from **SecureRandom**).
panHash	Base64 SHA-256 hash of byte[] = PAN.getBytes + salt.bytes.

The block validation fails if:

- The field is missing for MIT requests with **recurringPaymentManagedByMerchant=**true or **bindingManagedByMerchant=**true.
- A different **merchantLogin** is used.
- The key provided in the JWS header cannot be found for the current merchant.
- Signature validation fails (both technical and not errors).

Response parameters

Name	Type	Mandatory
orderId	ANS36	No
Unique order number generated by EPG after the registration of the order.		
formUrl	AN..512	No

URL of the card data collection page to which the customer's web browser is redirected. This parameter is not returned if the registration of the order was not successful (the error is described in the **ErrorCode** field).		
errorCode	N 1..3	No
Response code: · 0 — a successful transaction · Any other number — an error occurred when processing the request		
errorMessage	AN 1..512	No
Information about the transaction result: either a success message or the error description.		
error	AN..512	No
Description of the error.		

Error codes (errorCode) and messages (errorMessage)

Code	Message	Description
0	No system error.	No errors.

1	Access is denied.	The user does not have the necessary permissions to access this resource.
1	Some field is invalid	A form input or a data field does not meet the required format or validation rules.
5	System error.	Software or hardware issue or malfunction.

Request example

```
http://<host:port>/epg/rest/public/registerBindingOrder.do?userName=
 apiuser&password=apiuserpassword&sessionExpiredDate=
 2023-08-16T22:08:23&clientId=111&features=FORCE_SSL&merchantId=
 50112&phone=73807777777&orderNumber=11111111
```

Response example

```
{"errorCode":0,"orderId":"35d5e6bb-607f-46e7-92d3-83ebd48d17d4",
 "formUrl":"https://msk-ecom-wls02.bpc.in/p/NdXmu2B_"}
```

4.15.14 Request for processing a zero amount binding

The **processBindingOrder.do** method is used to complete a binding transaction created by the **registerBindingOrder.do** method (see Request for a zero amount binding).

Merchant options

The following options are configured at the merchant level:

Permission	Description
Merchant is allowed to use binding	Enables a merchant to use bindings.
Can create and update bindings without payment	Enables a merchant to create and update bindings without payments.

Request parameters

Name	Type	Mandatory
userName	AN 1..100	Yes
API user login.		
password	String	Yes
API user password.		
pan	N 13...19	No

Card number. This parameter is mandatory.		
expiryDate	N6	Yes
Card expiration date in the following format: *YYYYMM*.		
CVC	N3..4	Conditional
Card security code: CVV2 (Card Verification Value 2), CVC2 (Card Verification Code 2) or a similar code. The **cvc** parameter is optional for the merchant if the **Can pay by binding without CVV2/CVC2** permission is enabled for this merchant. If the **Can pay by binding without CVV2/CVC2** permission is not enabled, the **cvc** parameter is mandatory for the merchant.		
cardholderName	A 2..26	No
Name of the cardholder. This parameter is verified according to the following criteria: · Acceptable characters are: Latin letters, 0-9, $,), (, -, . , a space · Cardholder name must start with a letter · Minimum length: 2 characters · Maximum length: 26 characters · Null is valid · Uppercase and lowercase are acceptable		

mdOrder	ANS36	Yes
Order number generated by EPG after the registration of the order.		
language	A2	No
Language in the ISO 639-1 format. Error messages are sent in this language. If it is not specified, the default language specified in the merchant settings is used.		
clientBrowserInfo	Not applicable	No
JSON structure containing the client browser information for 3DS2. See the description of the clientBrowserInfo contents.		
threeDs2ReturnUrl	ANS 1..512	No
Return URL for user redirect from the issuer ACS after 3DS2 authentication is completed. If it is not specified, the user will be redirected to the specific EPG URL.		
threeDSCompInd	String	No

Specifies whether the 3DS Method Completion notification was received by the merchant: · Y — the 3DS Method Completion notification from the issuer ACS was received by the merchant using the address specified by the **threeDSMethodNotificationURL** parameter of the **checkPreliminary** method (see Request to check card eligibility for 3DS2 before the main request). · N — the 3DS Method Completion notification from the issuer ACS was not received by the merchant.		
checkCardValidity	Boolean	No
Enables the account validation request (**debitAccountVerification**) to be sent after the main request: · true — the request to verify card credentials is sent automatically after the **processBindingOrder** request only if **features** = FORCE_SSL. · false — the request to verify card credentials is not sent. This is the default value. **ATTENTION:** If **features** = FORCE_TDS, the card verification request is *always* sent regardless of the **checkCardValidity** parameter value (true or false).		

Response parameters

Name	Type	Mandatory
redirect	AN 512	No

URL to which the customer is redirected after executing the payment depending on the payment result. This parameter is returned in the response if **features=**FORSE_SSL in the **registerBindingOrder.do** request.

For NPA orders registered via the EPG web interface (using the **Create a link for binding** and **Create a link for card update** functions), the final successful and unsuccessful redirect URLs specified by the return URL after payment/binding and return URL on error parameters are used.

For NPA orders registered using API, the final successful and unsuccessful redirect URLs are the same as for payment transactions.

errorCode	N 1..3	No
Response code: · 0 — a successful transaction · Any other number — an error occurred when processing the request		
errorMessage	AN 1..512	No
Information about the transaction result: either a success message or the error description.		
error	AN..512	No
Description of the error.		
info	AN..512	No

Result of the payment attempt:

- Your order is proceeded, redirecting...
- Operation declined. Please check the data and available balance of the card. Redirecting...
- Sorry, payment cannot be completed. Redirecting...
- Payment declined. Please, contact the merchant. Redirecting...
- Payment declined. Please, contact your bank. Redirecting...
- Cannot connect to your bank. Please, contact your bank. Redirecting...
- Processing timeout. Please, try again later. Redirecting...

cReq	AN..512	Conditional
Challenge Request message. EMV 3-D Secure message sent by the 3DS SDK or 3DS Server where additional information is sent from the cardholder to the ACS to support the authentication process. It must be present for 3-D Secure 2 if a cardholder challenge is required.		
acsUrl	AN..512	Conditional
URL of the ACS server. This parameter is used in payments that require additional authentication on the issuing bank's ACS.		
handle3ds2MethodInEpg	Boolean	Conditional
Specifies whether 3DS2 method is used in EPG when executing the **processBindingOrder.do** request: · true · false		

Error codes (errorCode) and messages (errorMessage)

Value	Description	Description
0	No system error.	No errors.
1	Access is denied.	The user does not have the necessary permissions to access this resource.
1	Some field is invalid.	A form input or a data field does not meet the required format or validation rules.
1	System error.	Software or hardware issue or malfunction.

Request example

```
http://<host:port>/epg/rest/public/processBindingOrder.do?userName=
 userapi&password=userapipassword&mdOrder=
 f318e477-5d19-4d65-b2ab-673493cdfb1a&pan=
 4000010000000118&expiryDate=202512&cvc=443&phone=
 73807777777&cardholderName=test
```

Response example

{"redirect":"https://msk-ecom-wls02.bpc.in:443/epg/merchants/anna_testapi/finish.html?orderId=
e2f976ed-4de0-4677-a244-15dca62886a2&lang=en&status=card_added","errorCode":0}

4.16 Card-on-File (CoF) transactions for PCI DSS merchants

This section describes how payments are processed if merchants store card credentials on their side. These types of merchants are PCI DSS certified.

The following transactions are supported for merchants certified with PCI DSS:

- Creating a binding (saving the card for a non-payment transaction)
- CIT payments with a binding including:
  - Saving a card during payment or paying with a binding
  - Performing transactions with stored card (CoF)
  - Initial recurring/installment payment (CIT)
- MIT payments with a binding including:
  - Subsequent recurring/installment payment (MIT)
  - No-Show payment

4.16.1 Additional request and response parameters

The following optional parameters are used for the recurring and binding requests:

bindingManagedByMerchant	
Description	Specifies the entity that manages bindings: · true — bindings are stored and managed by merchants.

	· false — bindings are stored and managed by acquirer or EPG. This is the default value.
Affected APIs	The following APIs are affected: · **register.do** · **registerPreAuth.do** · **registerBindingOrder.do** For more information, see Order registration request and Request for a zero amount binding.
bindingUsage	
Description	Type of the card-on-file (CoF) transaction: · saveCard — card is stored within a transaction. · paymentBySavedCard — card stored during the previous transactions is used in the next transaction. **Note**: This parameter can only be used if **bindingManagedByMerchant =** true.
Affected APIs	The following APIs are affected: · **register.do** · **registerPreAuth.do** For more information, see Order registration request.
originalTransactionData	
Description	See the description of the Transaction Data block below.

Affected APIs	The affected API is **register.do**. For more information, see Order registration request.
transactionInitiator	
Description	Originator of the transaction: · CIT — cardholder-initiated transaction · MIT — merchant-initiated transaction
Affected APIs	The following APIs are affected: · **register.do** · **registerPreAuth.do** · **registerBindingOrder.do** For more information, see Order registration request and Request for a zero amount binding.
transactionData	
Description	This optional parameter is used in the responses of recurring and binding requests. See the description of the Transaction Data block below.
Affected APIs	The following APIs are affected: · **paymentOrder.do** · **getOrderStatus.do** (**getOrderStatusExtended.do**)

	For more information, see Payment request, Order status request, and Extended order status request.

Transaction Data block

A single signed data block is used to transfer all original transaction data fields to and from a PCI DSS merchant, instead of multiple separate fields.

An RSA key pair with **CertificateType.TRANSACTION_DATA_SIGNATURE** must be generated for the merchant.

The block includes the following:

Header

Field	Description
alg	Algorithm (RS256).
kid	ID of the corresponding record in the key_mgmt_certificate table.
typ	Header type (JOSE).
merchantLogin	Login of the merchant.

Payload

Field	Description

mdOrder	This field is reserved for future use.
transDate	This field is reserved for future use.
networkReferenceData	Network reference data returned from the authorization host.
transactionDataElements	Data elements of the authorization request.
isSsl	Specifies whether 3DS2 authentication was not performed for the transaction.
dsTransId	Directory Server Transaction ID, UUID (only for 3DS2 transactions).
acsTransId	Transaction ID in ACS (only for 3DS2 transactions).
initTransactionDate	Initiating transaction date (only for 3DS2 transactions).
version	Constant value of 1. This field is reserved for future use.
panSalt	Base64-encoded 16 bytes random salt (the resulting string will be approximately 24 character long; from **SecureRandom**).

panHash	Base64 SHA-256 hash of byte[] = PAN.getBytes + salt.bytes.

The block validation fails if:

- The field is missing for MIT requests with **recurringPaymentManagedByMerchant=**true or **bindingManagedByMerchant=**true.
- A different **merchantLogin** is used.
- The key provided in the JWS header cannot be found for the current merchant.
- Signature validation fails (both technical and not errors).

4.16.2 Creating a binding (saving the card for a non-payment transaction)

A binding is created as follows:

1. A merchant registers an order (**registerBindingOrder.do**) with the following parameter:
   - **bindingManagedByMerchant =** true — bindings are managed by the merchant and EPG is requested not to save the bindings in the system.
2. The merchant's system initiates the **processBindingOrder.do** request.
3. When authorization is completed, EPG responds to the merchant and passes **transactionData**:
   - In a response to the **paymentOrder.do** request only for SSL and Frictionless transactions.
   - Always in a response to the **getOrderStatus.do** (**getOrderStatusExtended.do**) request.

Notes: The **createBindingNoPayment.do** method is never used for the PCI DSS merchants. The **transactionData** data must be saved by the merchant if **bindingManagedByMerchant** = true. If **bindingManagedByMerchant** = false, a merchant can ignore **transactionData**.

4.16.3 CIT payments with a binding

The following cardholder-initiated (CIT) payments can be performed for merchants certified with PCI DSS:

- Saving a card during payment or paying with a binding
- Performing transactions with stored card (CoF)
- Initial recurring/installment payment (CIT)

4.16.4 MIT payments with a binding

The following merchant-initiated (MIT) payments can be performed for merchants certified with PCI DSS:

- Subsequent recurring/installment payment (MIT)
- No-Show payment

For more information, see:

- Additional request and response parameters
- Creating a recurring payment template

4.16.4.1 Additional request and response parameters

The following optional parameters are used for the MIT requests:

recurringPaymentManagedByMerchant	
Description	Entity that manages recurring payments: · true — recurring payments are stored and managed by merchants. · false — recurring payments are stored and managed by acquirer or EPG. This is the default value.
Affected APIs	The following APIs are affected: · **register.do** · **createRecurringTemplateNoPayment.do**

	For more information, see Order registration request and Request to create a template for recurring payments with or without 3DS.
recurringPaymentForm	
Description	Type of the recurring payment: · recurring § installment This parameter is mandatory if **recurringPaymentManagedByMerchant** = true.
Affected APIs	The **register.do** API is affected. For more information, see Order registration request.
recurringExpiry	
Description	Recurring payment expiration date in the following format: *YYYYMMDD*. Typically, this parameter is used for installment transactions. However, it can also be available in recurring transactions. If the value is empty, a default value of 99991231 is used for the Visa Areq A000000802-004 message extension.
Affected APIs	The **register.do** API is affected. For more information, see Order registration request

originalTransactionData	
Description	See the description of the Transaction Data block.
Affected APIs	The following APIs are affected: · **register.do** · **registerPreAuth.do** · **registerBindingOrder.do** For more information, see Order registration request and Request for a zero amount binding.
transactionInitiator	
Description	Originator of the transaction: · CIT — cardholder-initiated transaction · MIT — merchant-initiated transaction
Affected APIs	The following APIs are affected: · **register.do** · **registerPreAuth.do** · **registerBindingOrder.do** For more information, see Order registration request and Request for a zero amount binding.
originalDsTransId	
Description	Directory Server Transaction ID, UUID (only for 3DS2 transactions).

Affected APIs	The following APIs are affected: · **register.do** if **recurringPaymentManagedByMerchant =** true · **paymentOrder.do** For more information, see Order registration request and Payment request.
originalAcsTransId	
Description	Original transaction identifier in ACS. This is only for 3DS2 transactions.
Affected APIs	The following APIs are affected: · **register.do** if **recurringPaymentManagedByMerchant =** true · **paymentOrder.do** For more information, see Order registration request and Payment request.
originalInitTransactionDate	
Description	Initiating transaction date. This is only for 3DS2 transactions.
Affected APIs	The following APIs are affected: · **register.do** if **recurringPaymentManagedByMerchant =** true · **paymentOrder.do**

	For more information, see Order registration request and Payment request.
originalInstallmentNumber	
Description	Initiating installment payment number.
Affected APIs	The following APIs are affected: · **register.do** if **recurringPaymentManagedByMerchant =** true and **recurringPaymentForm =** installment. · **paymentOrder.do** For more information, see Order registration request and Payment request.
transactionData	
Description	This is an optional parameter is used for the responses. See the description of the Transaction Data block below.
Affected APIs	The following APIs are affected: · **paymentOrder.do** · **getOrderStatus.do** (**getOrderStatusExtended.do**) For more information, see Payment request, Order status request, and Extended order status request.

Transaction Data block

A single signed data block is used to transfer all original transaction data fields to and from a PCI DSS merchant, instead of multiple separate fields.

An RSA key pair with **CertificateType.TRANSACTION_DATA_SIGNATURE** must be generated for the merchant.

The block includes the following:

Header

Field	Description
alg	Algorithm (RS256).
kid	ID of the corresponding record in the key_mgmt_certificate table.
typ	Header type (JOSE).
merchantLogin	Login of the merchant.

Payload

Field	Description
mdOrder	This field is reserved for future use.
transDate	This field is reserved for future use.

networkReferenceData	Network reference data returned from the authorization host.
transactionDataElements	Data elements of the authorization request.
isSsl	Specifies whether 3DS2 authentication was not performed for the transaction.
dsTransId	Directory Server Transaction ID, UUID (only for 3DS2 transactions).
acsTransId	Transaction ID in ACS (only for 3DS2 transactions).
initTransactionDate	Initiating transaction date (only for 3DS2 transactions).
version	Constant value of 1. This field is reserved for future use.
panSalt	Base64-encoded 16 bytes random salt (the resulting string will be approximately 24 character long; from **SecureRandom**).
panHash	Base64 SHA-256 hash of byte[] = PAN.getBytes + salt.bytes.

The block validation fails if:

- The field is missing for MIT requests with **recurringPaymentManagedByMerchant=**true or **bindingManagedByMerchant=**true.

- A different **merchantLogin** is used.
- The key provided in the JWS header cannot be found for the current merchant.
- Signature validation fails (both technical and not errors).

4.16.4.2 Creating a recurring payment template

To create a template for the recurring payment without saving the card details to EPG:

1. A merchant initiates the **createRecurringTemplateNoPayment.do** request with **recurringPaymentManagedByMerchant =** true.
2. When authorization is completed, EPG responds to the merchant and passes the **transactionData** block:
   - In a response to the **paymentOrder.do** request only for SSL and Frictionless transactions.
   - Always in a response to the **getOrderStatus.do** (**getOrderStatusExtended.do**) request.

4.16.4.3 Subsequent recurring/installment payment (MIT)

A subsequent recurring (installment) payment is performed as follows:

1. A merchant registers an order with the following parameters:
   - **recurringPaymentForm** — either recurring (for recurring payments) or installment (for installment payments)
   - **recurringExpiry** — specified optionally
   - **recurringPaymentManagedByMerchant =** true — recurring payments are stored and managed by merchants
   - **transactionInitiator =** MIT — transaction is initiated by the merchant
   - **originalTransactionData** that the merchant has received when performed the CIT transaction.
   - **originalInstallmentNumber** — specified if **recurringPaymentForm =** installment
2. The merchant's system triggers the **paymentOrder.do** request.
3. When authorization is completed, EPG responds to the merchant and passes the **transactionData** block:
   - In a response to the **paymentOrder.do** request only for SSL and Frictionless transactions.

- Always in a response to the **getOrderStatus.do** (**getOrderStatusExtended.do**) request.

Note: The **transactionData** must be saved by the merchant if **bindingManagedByMerchant** = true.

4.16.4.4 No-Show payment

Note that merchants must save customer's card credentials using any available method. After that, they will receive the **transactionData** block in a response. For example:

- Creating a binding (saving the card for a non-payment transaction)
- Saving a card during payment or paying with a binding

No-Show payments are performed as follows:

1. A merchant registers an order with the following additional parameter:
   - **bindingManagedByMerchant** = true — bindings are managed by the merchant and EPG is requested not to save the bindings in the system.
   - **transactionInitiator** = MIT — transaction is initiated by the merchant.
   - **bindingUsage** = paymentBySavedCard — card stored during the previous transactions is used in the next transaction
   - **originalTransactionData** that the merchant has received when performed the CIT transaction.

   Note: The **originalTransactionData** value is validated by EPG only if **bindingManagedByMerchant** = true.
2. The merchant's system triggers the **paymentOrder.do** request with **noShowPayment** = true.
3. When authorization is completed, EPG responds to the merchant and passes the **transactionData** block:
   - In a response to the **paymentOrder.do** request only for SSL and Frictionless transactions.
   - Always in a response to the **getOrderStatus.do** (**getOrderStatusExtended.do**) request.

Note: The **transactionData** must be saved by the merchant if **bindingManagedByMerchant** = true.

4.17 P2P methods

P2P payments are payments that are made when transferring funds from one card to another. The card from which funds are transferred is called the *source card* and can belong to either the customer's bank or a third-party bank. The card to which the funds are transferred is called the *receiver card* and must belong to the customer's bank.

Merchant options

To be able to execute P2P methods, the merchant must have at least one of the following permissions enabled depending on the type of P2P transfer being performed (also see the p2p_type parameter):

- Can use 'standard' P2P transfers
- Can use P2P credit operations
- Can use P2P debit operations

4.17.1 Request for P2P order registration

The **p2p/register** method is used to register a P2P order.

Request parameters

Name	Type	Mandatory
userName	AN 1..100	Yes
Login of the API user on whose behalf requests are processed for a specific merchant.		

password	String	Yes
User's password.		
Note: The **userName** and **password** parameters can be passed either as a request URL parameter (see Request structure) or a JSON request body field for **p2p/register.do**. API call ends with an error if any of the two fields is present in both URL parameter and request body.		
amount	N 1..12	Yes
Amount of the funds transfer specified in the minor denomination of the currency.		
currency	N3	Yes
Payment currency in the ISO 4217 format. If it is not specified, the default currency code is used.		
orderNumber	ANS 1..32	No
Number (identifier) of the order in the merchant's online store system. It is unique for every store in the system and is generated on the order registration. If the **Require system to generate order numbers** permission is enabled for the merchant, the order number is generated automatically. Otherwise, the order number data is sent in the API.		

returnUrl	ANS 1..2000	No
URL to which the customer is redirected after a successful payment.		
clientId	ANS...225	No
Customer identifier in the merchant system. This parameter is mandatory for bindings.		
email	ANS...225	No
Customer's email address to which notifications are sent, if notifying customers is enabled.		
externalFee	N..9	No
Fee amount in the minor denomination of the transaction currency. This fee is provided by the external source (for example, an OLTP system). This parameter is transferred if the **3ds.amount.includeExternalFees** system setting is set to true. **Note:** If the merchant has the **Can use both external and internal fees in one transaction** permission, both external fee and internal fees can be provided. The transaction fee will be the sum of the values in fields 28 and 54.		

failUrl	ANS 1..2000	No
Web address to which the customer is redirected if the payment fails.		
language	A2	No
Language code in the ISO 639-1 format. If it is unspecified, SmartVista E-Commerce Payment Gateway uses the default language from the merchant settings. Error messages are also returned in this language.		
orderDescription	ANS...600	No
Description of the order.		
params[]	ANS...1024	No
Fields used to store additional information. The type is as follows: {"param":"value","param2":"value2"} **Note**: The parameter name length is 255 characters or less and the value length is 1024 characters or less. See JSON parameter list for information about which parameters are passed.		

threeDS2Params[]		Conditional
Parameters of the 3-D Secure 2 protocol authentication. The **threeDS2Params** parameter is a JSON-based structure. See 3-D Secure 2 parameter list for more information. Depending on the type of channel interface that is used to initiate the transaction (**deviceChannel**), the parameter is mandatory or optional: · Optional for browser-based authentication · Mandatory for application-based authentication		
sessionTimeoutSecs	N...9	No
Lifespan of the order, in seconds. The order lifespan duration can be taken from the following parameters (from the highest priority to the lowest): · **sessionTimeoutSecs** transferred in a request. It overrides all other order timeout settings. · If the **sessionTimeoutSecs** parameter is not specified, the value from the merchant's settings is used. It is configured by the **Alternative session timeout** option that must be enabled and the additional **Session duration** parameter that must be specified. · If none of the above mentioned settings is specified (neither **sessionTimeoutSecs** nor merchant's timeout), the default value set on the **Administration > System settings** page by the **default.session.timeout.milliseconds** system setting is used. The default value is 1200 seconds. If the request contains the **expirationDate** parameter, the **sessionTimeoutSecs** parameter is ignored.		

sessionExpiredDate	ANS19	No
Date and time when the order is terminated. The following format is used: *yyyy-MM-dd'T'HH:mm:ss* If this parameter is not specified, the **sessionTimeoutSecs** parameter is used to determine the date and time when the order is terminated.		
P2P transfer by the alias	Boolean	No
Specifies whether P2P Visa alias transfers are enabled.		
businessApplicationId	ANS...225	No
Business application identifier used by Visa (see Business application identifiers). It is specified for P2P Visa alias transfers.		
sourceBindingId	AN..255	No
Source binding ID. This is the **bindingID** associated with the debit part of the P2P transaction.		
destinationBindingId	AN..255	No
Destination binding ID. This is the **bindingID** associated with the credit part of the P2P transaction.		

Response parameters

Name	Type	Mandatory
errorCode	N 1..3	Yes
Error code.		
errorMessage	ANS 1..512	Yes
Description of the error in the language that was sent in the **language** parameter of the request.		
orderId	ANS36	No
Unique order number generated by EPG after the registration of the order. This number is absent if the registration of the order failed due to an error (the error is described in the **ErrorCode** field).		
formUrl	ANS...200	No
URL of the card data collection page to which the customer's web browser is redirected. This parameter is not returned if the registration of the order was not successful (the error is described in the **ErrorCode** field).		

orderNumber	ANS 1..32	Yes
Number (identifier) of the order in the merchant's online store system. It is unique for every store in the system and is generated on the order registration. If the **Require system to generate order numbers** permission is enabled for the merchant, the order number is generated automatically. Otherwise, the order number data is sent in the API.		

Error codes (errorCode) and messages (errorMessage)

Code	Message	Description
0	Success.	The request has been processed successfully.
1	Order number is duplicated. The order with the given number has already been processed.	Incorrectly specified order number.
5	Access denied.	The user does not have the necessary permissions to access this resource.
7	System error.	Software or hardware issue or malfunction.

12	**clientId** not equals **clientId** for this binding.	A mismatch of the **Client ID** value specified in the request and **Client ID** required for the binding.
14	Transaction failed.	Transaction processing was not successful. See P2P response codes for the list of failure reasons.

Request example

```
https://<EPG_URL>:<EPG_SSL_Port>/epg/rest/p2p/register?
 userName=<userName>&password=<userPassword>
{
        "amount": 1,
    "currency": "840",
    "returnUrl": "http://127.0.0.1/cart.html",
    "orderNumber": "88878890887535756777"
}
```

For more information, see Request structure.

Request example with an extended parameters list

```
/rest/p2p/register.do?userName={user}&password={password}

{
    "orderNumber": "15742960102892",
        "amount": 999999999999,
```


```
    "currency": "840",
    "returnUrl": "https://127.0.0.1/merchants/m01/finish.html",
            "email": "email@ru.ru",
            "language": "en",
            "params": [
        {"name":"p2p_type", "value":"DEBIT"},
        {"name":"recipientAccountNumber", "value":"12345678901234567890"},
        {"name":"recipientAccountBankCode", "value":"887766554"}
            ]
}
```


Response example


```
{
                    "errorCode": 0,
                    "errorMessage": "Successful",
                    "orderId": "fa55c77a-892a-41fc-bc70-aef4e5f7c8ce",
                    "formUrl": "https://epg.test.in:8443/epg/merchants/testMerchant/
 p2p_payment.html?mdOrder=fa55c77a-892a-41fc-bc70-aef4e5f7c8ce&language=en",
                    "orderNumber": "234443458440189"
}
```


4.17.2 Request for P2P funds transfer

The **p2p/perform** and **p2p/public/perform** methods are used to perform a card-to-card money transfer. The **p2p/public/perform** method is used by third parties integrated with EPG via the EPG APIs.

If the **p2p.verify.mandatory** setting is set to true, verifying the payment data with the P2P transfer verification request is mandatory prior to executing the transfer processing request.

Request parameters

Name	Type	Mandatory
orderId	ANS36	Yes
Unique order number generated by EPG after the registration of the order.		
amountInput	N 1..12	Conditional
Amount of the transaction. If this field is specified, it overrides the amount specified during order registration and will be used instead of that value. The value must be specified in the minor denomination of the currency.		
fromCard[]	ANS...1024	Conditional
block with the details of a card that is the source of the funds transfer (**fromCard**). See the details below. This parameter is mandatory for card-to-card transfers and is optional for transfers from an account to a card (when the **type** field has the CREDIT value). The account data is transferred in **params[]**.		
toCard[]	ANS...1024	Conditional

block with the details of a card that is the recipient of the funds transfer (**toCard**). See the details below. This parameter is mandatory for card-to-card transfers and is optional for transfers from a card to an account (when the **type** field has the DEBIT value). The account data is transferred in **params[]**.		
threeDS2Params[]		Conditional
Parameters of the 3-D Secure 2 protocol authentication. The **threeDS2Params** parameter is a JSON-based structure. See 3-D Secure 2 parameter list for more information. Depending on the type of channel interface that is used to initiate the transaction (**deviceChannel**), the parameter is mandatory or optional: · Optional for browser-based authentication · Mandatory for application-based authentication		
email	ANS...225	No
Email address of the customer.		
ip	ANS...39	No
IP address of the customer.		
language	A2	No

Language code in the ISO 639-1 format. If it is unspecified, SmartVista E-Commerce Payment Gateway uses the default language from the merchant settings.

Error messages are also returned in this language.

params[]	ANS...1024	No

Additional tag with attributes used to pass the additional parameters of a merchant.

The limit is 99 characters, and the following characters cannot be used: %, +, \r (end of a line), and \n (line break).

To pass *N* parameters, the request must contain *N* **Params** tags in which the name attribute contains the name of a parameter and the value attribute contains its value. The tag description is provided below.

type	A...8	No

Type of transfer:

- STANDARD — a standard cart-to-card transfer.
- DEBIT — a transfer without indication of the card to be credited. This option is used to transfer data from a card to an account. In this case, the account data is passed in the **params[]** tag of the **toCard** block.
- CREDIT — a transfer without indication of the card to be debited.

It can be a regular card-to-card transfer, or a debit or credit operation if data on only one card is present. If the value is not specified, it is populated automatically based on whether the card data is present or not (the sender card, recipient card, or both).

clientBrowserInfo	Not applicable	No
JSON structure (encoded as a string) containing the client browser information for 3DS2. See the description of the clientBrowserInfo contents. **Note:** This parameter must be passed as a string, not as a JSON object. All JSON special characters must be escaped according to the JSON specification.		
threeDSCompInd	String	No
Specifies whether the 3DS Method Completion notification was received by the merchant: · Y — the 3DS Method Completion notification from the issuer ACS was received by the merchant using the address specified by the **threeDSMethodNotificationURL** parameter of the **checkPreliminary** method (see Request to check card eligibility for 3DS2 before the main request). · N — the 3DS Method Completion notification from the issuer ACS was not received by the merchant.		
userName	AN 1..30	Yes
Login of the API user on whose behalf requests are processed for a specific merchant.		
password	AN 1..30	Yes

User's password.		
threeDs2ReturnUrl	ANS 1..512	No
Return URL for user redirect from the issuer ACS after 3DS2 authentication is completed. If it is not specified, the user will be redirected to the specific EPG URL.		
aReqFieldsOverride	Not applicable	No
JSON structure (encoded as a string) containing AReq override data for the specific transaction. See the description of the aReqFieldsOverride structure contents. **Note:** This parameter must be passed as a string, not as a JSON object. All JSON special characters must be escaped according to the JSON specification.		
sourceBindingId	AN..255	No
Source binding ID. This is the **bindingID** associated with the debit part of the P2P transaction.		
destinationBindingId	AN..255	No

Destination binding ID. This is the **bindingID** associated with the credit part of the P2P transaction.		
bindingNotNeeded	Boolean	No
Binding settings: · true — disables the creation of a binding after a successful payment (the client identifier sent with the order registration request is deleted from the order details after the payment). · false — a successful payment may result in the creation of a binding (under the relevant conditions). This is the default value.		
Note: The **userName** and **password** parameters are used only for the **p2p/public/perform** request. These parameters are passed only in the JSON request body.		

Format of the fromCard block

Name	Type	Mandatory
pan	N 13...19	Yes
Card number.		
expirationYear	N4	Yes

Year when the card validity period expires, in the following format: *YYYY*.		
expirationMonth	N2	Yes
Month when the card validity period expires, in the following format: *MM*.		
cvc	N3..4	Conditional
Card security code: CVV2 (Card Verification Value 2), CVC2 (Card Verification Code 2) and analogous codes. Whether this parameter is mandatory for payments that use the respective payment method, depends on the **Can pay by binding without CVV2/CVC2** and **Can pay by card without CVV2/CVC2** options: · If the option is selected for a merchant, this parameter is optional for this merchant. · If the option is not selected, this parameter is mandatory for the merchant. The options are configured through the administrative portal UI.		
cardholderName	ANS 2..26	No

Name of the cardholder. This parameter is verified according to the following criteria: · Acceptable characters are: Latin letters, 0-9, $,), (, -, . , a space · Cardholder name must start with a letter · Minimum length: 2 characters · Maximum length: 26 characters · Null is valid · Uppercase and lowercase are acceptable		
threeDsAuthResult[]	Not applicable	No
Tag with the result of the 3-D Secure authorization. The parameters of the threeDsAuthResult tag are described below.		

Format of the threeDsAuthResult tag

Name	Type	Mandatory
cavv	ANS..200	No
Cardholder Authentication Verification Value.		
xid	ANS..80	No
Electronic Commerce Transaction Identifier.		

eci	AN2	No
Electronic Commerce Indicator.		
tdsProtocol	String	No
3DS Procotol version used by the external MPI/3DSS. Supported values: 1.0.2, 2.1.0, 2.2.0.		
dsTransID	JWS	No
Directory Server Transaction ID, UUID.		
threeDSServerTransID	ANS36	Conditional
3DS Server Transaction ID, UUID.		

Note: To specify that the request contains 3DS authentication results received from external MPI/3DSS, the P2P Transfer request must have **EXTERNAL_MPI_RES=**true in the **params[]** block.

Format of the toCard block

Name	Type	Mandatory
pan	N 13...19	Yes
Card number.		
expirationYear	N4	No
Year when the card validity period expires, in the following format: *YYYY*.		
expirationMonth	N2	No
Month when the card validity period expires, in the following format: *MM*.		
cvc	N3..4	No
Card security code: CVV2 (Card Verification Value 2), CVC2 (Card Verification Code 2) and analogous codes. Whether this parameter is mandatory for payments that use the respective payment method, depends on the **Can pay by binding without CVV2/CVC2** and **Can pay by card without CVV2/CVC2** options: · If the option is selected for a merchant, this parameter is optional for this merchant. · If the option is not selected, this parameter is mandatory for the merchant. The options are configured through the administrative portal UI.		

cardholderName	ANS 2..26	No
Name of the cardholder. This parameter is verified according to the following criteria: · Acceptable characters are: Latin letters, 0-9, $,), (, -, . , a space · Cardholder name must start with a letter · Minimum length: 2 characters · Maximum length: 26 characters · Null is valid · Uppercase and lowercase are acceptable		
threeDsAuthResult[]	Not applicable	No
Tag with the result of the 3-D Secure authorization. The parameters of the threeDsAuthResult tag are described below.		

Format of the threeDsAuthResult tag

Name	Type	Mandatory
cavv	ANS..200	No
Cardholder Authentication Verification Value.		
xid	ANS..80	No

Electronic Commerce Transaction Identifier.		
eci	AN2	No
Electronic Commerce Indicator.		

Format of the params tag

Name	Type	Mandatory
name	ANS..255	Yes
Name of an additional parameter.		
value	ANS..1024	Yes
Value of the additional parameter.		

The following customer parameters can be passed optionally in the **params** tags within a request for a card-to-card money transfer (the format of values is determined by processing).

Note: These data elements can also be specified in the **fromCard[]** and **toCard[]** blocks.

Sender's parameters

The parameters described below are applicable for P2P Transfer and P2P Debit transactions.

Parameter	Type	Parameter system name
payerCity	AN..40	**p2p.city.length**
Payer's city name.		
payerCountry	AN..3	**p2p.country.mask**
Alpha-3 ISO 3166 code of the sender's country.		
payerAddress	ANS..99	**p2p.address.length**
Payer's address.		
payerPostalCode	ANS..10	**p2p.postalCode.length**
Postal code in the payer's address.		
payerState	ANS..99	**p2p.state.length**
State code in the payer's address.		

Note: This is applicable for USA and CANADA remittance transactions.		
payerPhone	ANS..20	**p2p.phone.length**
Payer's phone number.		
payerDateOfBirth	ANS..20	**p2p.dateOfBirth.mask**
Payer's date of birth. The format of the field is *YYYYMMDD*.		
payerIdType	AN..8	**p2p.idType.mask**
Type of the identification document provided by the payer: · IDTP1 — Passport · IDTP2 — Driving license · IDTP3 — Social · IDTP4 — Citizen ID · IDTP5 — VAT certificate of registration · IDTP6 — Refugee certificate · IDTP7 — Residence permit · IDTP8 — Transborder Passport · IDTP9 — Official Passport · IDTP10 — Temporary Passport · IDTP11 — Seaman's Passport If **cardholderIDType** must be transferred in the transaction, one of eleven options should be available to indicate the appropriate type. Only one ID can be specified.		

payerIdNumber	AN..99	**p2p.idNumber.length**
The number of the provided identification document. Required if the **cardholderIDType** has been defined.		
payerIdExpiration	ANS..20	**p2p.idExpirationDate.mask**
The expiration date of the provided identification document, in the following format: *YYYYMMDD*. Required if the **cardholderIDType** has been defined.		
payerIdCountry	AN..3	**p2p.country.mask**
ISO 3166 code of the country where the identification document was issued. Required if the **cardholderIDType** has been defined.		
payerLastName	ANS..20	**p2p.lastName.length**
Payer lats name.		
payerFirstName	ANS..20	**p2p.firstName.length**
Payer first name.		
payerMiddleName	ANS..20	**p2p.middleName.length**

Payer middle name.		
payerSuffixName	AN..20	**p2p.suffixName.length**
Payer suffix.		
payerName	AN..99	**p2p.name.length**
Payer combined name.		

Recipient's parameters

The parameters described below are applicable for P2P Transfer and P2P Credit transactions.

Parameter	**Type**	**Parameter system name**
recipientCity	AN..40	**p2p.city.length**
Recipient's city name.		
recipientCountry	AN..3	**p2p.country.mask**
Alpha-3 ISO 3166 code of the recipient's country.		

recipientAddress	ANS..99	**p2p.address.length**
Recipient’s address.		
recipientPostalCode	ANS..10	**p2p.postalCode.length**
Postal code in the recipient’s address.		
recipientState	ANS..99	**p2p.state.length**
State code in the recipient’s address. **Note:** Applicable for USA and CANADA remittance transactions.		
recipientPhone	ANS..20	**p2p.phone.length**
Recipient’s phone number.		
recipientDateOfBirth	ANS..20	**p2p.dateOfBirth.mask**
Recipient’s date of birth. The format of the field is *YYYYMMDD*.		
recipientIdType	AN..8	**p2p.idType.mask**

Type of the identification document provided by the recipient:

- IDTP1 — Passport
- IDTP2 — Driving license
- IDTP3 — Social
- IDTP4 — Citizen ID
- IDTP5 — VAT certificate of registration
- IDTP6 — Refugee certificate
- IDTP7 — Residence permit
- IDTP8 — Transborder Passport
- IDTP9 — Official Passport
- IDTP10 — Temporary Passport
- IDTP11 — Seaman's Passport

If **cardholderIDType** must be transferred in the transaction, one of eleven options should be available to indicate the appropriate type. Only one ID can be specified. Only one ID can be specified.

recipientIdNumber	AN..99	**p2p.idNumber.length**
Number of provided identification document. Required if **cardholderIDType** was defined.		
recipientIdExpiration	ANS..20	**p2p.idExpirationDate.mask**
Expiration date of the provided identification document, in the following format: *YYYYMMDD*. Required if **cardholderIDType** was defined.		
recipientIdCountry	AN..3	**p2p.country.mask**

ISO 3166 code of the country where the identification document was issued. Required if **cardholderIDType** was defined.		
recipientLastName	ANS..20	**p2p.lastName.length**
Recipient's last name.		
recipientFirstName	ANS..20	**p2p.firstName.length**
Recipient's first name.		
recipientMiddleName	ANS..20	**p2p.middleName.length**
Recipient's middle name.		
recipientSuffixName	AN..20	**p2p.suffixName.length**
Recipient's suffix.		
recipientName	AN..99	**p2p.name.length**
Recipient's combined name.		

Response parameters

Name	Type	Mandatory
errorCode	N 1..3	Yes
Error code.		
errorMessage	ANS 1..512	Conditional
Description of the error in the language that was sent in the **language** parameter of the request. This parameter is returned in case of an error.		
acsUrl	AN..512	No
URL of the ACS server. This parameter is used in payments that require additional authentication on the issuing bank's ACS. This parameter is returned in case a successful request processing.		
additionalResponseCodes	AN..1024	Conditional
Result of the cardholder billing address check. This parameter is transferred in case the AVS check is used (the **AVS enabled** option must be enabled for the merchant). The additionalResponseCodes block contents are described below.		

This parameter is returned when the request is successfully processed.		
cReq	ANS..512	Conditional
Challenge Request message. EMV 3-D Secure message sent by the 3DS SDK or 3DS Server where additional information is sent from the cardholder to the ACS to support the authentication process. It must be present for 3-D Secure 2 if a cardholder challenge is required. This parameter is returned when the request is successfully processed.		
info	ANS..512	Conditional
The result of a money transfer attempt. The following message is displayed in the case of a successful transfer: Your payment has been processed, redirecting... The following message is displayed in the case of an error: Redirecting... This parameter is returned when the request is successfully processed.		
paReq	ANS..512	Conditional
Payer Authentication Request is a message sent from the MPI to ACS via the cardholder device. PAReq requests the issuer to authenticate its cardholder and contains the cardholder, merchant, and transaction-specific information necessary to perform authentication. It is used in 3-D Secure 1. This parameter is not used in payments that do not require additional authentication on the issuing bank's ACS.		

This parameter is returned when the request is successfully processed.		
redirect	ANS..512	Conditional
URL to which the customer is redirected after executing the payment, depending on the payment result. This parameter is returned when the request is successfully processed.		
termUrl	ANS..512	Conditional
Return address from ACS for the customer to complete the payment. This parameter is used in payments that require additional authentication on the issuing bank's ACS. This parameter is returned when the request is successfully processed.		
processingErrorType	Not applicable	Yes
Block containing details on processing errors if they occur. This is for non-P2P transactions.		
value	AN..	No
Processing error message value.		
messageCode	AN..	No

Processing error message code.

Format of the additionalResponseCodes tag

Name	Type	Mandatory
type	AN..3	No
Type of additional response code. The only available value is AVS.		
responseCode	A1	No
AVS response code.		
responseMessage	AN..512	No
AVS response message.		

Error codes (errorCode) and messages (errorMessage)

Code	Message	Description
0	Success.	The request has been processed successfully.

12	**clientId** not equals **clientId** for this binding.	A mismatch of the **Client ID** value specified in the request and **Client ID** required for the binding.
14	Transaction failed.	Transaction processing was not successful. See P2P response codes for the list of failure reasons.

Request example

```
https://<EPG_URL>:<EPG_SSL_Port>/epg/rest/p2p/perform
{
"orderId": "26c5861f-29dd-4e2e-b952-e4fa630a29cb",
"language": "en",
"amountInput": "3000",
"fromCard":
\{
"pan": "4400000031380700",
"cvc": "337",
"expirationYear": "2021",
"expirationMonth": "8",
"cardholderName": "TEST"
}
,
"toCard":
{
"pan": "4400000041520600"
},
"type": "STANDARD"
}
```

For more information, see Request structure.

Example of p2p/public/perform request including aReqFieldsOverride


```
"userName": "3dsmerchant_api",
    "password": "3dsmerchant_api1",
    "orderId": "8bb55cc2-e918-43f7-b14c-7d8635963f5e",
            "type": "STANDARD",
    "threeDSCompInd": "Y",
    "threeDs2ReturnUrl": "https://my-site.com/p2p/completed?orderNumber=
                                3dsmerchant_1690290799923",
    "fromCard": {
        "pan": "4000010000000001",
        "cvc": "123",
        "expirationYear": 2049,
        "expirationMonth": 12,
        "cardholderName": "AGENT SMITH"
            },
            "toCard": {
        "pan": "4000010000000019"
            },
    "clientBrowserInfo": "{\"userAgent\":\"Mozilla/5.0(Macintosh;
     IntelMacOSX10.15;rv:101.0)Gecko/20100101Firefox/101.0\",\"os\":\"MacOS\",
            ...
     \"browserLanguage\":\"it-IT\",\"browserTimeZone\":\"Europe/Rome\",
            \"browserTimeZoneOffset\":-120}",
    "aReqFieldsOverride": "{\"mcc\":\"4814\",\"threeDSRequestorURL\":
     \"https://e-wallet.com\",\"acquirerMerchantID\":\"e-wallet-providerId\",
      \"threeDSRequestorName\":\"EWallet 3DS Requestor\",\"merchantName\":
       \"EWallet\",\"merchantCountryCode\":\"643\"}"
}
```


Response example


```
{
                    "errorCode": 0,
                    "info": "Your order is proceeded, redirecting...",
                    "acsUrl": "https://epg.test.in:8443/acs/pareq/aa3bc97a81974c559385a74a0c943588"
```

"paReq": "eJxVUk1TwjAQ/StM7yVp+hWYJQ5aHTkUHcWDx5huIWpLSVss/
noTRMXbe7s7723eBi6G6n20R9PqbT3zgjH1RlirbaHr9cx7Wt343LsQsNoYxOwRVW9QQI5tK9c40sXMU
+WESY6Fn2Ip/SgtY59PwthnKY9eEqViybgn4H7+gDsBJyNhfcYMyA+1ikZtZN0JkGp3uViKKA5ZmgI5U
ajQLDJhC98Aalmh6LDtZKOBHBmobV935iCSKATyQ6A372LTdU07JaSR5s1v0ex1OzY9ENcD8md+3zvUW
q1BF2KZzT/y13WUZ9csz9TnMnselqt1dJc9z4C4CShkh4JRRgNG01EQTimbhhzIsQ6yckuIkFJqX/JNo
HEe8/POeQVswsYe4CB4ZFu/DHBotjXaCRvbLwbyt/HVrQtPdTadIJjESclDmuiq0ZOnPi9xoLpcsNc3F
+lxyClqGxALKD9KOgLEyZDTtcjp0Bb9+wBfvSC5Tg==",
"termUrl": "https://epg.test.in:8443/epg/rest/finish3ds.do"
}

4.17.3 Request for P2P payment status

The **p2p/getFinishedPaymentInfo.do** method is used to get the status of a P2P money transfer.

Request parameters

Name	Type	Mandatory
userName	AN 1..100	Yes
Login of the API user on whose behalf requests are processed for a specific merchant.		
password	String	Yes
User’s password.		

orderId	ANS36	Yes
Unique order number generated by EPG after the registration of the order.		

Response parameters

Name	Type	Mandatory
merchantShortName	AN 1..100	Yes
Merchant login in the EPG system.		
authorizationResponseId (the deprecated name is **approvalCode**)	AN6	No
Payments network authorization code. The field has a fixed length of six characters; it can contain both numbers and letters.		
OrderNumber	AN 1..32	Yes
Number (identifier) of the order in the merchant's online store system. It is unique for every store in the system and is generated on the order registration. If the **Require system to generate order numbers** permission is enabled for the merchant, the order number is generated automatically. Otherwise, the order number data is sent in the API.		

backUrl	AN..200	No
URL to which the customer is redirected after a successful payment.		
failUrl	ANS..200	No
Web address to which the customer is redirected if the payment fails.		
terminalId	ANS..8	Yes
Card Acceptor Terminal Identification. It is a unique ID of the merchant's terminal.		
orderDescription	AN..512	No
Free form description of the order.		
merchantFullName	AN..200	Yes
Full name of the merchant.		
currency	N3	Yes
Payment currency code in the ISO 4217 format.		

formattedAmount	N 1..19	Yes
Formatted order registration amount (with a dot separating the minor currency units).		
originalActionCodeDescription	AN..512	Yes
Custom description of original action code for SV IPS Payment Ways.		
actionCode	N 1..6	Yes
Processing response code.		
actionCodeDescription	AN..512	Yes
Processing response message.		
formattedFeeAmount	N 1..19	Yes
Formatted order processing fee (with a dot separating the minor currency units).		
amount	N 1..19	Yes
Order amount in the minor denomination (for example, cents).		

panMasked	N 13...19	No
Masked number of the payment card.		
successUrl	AN..512	Yes
URL to which the customer is redirected after a successful payment.		
p2p.formattedFeeAmount	N 1..19	Yes
Formatted order processing fee (with a dot separating the minor currency units) in case of a P2P transaction.		
feeAmount	N 1..19	Yes
Order processing fee.		
orderParams	Not applicable	No
block with additional order parameters.		
refNum	AN..24	No
Reference number.		

paymentDate	UTC	No
Date of the order registration.		
merchantUrl	AN..512	Yes
URL of the merchant site (online store).		
status	String	Yes
Status of the current payment.		

Error codes (errorCode) and messages (errorMessage)

Value	Description	Description
0	Success.	The request has been processed successfully.
5	Access denied.	The user does not have the necessary permissions to access this resource.
7	System error.	Software or hardware issue or malfunction.

14	Transaction failed.	Transaction processing was not successful. See P2P response codes for the list of failure reasons.

Request example

```
http://wls01-ecom.bpc.in/epg/rest/p2p/getFinishedPaymentInfo.do?orderId=
 a2586997-80ab-4e2a-a393-7310904f3201&language=en
```

Response example


```
{"merchantShortName":"testapi","approvalCode":"null","orderNumber":"88878890887535756777",
 "backUrl":"http://127.0.0.1/cart.html","failUrl":"http://127.0.0.1/cart.html","terminalId":
 "10000003":"orderDescription":"","merchantFullName":"test_api","currency":"840",
 "formattedAmount":"0.01","actionCodeDescription":"Payment declined.
 Please, contact with merchant.","formattedFeeAmount":"0.00","amount":"1","panMasked":"",
 "successUrl":"http://127.0.0.1/cart.html","p2p.formattedFeeAmount":"0.00","feeAmount":"0",
 "orderParams":{},"refNum":"null","paymentDate":"2019-03-26 14:22:58.226",
 "merchantUrl":"http://google.com","status":"DECLINED"}
```


4.17.4 Request for P2P transfer verification

The **p2p/verify** and **p2p/public/verify** requests are used to check whether a P2P transfer is possible before making the transfer. The **p2p/public/verify** method is used by third parties integrated with EPG via the EPG APIs. In case of a successful response these requests return the transfer fee details.

This request relates to the following system settings configured on the **Administration > System settings** page of the EPG administration portal:

- **p2p.verify.mandatory** — when this setting is set to true, the transfer verification request must be called to check the data of a P2P money transfer before executing the transfer processing request.
- **3ds2.amount.includeFees** — when this setting is set to true, it enables including the transaction fee in the **purchaseAmount** field of the AReq message (for the 3-D Secure 2 protocol).
- **3ds.amount.includeExternalFees** — when this setting is set to true, it enables including the fee provided by the external source (for example, an OLTP system) in the authorization request amount for both protocols 3-D Secure 1 (PaReq) and 3-D Secure 2 (AReq).
- **p2p.anonymous.amount.change.allowed** — when this setting is set to false, it disables changing the transaction amount for the following P2P methods: perform.do or verify.do for the web API; performP2P or verifyP2P for the SOAP API. After the transaction verification request is executed, the amount cannot be changed regardless of this setting.

Request parameters

Name	Type	Mandatory
orderId	ANS36	Yes
Unique order number generated by EPG after the registration of the order.		
amount	N 1..12	Conditional

Amount of the transaction. If this field is specified, it overrides the amount specified during order registration and will be used instead of that value. The value must be specified in the minor denomination of the currency. **Note:** The amount field is required if the **p2p.anonymous.amount.change.allowed** setting on the **Administration > System settings** page is set to true. The default setting value is false, which means the ability to change the amount is disabled.		
fromCard[]	ANS...1024	No
block with the details of a card that is the source of the funds transfer (**fromCard**). See the details below.		
toCard[]	ANS...1024	No
block with the details of a card that is the recipient of the funds transfer (**toCard**). See the details below.		
externalFee	N..9	No

Fee amount in the minor denomination of the transaction currency. This fee is provided by the external source (for example, an OLTP system).

This parameter is transferred if the **3ds.amount.includeExternalFees** system setting is set to true.

Note: External fees are not supported simultaneously with internal fees calculated in the system. Only one type of fee, either external or internal, can be used at a time.

language	A2	No
Language code in the ISO 639-1 format. If it is unspecified, SmartVista E-Commerce Payment Gateway uses the default language from the merchant settings. Error messages are also returned in this language.		
userName	AN 1..30	Yes
Login of the API user on whose behalf requests are processed for a specific merchant.		
password	AN 1..30	Yes
User’s password.		

sourceBindingId	AN..255	No
Source binding ID. This is the **bindingID** associated with the debit part of the P2P transaction.		
destinationBindingId	AN..255	No
Destination binding ID. This is the **bindingID** associated with the credit part of the P2P transaction.		
Note: The **userName** and **password** parameters are used only for the **p2p/public/verify** request. These parameters are only passed in the JSON request body.		

Format of the fromCard[] and toCard[] blocks

Name	Type	Mandatory
pan	N 13...19	No
Number of the card to be debited (for the **fromCard[]** block) or of the card to be credited (for the **toCard[]** block).		
expirationYear	N..4	No
Year when the card validity period expires, in the following format: *YYYY*.		

The accepted values are from 2000 to 2200.		
expirationMonth	N..2	No
Month when the card validity period expires, in the following format: *MM*. The following values are accepted: 01, 02, 03, 04, 05, 06, 07, 08, 09, 10, 11, 12.		
cardholderName	ANS 2..26	No
Cardholder name. It is only specified for paid orders.		
threeDsAuthResult[]	Not applicable	No
Tag with the result of the 3-D Secure authorization. The parameters of the **threeDsAuthResult** tag are described below.		

Format of the threeDsAuthResult tag

Name	Type	Mandatory
cavv	ANS..200	No
Cardholder Authentication Verification Value.		

xid	ANS..80	No
Electronic Commerce Transaction Identifier.		
eci	AN2	No
Electronic Commerce Indicator.		

Response parameters

Name	Type	Mandatory
errorCode	N 1..3	Yes
Error code.		
errorMessage	ANS 1..512	Yes
Description of the error.		
feeDescriptionList[]	ANS...1024	Yes
block with the details about fees charged in the scope of the transaction.		

formattedAmount	N 1..19	Yes
Formatted order processing amount (with a dot separating the minor currency units) in case of a P2P transaction.		
feeFormattedAmount	N 1..19	Yes
Formatted order processing fee (with a dot separating the minor currency units) in case of a P2P transaction.		
acquirerFeeFormattedAmount	N 1..19	Yes
Formatted acquirer fee amount. This field is always available in the response irrespective of the **p2pDisplayAcquirerIssuerFees** value in the **getSessionStatus.do** request for P2P orders.		
issuerFeeFormattedAmount	N 1..19	Yes
Formatted issuer fee amount. This field is always available in the response irrespective of the **p2pDisplayAcquirerIssuerFees** value in the **getSessionStatus.do** request for P2P orders.		
totalFormattedAmount	N 1..19	Yes
Formatted total order processing amount (with a dot separating the minor currency units) in case of a P2P transaction.		

Format of the feeDescriptionList[] block

Name	Type	Mandatory
feeAmount	N 1..19	No
Fee amount in the minor denomination of the currency.		
feeCurrency	N3	No
ISO 4217 code of the fee currency.		
feeDescription	AN..512	No
Description of the fee.		

Error codes (errorCode) and messages (errorMessage)

Code	Message	Description
0	Success.	The request has been processed successfully.
5	Operation failed.	Failed to send the request.

7	System error.	Software or hardware issue or malfunction.
12	**clientId** not equals **clientId** for this binding.	A mismatch of the **Client ID** value specified in the request and **Client ID** required for the binding.
14	Transaction failed.	Transaction processing was not successful. See P2P response codes for the list of failure reasons.

Request example

```
https://<EPG_URL>:<EPG_SSL_Port>/epg/rest/p2p/verify
 {
                    "orderId": "d31111be-fa99-4ba4-9f1b-5b868c825797",
                    "amount": "3000",
                    "fromCard":
                    {
                              "pan": "4400000041520600"
                    }

                    ,
                    "toCard":
                    {
                              "pan": "4400000031380700"
                    }
```

```
}
```

For more information, see Request structure.

Response example

```
{
    "errorCode": 0,
    "errorMessage": "Successful",
    "feeDescriptionList": [
        {   "feeAmount": 111,
        "feeCurrency": "840",
        "feeDescription": "External fee"
        }
        ],

    "formattedAmount": "USD 10.00",
    "feeFormattedAmount": "USD 1.11",
    "acquirerFeeFormattedAmount": "USD 0.00",
    "issuerFeeFormattedAmount": "USD 0.00",
    "totalFormattedAmount": "USD 11.11"

}
```

The **p2p/alias/resolve** method is used to send a request for a P2P transfer to the recipient's alias.

The **p2p/perform.do** method is used to initiate a P2P transfer using the alias.

Request parameters

Name	Type	Mandatory
alias	AN 1..320	Yes
Recipient’s alias.		
aliasType	AN 1..12	No
Type of alias: · EMAIL_ID · PHONE_NUMBER · NATIONAL_ID · IBAN		
clientBrowserInfo	Not applicable	No
JSON structure containing the client browser information for 3DS2. See the description of the clientBrowserInfo structure contents.		

Request example

```
{
   "orderId": "1234567890",
   "email": "",
   "fromCard": {
            "pan": "4400000041520600"
   },
   "toAlias":{
            "alias": "m@mail.ru"
```

```
    "aliasType": "EMAIL_ID"
  }
}
```

Response example

```
{
 {"bankName":"Test Bank 1","recipientName":"John K Kamau"}
 }
```

4.17.5 Request for P2P transfer verification without registration

The **verify** (**p2p/unregistered/verify**) method is used for a merchant to check a P2P transaction and the fee for it without registering an order.

Request parameters

Name	Type	Mandatory
userName	AN 1..100	Yes
Login of the API user on whose behalf requests are processed for a specific merchant. **Note:** This parameter is passed as request URL parameter, see Request structure.		

password	String	Yes
User's password. **Note:** This parameter is passed as request URL parameter, see Request structure.		
amount	N 1..12	Yes
Transaction amount.		
currency	N3	Yes
Payment currency code in the ISO 4217 format.		
fromCard[]	ANS...1024	No
block with the details of a card that is the source of the funds transfer (**fromCard**). See the details below.		
toCard[]	ANS...1024	No
block with the details of a card that is the recipient of the funds transfer (**toCard**). See the details below.		

externalFee	N..9	No
Fee amount in the minor denomination of the transaction currency. This fee is provided by the external source (for example, an OLTP system). This parameter is transferred if the **3ds.amount.includeExternalFees** system setting is set to true. **Note:** External fees are not supported simultaneously with internal fees calculated in the system. Only one type of fee, either external or internal, can be used at a time.		
language	A2	No
Language code in the ISO 639-1 format. If it is unspecified, SmartVista E-Commerce Payment Gateway uses the default language from the merchant settings. Error messages are also returned in this language.		

Format of the fromCard[] and toCard[] blocks

Name	Type	Mandatory
pan	N 13...19	Yes
Number of the card to be debited (for the **fromCard[]** block) or of the card to be credited (for the **toCard[]** block).		

expirationYear	N..4	No
Year when the card validity period expires, in the following format: *YYYY*. The accepted values are from 2000 to 2200.		
expirationMonth	N..2	No
Month when the card validity period expires, in the following format: *MM*. The following values are accepted: 01, 02, 03, 04, 05, 06, 07, 08, 09, 10, 11, 12.		
cardholderName	ANS 2..26	No
Cardholder name. It is only specified for paid orders.		
threeDsAuthResult[]	Not applicable	No
Tag with the result of the 3-D Secure authorization. The parameters of the **threeDsAuthResult** tag are described below.		

Format of the threeDsAuthResult tag

Name	Type	Mandatory
cavv	ANS..200	No

Cardholder Authentication Verification Value.		
xid	ANS..80	No
Electronic Commerce Transaction Identifier.		
eci	AN2	No
Electronic Commerce Indicator.		

Response parameters

Name	Type	Mandatory
errorCode	N 1..3	Yes
Error code.		
errorMessage	ANS 1..512	Yes
Description of the error.		
feeDescriptionList[]	ANS...1024	Yes

block with the details about fees charged in the scope of the transaction.

Format of the feeDescriptionList[] block

Name	Type	Mandatory
feeAmount	N 1..19	No
Fee amount in the minor denomination of the currency.		
feeCurrency	N3	No
ISO 4217 code of the fee currency.		
feeDescription	AN..512	No
Description of the fee.		

Error codes (errorCode) and messages (errorMessage)

Code	Message	Description
0	Success.	The request has been processed successfully.

5	Operation failed.	Failed to send the request.
7	System error.	Software or hardware issue or malfunction.
14	Transaction failed.	Transaction processing was not successful. See P2P response codes for the list of failure reasons.

Request example

```
https://<EPG_URL>:<EPG_SSL_Port>/epg/rest/p2p/unregistered/verify?
 userName=<userName>&password=<userPassword>
                        {
                        "language":"EN",
                        "amount": "3000",
                        "currency": "840",
                        "fromCard":\
                        {
                                "pan": "4400000041520600"
                        }
                        ,
                        "toCard":
                        {
                                "pan": "4400000031380700"
                        }
 }
```

For more information, see Request structure.

Response example


```
{
                "errorCode": 0,
                "errorMessage": "Successful",
                "feeDescriptionList":[\
                {
                        "feeAmount": 5,
                        "feeCurrency": "810",
                        "feeDescription": "Issuer fee"
                }]
 }
```


4.17.6 Request for information about P2P transaction

The **p2p/info** request is used to get the information about a P2P transaction.

Request parameters

Name	Type	Mandatory
orderId	ANS36	Yes
Unique order number generated by EPG after the registration of the order.		
language	A2	No
Language code in the ISO 639-1 format. If it is unspecified, SmartVista E-Commerce Payment Gateway uses the default language from the merchant settings.		

Error messages are also returned in this language.

Response parameters

Name	Type	Mandatory
actionCodeDescription	ANS...33	Yes
Text description of the processing code.		
amount	N 1..12	Yes
Payment amount in cents.		
amountFormatted	NS 1..13	Yes
Formatted amount with a decimal point.		
authorizationResponseId	AN6	Yes
Authorization Response identifier of the statement transaction.		
backUrl	ANS...200	Yes

Landing page for the cardholder's redirection after a payment's completion. This parameter is the same as **returnUrl** from the order registration request.

The URL is a composed value, which contains the following parameter values from the original request:

- **Scheme**
- **ServerName**
- **ServerPort**
- **ContextPath**
- List of merchants
- merchant **finishPageLogin** or the value from **default.payment.page.login** (if **finishPageLogin** was not specified)
- /finish.html with an appended **orderId**
- **lang** (for transactions with the specified language)
- **status** (for transactions with the transaction state)

Depending on the transaction result, the URL is as follows:

- If the transaction is successful or **failUrl** was not specified either of the following is returned:
  - The original **returnUrl** value if the **Show finish payment page** option is enabled for the merchant
  - The **returnUrl** value with an appended **orderId** parameter.
- If the transaction failed either of the following is returned:
  - The **failUrl** value if it was specified or (if there is no **failUrl**) the **returnUrl** value with an appended **orderId** parameter, if the **Show finish payment page** option is disabled for the merchant.
  - The **failUrl** value if it was specified or (if there is no **failUrl**) the **returnUrl**, but without an appended **orderId** parameter if the request attributes are presented and they are not a value from **ServletRequestAttributes**.

cardholderName	ANS 2..26	Yes

Cardholder name.		
currency	N3	Yes
Numeric currency code.		
currencyName	A3	Yes
Payment currency alphabetic code in the ISO 4217 format.		
date	ANS 1..20	Yes
Payment date in the following format: *MM*/*d*/*yy HH*:*MM*.		
emailEnabled	Boolean	Yes
Specifies whether the email notifications are enabled or not for the current order.		
emailRegExp	ANS...2000	Yes
Regular expression to check the format of the email address entered on the order payment page. The value is the same as that for the **emailMask** system setting on the **Administration > System settings** page.		
expiry	NS7	Yes

Card expiration date in the following format: *YYYY*/*MM*.		
failUrl	ANS..200	Yes

Landing page for the cardholder's redirection after a payment failure. This parameter is the same as **returnUrl** from the order registration request or to **failUrl** if it is present in the registration request.

The URL is a composed value, which contains the following parameter values from the original request:

- **Scheme**
- **ServerNam**
- **ServerPor**
- **ContextPath**
- List of merchants
- merchant **finishPageLogin** or the value from **default.payment.page.login** (if **finishPageLogin** was not specified)
- /finish.html with an appended **orderId**
- **lang** (for transactions with the specified language)
- **status** (for transactions with the transaction state).

Depending on the transaction result, the URL is as follows:

- The **failUrl** value if it was specified or (if there is no **failUrl**) the **returnUrl** value with an appended **orderId** parameter, if the **Show finish payment page** option is disabled for the merchant.
- The **failUrl** value if it was specified or (if there is no **failUrl**) the **returnUrl**, but without an appended **orderId** parameter if the request attributes are presented and they are not a value from **ServletRequestAttributes**.

feeAmount	N...9	Yes
Fee amount in the minor currency units.		
formattedFeeAmount	NS...10	Yes
Formatted fee amount with a delimiter.		
merchantFullName	ANS...25	Yes
Merchant name.		
merchantShortName	ANS...100	Yes
Merchant short name. It is the same value as the merchant login, and EPG validates it at the merchant creation stage. The value must match the **merchantLoginMask** system setting. The default **merchantLoginMask** value is ^(?!.\s).$.		
merchantUrl	ANS...200	Yes
Merchant URL used during the merchant registration in EPG.		
status	A...9	Yes

Order status: · APPROVED · CREATED · DECLINED · DEPOSITED · REFUNDED · REVERSED		
paymentDate	ANS23	Yes
Payment date in the following format: *YYYY-MM-DD HH:MM:SS.sss*.		
orderDescription	ANS...600	Yes
Free form description of the order.		
orderNumber	ANS...32	Yes
Number (identifier) of the order in the merchant's online store system. It is unique for every store in the system and is generated on the order registration. If the **Require system to generate order numbers** permission is enabled for the merchant, the order number is generated automatically. Otherwise, the order number data is sent in the API.		
orderParams	ANS...1024	Yes

Additional order parameters collected at the order registration step and the card entry step. The value is empty if additional order parameters were not used.		
panMasked	NS...9	Yes
Masked PAN used for the order. It is empty for orders in the CREATED state.		
refNum	N..12	Yes
Reference number from the processing system. It is empty for orders in the CREATED state.		
successUrl	ANS...200	Yes
Landing page for the cardholder. This URL is the same as **backUrl**.		
terminalId	ANS..8	Yes
Terminal identifier used in the communication with the processing system. EPG validates the value of this parameter at the merchant creation stage. The value must meet the criteria set by the regular expression in the **merchant.terminalId.pattern** system setting on the **Administration > System settings** page. The default value is ^([A-Za-z0-9]) {8}$.		
debitPanMasked	ANS...19	Conditional

Masked number of a card that is the source of the funds to transfer. This parameter is returned for transactions with the **Payment way** parameter (payment method) set to P2P.		
digest[]	ANS...1024	Conditional
Encoded string. Based on the transaction status, it can be the following parameters: · **formattedAmount** · **currency** · **authorizationResponseId** · **orderNumber** · **refNum** · **paymentDate** · **formattedFeeAmount** · **secretToken** This parameter is returned for merchants that have secret tokens.		
creditPanMasked	ANS...19	Conditional
Masked number of a card that is the recipient of the funds. This parameter is returned for transactions with the **Payment way** parameter (payment method) set to P2P.		
p2p.creditPan	ANS...19	Conditional

Masked number of a card that is the recipient of the funds. This parameter is returned for transactions with the **Payment way** parameter (payment method) set to P2P.		
p2p.formattedFeeAmount	NS...10	Conditional
Formatted fee amount with a delimiter. This parameter is returned for transactions with the **Payment way** parameter (payment method) set to P2P.		
additionalResponseCodes	ANS...1024	Conditional
Result of the cardholder billing address check. This parameter is returned for transactions of a merchant that has the **AVS enabled** option selected.		
email	ANS...225	No
Customer email address to send notifications. If it was specified.		

Request example

```
https://<EPG_URL>:<EPG_SSL_Port>/epg/rest/p2p/info?
 &orderId=d38cc71a-9522-43f7-b9d5-6bd4e01801&language=en
```

Response example

```
{
 "date": "12/4/20 1:28 PM",
```

"merchantShortName": "testapi",
"debitPanMasked": "4400 **** **** 0600 ",
"orderNumber": "277166461053221",
"backUrl": "https://epg.test.in:8443/p2p_register_verify.html"
"failUrl": "https://epg.test.in:8443/error_page.html"
"terminalId": "00999208",
"orderDescription": "Order description",
"amountFormatted": "30,00",
"merchantFullName": "testapi",
"creditPanMasked": "4400 **** **** 0700 ",
"currencyName": "USD",
"currency": "840",
"expiry": "07/2021",
"actionCodeDescription": "No error",
"formattedFeeAmount": "0,00",
"p2p.creditPan": "440000****0700",
"amount": "3000",
"emailEnabled": "true",
"panMasked": "4400 **** **** 0600 ",
"successUrl": "https://epg.test.in:8443/p2p_register_verify.html",
"authorizationResponseId": "131996",
"p2p.formattedFeeAmount": "0,00",
"feeAmount": "0",
"orderParams":
{ "sdkMaxTimeout": "10", "deviceChannel": "01" }

,
"emailRegExp": "^([a-z0-9_-]\\.)[a-z0-9_-]@[a-z0-9_-](
.[a-z0-9_-]).[a-z]
{2,6}
$",
"refNum": "000004131994",
"cardholderName": "TEST",
"paymentDate": "2020-12-04 13:28:27.661",
"merchantUrl": "https:/my-shop-test.ru",
"status": "DEPOSITED"
}

4.17.7 Request for money transfer status

The **p2p/status** request is used to obtain the status of a registered order.

The status of the order is to be defined by the value of the **orderStatus** parameter in the response.

Request parameters

Name	Type	Mandatory
orderId	ANS36	Yes
Unique order number generated by EPG after the registration of the order. Either **orderId** or **OrderNumber** (or both) must be specified in the request.		
orderNumber	AN 1..32	Yes
Number (identifier) of the order in the merchant's online store system. It is unique for every store in the system and is generated on the order registration. If the **Require system to generate order numbers** permission is enabled for the merchant, the order number is generated automatically. Otherwise, the order number data is sent in the API. Either **orderId** or **OrderNumber** (or both) must be specified in the request.		

Response parameters v.1

Name	Type	Mandatory

errorCode	N 1..3	Yes
Error code.		
errorMessage	AN 1..512	Yes
Description of the error in the language that was sent in the **language** parameter of the request.		
orderNumber	AN 1..32	No
Number (identifier) of the order in the merchant's online store system. It is unique for every store in the system and is generated on the order registration. If the **Require system to generate order numbers** permission is enabled for the merchant, the order number is generated automatically. Otherwise, the order number data is sent in the API.		
orderStatus	N1	No
Order status is determined in the payments system using the value of this parameter. The possible values of the parameter are as follows: · 0 — order is registered, but not paid. · 2 — full authorization of the order amount is performed. · 5 — authorization has been initiated via the issuer's ACS. · 6 — authorization is declined.		

panMaskedFrom	N 13...19	No
Masked number of the card to be debited.		
panMaskedTo	N 13...19	No
Masked number of the card to be credited.		
amount	N 1..19	No
Amount of the money transfer specified in the minor denomination of the currency.		
currency	N3	No
Payment currency in the ISO 4217 format.		
creationDate	UTC	No
Date of the order registration.		
orderDescription	AN..600	No
Description of the order.		

ip	AN..30	No
IP address of the customer.		
resultCode	N..2	Yes
Request execution error code: · 0 — success · 1 — error		
orderParams[]	See description	No
Tag containing the merchant attributes. More than one **orderParams** tag can be presented in the response. The name attribute contains the **name** of a parameter and the **value** attribute contains its value. See the **orderParams** tag description below.		
operationList[]	See description.	No
Tag containing information about the transactions completed in the order. More than one **operationList** tag can be present in the response. The tags that can be used are listed in the **operationList** tag table below.		

Response parameters v.2

Name	Type	Mandatory
errorCode	N 1..3	Yes
Error code.		
errorMessage	AN 1..512	Yes
Description of the error in the language that was sent in the **language** parameter of the request.		
orderNumber	AN 1..32	No
Number (identifier) of the order in the merchant's online store system. It is unique for every store in the system and is generated on the order registration. If the **Require system to generate order numbers** permission is enabled for the merchant, the order number is generated automatically. Otherwise, the order number data is sent in the API.		
orderStatus	N1	No
Order status is determined in the payments system using the value of this parameter. The possible values of the parameter are as follows: · 0 — order is registered, but not paid. · 2 — full authorization of the order amount is performed. · 5 — authorization has been initiated via the issuer's ACS.		

· 6 — authorization is declined.		
panMaskedFrom	N 13...19	No
Masked number of the card to be debited.		
panMaskedTo	N 13...19	No
Masked number of the card to be credited.		
amount	N 1..19	No
Amount of the money transfer specified in the minor denomination of the currency.		
fee	N...9	Yes
Fee amount in the minor currency units.		
currency	N3	No
Payment currency in the ISO 4217 format.		
creationDate	UTC	No
Date of the order registration.		

orderParams[]	See description	No
Tag containing the merchant attributes. More than one **orderParams** tag can be presented in the response. The name attribute contains the **name** of a parameter and the **value** attribute contains its value. See the **orderParams** tag description below.		
operationList[]	See description.	No
Tag containing information about the transactions completed in the order. More than one **operationList** tag can be present in the response. The tags that can be used are listed in the **operationList** tag table below.		
orderDescription	AN..600	No
Description of the order.		
ip	AN..30	No
IP address of the customer.		
resultCode	N..2	Yes
Request execution error code: · 0 — success		

- 1 — error

Format of the orderParams tag

Name	Type	Mandatory
name	ANS..255	Yes
Name of an additional parameter.		
value	ANS..1024	Yes
Value of the additional parameter.		

Format of the operationList tag

Name	Type	Mandatory
operationType	ANS..20	No

Transaction type: · P2P_VERIFY — requests the fee amount · P2P_DEBIT — debits the funds · P2P_CREDIT— credits the funds · P2P_DEBIT_REVERSAL — a debit cancellation · P2P_CREDIT_REVERSAL — a credit cancellation · P2P_TRANSFER — the card-to-card funds transfer · P2P_TRANSFER_REVERSAL — automatic return of the funds in case of an error during the transfer		
amount	N 1..19	No
Transaction amount in minor denomination of the currency.		
currency	N3	No
Payment currency code in the ISO 4217 format.		
datetime	UTC	No
Date and time of the transaction.		
resultCode	N..2	No
Transaction error code.		

resultCodeDescription	ANS..512	No
Description of the error code.		
maskedPan	N 13...19	No
Masked card number.		
cardholderName	ANS 2..26	No
Name of the cardholder. This parameter is verified according to the following criteria: · Acceptable characters are: Latin letters, 0-9, $,), (, -, . , a space · Cardholder name must start with a letter · Minimum length: 2 characters · Maximum length: 26 characters · Null is valid · Uppercase and lowercase are acceptable		
refNum	AN12	No
Unique remote reference (or ID) number that is assigned to the transaction on its completion.		

Error codes (errorCode) and messages (errorMessage)

Value	Description	Description
0	Success.	The request has been processed successfully.
5	Access denied.	The user does not have the necessary permissions to access this resource.
7	System error.	Software or hardware issue or malfunction.
14	Transaction failed.	Transaction processing was not successful. See P2P response codes for the list of failure reasons.

Request example

```
https://wls01-ecom.bpc.in/epg/rest/p2p/status?userName=test_api&password=1

{
   "orderNumber": "111hdvg38fdfj5g7uu1",
   "orderId": "c5ced013-d353-4794-8c08-b04006f56972"
}
```

Response example v.1

```
{"errorCode":0,"errorMessage":"Successful","orderNumber":"111hdvg38fdfj5g7uu","orderStatus":2,
 "panMaskedFrom":"220000****0500","amount":100,"fee":30,"currency":"840",
```

"creationDate":1602866480959,"orderDescription":"test data","ip":"10.0.2.233",
"orderParams":[],"operationList":[{"operationType":"P2P_VERIFY","amount":100,
"currency":840,"datetime":1602866500877,"resultCode":0,"resultCodeDescription":
"Request processed successfully","maskedPan":"000000****0000","authorizationResponseId"
:"514187","refNum":"000003514187"},{"operationType":"P2P_DEBIT","amount":100,"currency":840,
"datetime":1602866552718,"resultCode":0,"resultCodeDescription":"Request processed successfully",
"maskedPan":"220000****0500","cardholderName":"VERA TEST","authorizationResponseId":"514195",
"refNum":"000003514195"}],"resultCode":0}

Response example v.2

{
"errorCode": 0,
"errorMessage": "Successful",
"orderNumber": "001",
"orderStatus": 0,
"panMaskedFrom": "",
"amount": 130,
"fee": 0,
"currency": "840",
"creationDate": 1653394827125,
"orderParams": [],
"operationList": [],
"resultCode": 0
}

5. Reference information

This section provides reference information that includes:

5.1 JSON parameter list

The following parameters may be passed within **jsonParams[]** (or **params[]** for P2P payments) for a specific merchant:

Name	Method	Type
walletNumber	· **register.do** (as **jsonParams[]**) · **registerPreAuth.do** (as **jsonParams[]**)	N...23
Number of a mobile wallet used for card-to-wallet and wallet-to-card money transfers, a numeric value. The maximum number length is 23 digits.		

This parameter is optional.		
dc	· **register.do** (as **jsonParams[]**) · **registerPreAuth.do** (as **jsonParams[]**)	String
Specifies the type of money transfer to perform for the specified wallet: · DEBIT · CREDIT This parameter is mandatory if **walletNumber** is specified.		
suppressShippingAddress	· **register.do** (as **jsonParams[]**)	Boolean
Ignores the information about the shipping address and must be set to true when an order contains only digital goods. The default value is false. This parameter is optional and is used for payments with Masterpass.		
cartId	· **register.do** (as **jsonParams[]**)	AN..32
Identifier of an order in the Masterpass system. This parameter coincides with **orderNumber**, the identifier of the order in the merchant system. This parameter is mandatory and is used for payments with Masterpass.		
destination_external_card_id	· **p2p/register.do** (as **params[]**)	N...12

Options for linking an order with the card data: · card_id — use the identifier of the customer card stored in SVFE (or another OLTP system). · binding — use the identifier of the binding created for the customer's card in the merchant system. This parameter is optional. If this parameter is not specified, the card data must be provided in the **toCard** field in the further request (a transfer verification or payment request).		
destination_type	· **p2p/register.do** (as **params[]**)	String
Type of card identifier for external systems: · card_id — a value of **destination_external_card_**id must be a card ID in SVFE. · binding — a value of **destination_external_card_id** must be a binding ID. This parameter is optional.		
postalCode	For Web requests: · **paymentorder.do** (as **jsonParams[]**) · **applepay/payment.do** (as **jsonParams[]**) · **finish3dsPayment.do** (as **jsonParams[]**). The parameter is received from ACS · **p2p/perform.do** (as **params[]**)	AN..9

Cardholder's postal code for the AVS checks. This parameter is optional. Note: This parameter must be transferred in a request if the **AVS enabled** option is selected for the merchant.		
streetAddress	For Web requests: · **paymentorder.do** (as **jsonParams[]**) · **applepay/payment.do** (as **jsonParams[]**) · **finish3dsPayment.do** (as **jsonParams[]**). The parameter is received from ACS · **p2p/perform.do** (as **params[]**)	AN..40
Cardholder's street address for the AVS checks. This parameter is optional. Note: This parameter must be transferred in a request if the **AVS enabled** option is selected for the merchant.		
paymentWay	For Web requests: · **register.do** (as **jsonParams[]**) · **registerPreAuth.do** (as **jsonParams[]**) For SOAP requests: · **registerOrder** (as **jsonParams[]**)	AN..32

	· **registerOrderPreAuth** (as **jsonParams[]**)	
Payment method used for the payment. This parameter is optional. Use this parameter in **jsonParams[]** if you want to redefine a predefined payment method and to use custom payment methods that differ from the standard methods. For more information, see Customizing payment methods.		
challengeIndicator	· **register.do** (as **jsonParams[]**) · **registerPreAuth.do** (as **jsonParams[]**) · **addParams.do** (as **params[]**)	N2
Value that the merchant can submit to request an acquirer exemption: · 05 - No challenge requested — transactional risk analysis has already been performed · 06 - No challenge requested — data share only · 07 - No challenge requested — strong consumer authentication has already been performed This parameter is optional.		
merchantFraudRate	· **register.do** (as **jsonParams[]**) · **registerPreAuth.do** (as **jsonParams[]**) · **addParams.do** (as **params[]**)	N1..2

Merchant fraud rate expressed in basis points, where 1 basis point is 0.01%: · 1 — represents a fraud rate <=1 bp · 2 — represents a fraud rate 1+..6 bp · 3 — represents a fraud rate 6+..13 bp · 4 — represents a fraud rate 13+..25 bp · 5 — represents a fraud rate >25 bp This parameter is optional.		
secureCorporatePayment	· **register.do** (as **jsonParams[]**) · **registerPreAuth.do** (as **jsonParams[]**) · **addParams.do** (as **params[]**)	A1
Specifies whether a payment is a subject of a secure corporate payment exemption: · Y — yes · N — no This parameter is optional.		
senderAccountType	**p2p/register.do** (as **params[]**)	String

Type of the account from which funds is transferred: · RTN_BANK · IBAN · CARD_ACCOUNT · EMAIL · PHONE_NUMBER · BIC_BAN · WALLET_ID · SOCIAL_NETWORK_ID · OTHER This parameter is optional.		
receiverAccountType	**p2p/register.do** (as **params[]**)	String
Type of the account to which funds is transferred: · RTN_BANK · IBAN · CARD_ACCOUNT · EMAIL · PHONE_NUMBER · BIC_BAN · WALLET_ID · SOCIAL_NETWORK_ID · OTHER This parameter is optional.		
destWalletNum	**register.do** (as **jsonParams[]**)	AN..30

Recipient‘s wallet number used for replenishment. This parameter is optional. This parameter is transmitted in **messageextension** only if this information is received in the **jsonParams** block and works only for MCC (6050 and 6051). Note: The **messageextension** parameters are only populated for NSPK MIR.		
destPhoneNum	**register.do** (as **jsonParams[]**)	AN..30
Recipient‘s mobile phone number used for replenishment. This parameter is passed to 3DS in **messageExtension** and only works for MCC 4814. This parameter is optional Note: The **messageextension** parameters are filled only for NSPK MIR.		
recipientAccountNumber	**P2P/register.do** (as **params[]**)	N..20
Number of the bank account of the payee in the recipient’s bank. It is used for card-to-account money transfers. If **MCC** (**6538**) is used and this parameter is specified, **destAcctNum** is filled in **messageextension** to be transmitted in the AReq message for the NSPK MIR payments network in the following format: ‘recipientAccountBankCode + recipientAccountNumber’ This parameter is optional. Notes: The **messageextension** parameters are filled only for NSPK MIR.		
recipientAccountBankCode	**P2P/register.do** (as **params[]**)	N..9

<table>
<tr><td colspan="3">Business Identifier Code (BIC) of the recipient’s bank. It is used for card-to-account money transfers. If MCC (6538) is used and this parameter is specified, destAcctNum is filled in messageextension to be transmitted in the AReq message for the NSPK MIR payments network in the following format: ‘recipientAccountBankCode + recipientAccountNumber’

This parameter is optional.

Note: The messageextension parameters are filled only for NSPK MIR.

IMPORTANT: If either recipientAccountNumber or recipientAccountBankCode is specified, both these parameters must be present.</td></tr>
<tr><td>failURL</td><td>· register.do (as jsonParams[])
· registerPreAuth.do (as jsonParams[])
· P2P/register.do params[]
· register.do (as jsonParams[])
· create.do (as jsonParams[])</td><td>AN..255</td></tr>
<tr><td colspan="3">URL to which the user is redirected in the case of an unsuccessful payment.

This parameter is optional.</td></tr>
<tr><td>merchantTaxIdType</td><td>· register.do (as jsonParams[])
· registerPreAuth.do (as jsonParams[])
· ApplePay/GooglePay/SamsungPay Payment APIs additionalParameters[]
· mastercard/scof/payment/decrypted (as jsonParams[])</td><td>N..1</td></tr>
</table>

<table>
<tr><td colspan="3">Tax ID type:

· 1 – Corporate
· 2 – Small business, includes individual

This parameter is optional. If the merchantTaxIdType value is set, the merchantTaxId parameter must be configured.</td></tr>
<tr><td>merchantTaxId</td><td>· register.do (as jsonParams[])
· registerPreAuth.do (as jsonParams[])
· ApplePay/GooglePay/SamsungPay Payment APIs additionalParameters[]
· mastercard/scof/payment/decrypted jsonParams[]</td><td>ANS..20</td></tr>
<tr><td colspan="3">Merchant tax ID. If the api.validation.merchantTaxId.regex parameter in the system configuration is set, the merchantTaxId value must comply with this pattern.

This parameter is optional</td></tr>
<tr><td>descriptorName</td><td>· register.do (as jsonParams[])
· registerPreAuth.do (as jsonParams[])
· refund.do (as jsonParams[])</td><td>ANS1..22</td></tr>
<tr><td colspan="3">Merchant name, Service Organization name, Point-of-Sale name, Bank Branch name, where operation is taking place, or its location name. This information is included in the Dynamic Descriptor.</td></tr>
</table>

descriptorLocation	· **register.do** (as **jsonParams[]**) · **registerPreAuth.do** (as **jsonParams[]**) · **refund.do** (as **jsonParams[]**)	ANS1..13
The city where the terminal is located. This information is included in the Dynamic Descriptor.		
descriptorCountryCode	· **register.do** (as **jsonParams[]**) · **registerPreAuth.do** (as **jsonParams[]**) · **refund.do** (as **jsonParams[]**)	ANS..2
Code of the country where the terminal is located according to ISO 3166-1. This information is included in the Dynamic Descriptor.		
p2p_type	**p2p/register.do** (as **params[]**)	String
P2P transaction type: · Standard — standard P2P transfer. This is the default value. · Debit — funds transfer from the sender's card · Credit — funds transfer to the sender's card		
senderAccountNumber	**p2p/register.do** (as **params[]**)	N..20
Account number of the P2P payment sender for the Credit transactions. This parameter is optional.		
eCertTransactionCartId	· **register.do** (as **jsonParams[]**) · **registerPreAuth.do** (as **jsonParams[]**)	String (24)

	· **refund.do** (as **jsonParams[]**)	
Identifier of the cart received as **purchaseBasketId** from the NSPK Electronic Certificate Front Office system.		
eCertRefundCartId	**refund.do** (as **jsonParams[]**)	String (24)
Identifier of the cart with returned items received as **returnBasketId** from the NSPK Electronic Certificate Front Office system.		
clientConfirmationReceived	· **processform.do** (as **jsonParams[]**) · **processBindingForm.do** (as **jsonParams[]**) · **paymentorder.do** (as **jsonParams[]**) · **paymentOrderBinding.do** (as **jsonParams[]**)	Boolean
Specifies that the customer has accepted the agreement terms for the recurring payment. This parameter is only checked if **MIT.CoF.clientConfirmation.enabled=**true in the system settings. Possible values: · true — the agreement was accepted and the initial recurring payment can be performed. · false — the agreement was not confirmed and the initial recurring payment cannot be performed.		
paymentWithDelayedClearing	**registerPreAuth.do** (as **jsonParams[]**)	Boolean

<table>
<tr><td colspan="3">Specifies whether a merchant can place a customer’s payment on hold temporarily:
· true — the payment with delayed clearing has been initiated and message 200 has been transmitted to SVFE. In EPG, Field 47 tag 239 is set to 1.
· false — a normal preauthorization. The payment is also processed as a normal preauthorization if the paymentWithDelayedClearing parameter is not set. This is the default value.

The difference between the preauthorized transaction is that this transaction is recognized as an ordinary purchase by SVFE. However, the payment with the delayed clearing is processed after some time (typically seven days) when the completion request is executed. To be able to use this parameter, the merchant must have the Merchant is allowed to process payment with delayed clearing permission enabled.</td></tr>
<tr><td>force_terminal_id</td><td>· registerPreAuth.do (as jsonParams[])
· register.do (as jsonParams[])
· paymentorder.do (as jsonParams[])
· processform.do (as jsonParams[])
· p2p/register.do (p2p/perform.do) (as params[])</td><td>ANS..8</td></tr>
<tr><td colspan="3">Identifier of the terminal that must be used in the transaction. A specific terminal ID can be specified if the merchant.acquirer.config.unique system setting is disabled.

If the force_terminal_id is not set, the default acquirer terminal is used.</td></tr>
<tr><td>fundingTypeIndicator</td><td>p2p/register.do (as params[])</td><td>String</td></tr>
</table>

Funding/Payment Transaction Type Indicator:

- AA – Account to account
- AC – Card activation
- BB – Business to business
- BD – Business disbursement
- BI – Money transfer—bank-initiated
- BP – Non-card bill payment
- CA – Cash2ATM
- CC – Cash claim
- CI – Cash in
- CO – Cash out
- CP – Card bill payment
- FD – Funds disbursement (general)
- FT – Fund transfer
- GD – Government disbursement
- GP – Gambling payout (other than online gambling)
- LO – Loyalty and offers
- MA – Mobile air time payment
- MD – Merchant disbursement
- MI – Money transfer—merchant-initiated
- MP – Face-to-face merchant payment
- MS – Merchant settlement
- OG – Online gambling payout
- P1 – Person to person
- PD – Payroll/pension disbursement
- PG – Payment to government
- PP – Person to person
- PS – Payment for goods and services (general)
- TU – Top-Up for enhanced prepaid loads
- WT – Wallet transfer
- B1 – Utility Payment, Brazil domestic transactions
- B2 – Government Services, Brazil domestic transactions
- B3 – Mobile phone top-ups, Brazil domestic transactions

· B4 – Coupon		
sourceOfFunds	**p2p/register.do** (as **params[]**)	N...2
Source of the funds used by the sender and applicable for all remittance transactions: · 1 – Credit card · 2 – Debt card · 3 – Prepaid card · 4 – Deposit/Debit account · 5 – Credit account · 6 – Mobile money account · 7 – Cash · 8 – Check · 9 – Other		
digitalCommerceDomainIdentifier	· **register.do** (as **jsonParams[]**) · **registerPreAuth.do** (as **jsonParams[]**)	n ..1
Entity responsible for populating the digital commerce solutions indicators defined by the **digitalCommerceProgramType** parameter according to Mastercard requirements: · 2 – Acquirer (applicable for EPG payments) · 3 – Issuer		
digitalCommerceProgramType	· **register.do** (as **jsonParams[]**) · **registerPreAuth.do** (as **jsonParams[]**)	n ..1

Digital commerce program type specified when processing in-store biometric initiated transactions:

- 1 – SQR — refers to Square, a platform that enables individuals to create a free, reusable digital identity for secure and efficient online transactions
- 2 – Click to Pay — a type of digital commerce program focused on simplifying the online checkout process and enhancing security
- 3 – Biometrics Checkout Program — a type of E-Commerce program that utilizes unique physical characteristics, such as fingerprints or facial recognition, to authenticate payments at checkout
- 4 – Gen POI — refers to a generic or broad category of Point of Interest that encompasses various types of businesses and facilities

Mastercard updates in-store biometric-initiated transactions certified against the Mastercard Biometric Checkout requirements to reflect the combination of the **digitalCommerceDomainIdentifier** and the **digitalCommerceProgramType** parameter values.

Note: If values of the **digitalCommerceDomainIdentifier** and **digitalCommerceProgramType** parameters are submitted in the **epg/api/v1/mastercard/scof/payment/decrypted** request, the values submitted during the registration of the order (using the **register.do** or **registerPreAuth.do method**) are overwritten.

The **jsonParams[]** group also passes the 3DS authentication result (see Passing 3DS authentication result from external MPI or 3DSS).

The **jsonParams[]** group can also be used to submit customer-specific parameters. These parameters are sent in the authorization request to the online transaction processing system (OLTP) if configured as SV ISO fields in the EPG web interface.

The information from the descriptor fields (**descriptorName**, **descriptorLocation**, and **descriptorCountryCode**) is used by EPG to generate field **DE43** (**Card acceptor name/Location**) which is sent to SVFE. EPG generates **DE43** only if all three fields are populated.

A merchant sends the descriptor information for every order if a dynamic descriptor is necessary. Otherwise, the default data from SVFE is sent. A Dynamic Descriptor includes information about the merchant to be printed in the cardholder's bank statement.

Note: Parameters starting with _internal_ are prohibited from use in the **params** block in **/p2p/register.do**.

5.2 Additional parameter list

The following parameters may be passed within **params[]** (or **additionalParameters**) for a specific merchant:

Name	Method	Type
walletNumber	· **registerOrder** (as **params[]**) · **registerOrderPreAuth** (as **params[]**)	N...23
Number of a mobile wallet used for card-to-wallet and wallet-to-card money transfers, a numeric value. The maximum number length is 23 digits. This parameter is optional.		
dc	· **registerOrder** (as **params[]**) · **registerOrderPreAuth** (as **params[]**)	String
Specifies the type of money transfer to perform for the specified wallet: · DEBIT · CREDIT This parameter is mandatory if **walletNumber** is specified.		

suppressShippingAddress	· **registerOrder** (as **params[]**)	Boolean
Ignores the information about the shipping address and must be set to true when an order contains only digital goods. The default value is false. This parameter is optional and is used for payments with Masterpass.		
cartId	· **registerOrder** (as **params[]**)	AN..32
Identifier of an order in the Masterpass system. This parameter coincides with **orderNumber**, the identifier of the order in the merchant system. This parameter is mandatory and is used for payments with Masterpass.		
destination_external_card_id	· **registerP2P** (as **params[]**)	N...12
Options for linking an order with the card data: · card_id — use the identifier of the customer card stored in SVFE (or another OLTP system). · binding — use the identifier of the binding created for the customer's card in the merchant system. This parameter is optional. If this parameter is not specified, the card data must be provided in the **toCard** field in the further request (a transfer verification or payment request).		
destination_type	· **registerP2P** (as **params[]**)	String

Type of card identifier for external systems: · card_id — a value of **destination_external_card_**id must be a card ID in SVFE. · binding — a value of **destination_external_card_id** must be a binding ID. This parameter is optional.		
postalCode	· **paymentOrder** (as **params[]**) · **applePay** (as **params[]**) · **performP2P** (as **params[]**)	AN..9
Cardholder's postal code for the AVS checks. This parameter is optional. Note: This parameter must be transferred in a request if the **AVS enabled** option is selected for the merchant.		
streetAddress	· **paymentOrder** (as **params[]**) · **applePay** (as **params[]**) · **performP2P** (as **params[]**)	AN..40
Cardholder's street address for the AVS checks. This parameter is optional. Note: This parameter must be transferred in a request if the **AVS enabled** option is selected for the merchant.		
paymentWay	· **registerOrder** (as **params[]**) · **registerOrderPreAuth** (as **params[]**)	AN..32
Payment method used for the payment. This parameter is optional. Use this parameter in **params[]** if you want to redefine a predefined payment method and to use custom payment methods that differ from the standard methods.		

For more information, see Customizing payment methods.		
challengeIndicator	· **registerOrder** (as **params[]**) · **registerPreAuth.do** (as **params[]**)	N2
Value that the merchant can submit to request an acquirer exemption: · 05 - No challenge requested — transactional risk analysis has already been performed · 06 - No challenge requested — data share only · 07 - No challenge requested — strong consumer authentication has already been performed This parameter is optional.		
merchantFraudRate	· **registerOrder** (as **params[]**) · **registerOrderPreAuth** (as **params[]**) · **addParams.do** (as **params[]**)	N1..2
Merchant fraud rate expressed in basis points, where 1 basis point is 0.01%: · 1 — represents a fraud rate <=1 bp · 2 — represents a fraud rate 1+..6 bp · 3 — represents a fraud rate 6+..13 bp · 4 — represents a fraud rate 13+..25 bp · 5 — represents a fraud rate >25 bp This parameter is optional.		
secureCorporatePayment	· **registerOrder** (as **params[]**) · **registerOrderPreAuth** (as **params[]**) · **addParams** (as **params[]**)	A1

Specifies whether a payment is a subject of a secure corporate payment exemption: · Y — yes · N — no This parameter is optional.		
senderAccountType	**registerP2P** (as **params[]**)	String
Type of the account from which funds is transferred: · RTN_BANK · IBAN · CARD_ACCOUNT · EMAIL · PHONE_NUMBER · BIC_BAN · WALLET_ID · SOCIAL_NETWORK_ID · OTHER This parameter is optional.		
receiverAccountType	**registerP2P** (as **params[]**)	String

Type of the account to which funds is transferred: · RTN_BANK · IBAN · CARD_ACCOUNT · EMAIL · PHONE_NUMBER · BIC_BAN · WALLET_ID · SOCIAL_NETWORK_ID · OTHER This parameter is optional.		
destWalletNum	**registerOrder** (as **params[]**)	AN..30
Recipient‘s wallet number used for replenishment. This parameter is optional. This parameter is transmitted in **messageextension** only if this information is received in the **params** block and works only for MCC (6050 and 6051). Note: The **messageextension** parameters are only populated for NSPK MIR.		
destPhoneNum	**registerOrder** (as **params[]**)	AN..30
Recipient‘s mobile phone number used for replenishment. This parameter is passed to 3DS in **messageExtension** and only works for MCC 4814. This parameter is optional Note: The **messageextension** parameters are filled only for NSPK MIR.		
recipientAccountNumber	**registerP2P** (as **params[]**)	N..20

<table>
<tr><td colspan="3">Number of the bank account of the payee in the recipient’s bank. This parameter is optional. It is used for card-to-account money transfers. If the MCC (6538) is used and this parameter is specified, destAcctNum is filled in messageextension to be transmitted within the AReq message for the NSPK MIR payments network in the following format: ‘recipientAccountBankCode + recipientAccountNumber’

Notes: The messageextension parameters are filled only for NSPK MIR.</td></tr>
<tr><td>recipientAccountBankCode</td><td>registerP2P (as params[])</td><td>N..9</td></tr>
<tr><td colspan="3">Business Identifier Code (BIC) of the recipient’s bank. This parameter is optional. It is used for card-to-account money transfers. If the MCC (6538) is used and this parameter is specified, destAcctNum is filled in messageextension to be transmitted within the AReq message for the NSPK MIR payments network in the following format: ‘recipientAccountBankCode + recipientAccountNumber’.

Note: The messageextension parameters are filled only for NSPK MIR.

IMPORTANT: If either recipientAccountNumber or recipientAccountBankCode is specified, both these parameters must be present.</td></tr>
<tr><td>failURL</td><td>· registerOrder (as params[])
· registerOrderPreAuth (as params[])
· registerP2P
params[]
· registerOrder (as params[])
· create (as params[])</td><td>AN..255</td></tr>
<tr><td colspan="3">URL to which the user is redirected in the case of an unsuccessful payment.

This parameter is optional.</td></tr>
</table>

merchantTaxIdType	· **registerOrder** (as **params[][]**) · **registerOrderPreAuth** (as **Params[]**) · **ApplePay/GooglePay/SamsungPay Payment APIs additionalParameters[]** · **mastercard/scof/payment/decrypted** (as **params[]**)	N..1
Tax ID type: · 1 – Corporate · 2 – Small business, includes individual This parameter is optional. If the **merchantTaxIdType value is set,** the **merchantTaxId** parameter must be configured.		
merchantTaxId	· **registerOrder** (as **params[]**) · **registerOrderPreAuth** (as **params[]**) · **ApplePay/GooglePay/SamsungPay Payment APIs additionalParameters[]** · **mastercard/scof/payment/decrypted params[]**	AN..20
Merchant tax ID. This parameter is optional. If the **api.validation.merchantTaxId.regex** parameter in the system configuration is set, the **merchantTaxId** value must comply with this pattern.		
descriptorName	· **registerOrder** (as **params[]**) · **registerOrderPreAuth** (as **params[]**) · **refundOrder** (as **params[]**)	ANS1..22
Merchant name, Service Organization name, Point-of-Sale name, Bank Branch name, where operation is taking place or its location name. This information is included in the Dynamic Descriptor.		

descriptorLocation	· **registerOrder** (as **params[]**) · **registerOrderPreAuth** (as **params[]**) · **refundOrder** (as **params[]**)	ANS1..13
Terminal location. The city where the terminal is located. This information is included in the Dynamic Descriptor.		
descriptorCountryCode	· **registerOrder** (as **params[]**) · **registerOrderPreAuth** (as **params[]**) · **refundOrder** (as **params[]**)	ANS..2
The code of the country where the terminal is located according to ISO 3166-1. This information is included in the Dynamic Descriptor.		
p2p_type	**registerP2P** (as **params[]**)	String
P2P transaction type: · Standard — standard P2P transfer. This is the default value. · Debit — funds transfer from the sender's card · Credit — funds transfer to the sender's card		
senderAccountNumber	**registerP2P** (as **params[]**)	N..20
Account number of the P2P payment sender for the Credit transactions. This parameter is optional.		
eCertTransactionCartId	· **registerOrder** (as **params[]**) · **registerOrderPreAuth** (as **params[]**) · **refundOrder** (as **params[]**)	String (24)

The identifier of the cart received as **purchaseBasketId** from the NSPK Electronic Certificate Front Office system.		
eCertRefundCartId	**refundOrder** (as **params[]**)	
The identifier of the cart with returned items received as **returnBasketId** from the NSPK Electronic Certificate Front Office system.		
clientConfirmationReceived	· **processform** (as **params[]**) · **processBindingForm** (as **params[]**) · **paymentOrder** (as **params[]**) · **paymentOrderBinding**(as **params[]**)	Boolean
Specifies that the customer has accepted the agreement terms for the recurring payment. This parameter is only checked if **MIT.CoF.clientConfirmation.enabled=**true in the system settings. Possible values: · true — the agreement was accepted, the initial recurring payment can be performed. · false — the agreement was not confirmed, the initial recurring payment cannot be performed.		
paymentWithDelayedClearing	**registerOrderPreAuth** (as **params[]**)	Boolean

Specifies whether a merchant can place a customer's payment on hold temporarily:

- true — the payment with delayed clearing has been initiated and message 200 has been transmitted to SVFE. In EPG, Field **47** tag **239** is set to 1.
- false — a normal preauthorization. The payment is also processed as a normal preauthorization if the **paymentWithDelayedClearing** parameter is not set. This is the default value.

The difference between the preauthorized transaction is that this transaction is recognized as an ordinary purchase by SVFE. However, the payment with the delayed clearing is processed after some time (typically seven days) when the completion request is executed. To be able to use this parameter, the merchant must have the **Merchant is allowed to process payment with delayed clearing** permission enabled.

force_terminal_id	· **registerOrderPreAuth** (as **params[]**) · **registerOrder** (as **params[]**) · **paymentOrder** (as **params[]**) · **processform** (as **params[]**) · **registerP2P (performP2P)** (as **params[][]**)	ANS..8

The identifier of the terminal that must be used in the transaction. A specific terminal ID can be specified if the **merchant.acquirer.config.unique** system setting is disabled.

If the **force_terminal_id** is not set, the default acquirer terminal is used.

fundingTypeIndicator	**registerP2P** (as **params[]**)	String

Funding/Payment Transaction Type Indicator:

- AA – Account to account
- AC – Card activation
- BB – Business to business
- BD – Business disbursement
- BI – Money transfer—bank-initiated
- BP – Non-card bill payment
- CA – Cash2ATM
- CC – Cash claim
- CI – Cash in
- CO – Cash out
- CP – Card bill payment
- FD – Funds disbursement (general)
- FT – Fund transfer
- GD – Government disbursement
- GP – Gambling payout (other than online gambling)
- LO – Loyalty and offers
- MA – Mobile air time payment
- MD – Merchant disbursement
- MI – Money transfer—merchant-initiated
- MP – Face-to-face merchant payment
- MS – Merchant settlement
- OG – Online gambling payout
- P1 – Person to person
- PD – Payroll/pension disbursement
- PG – Payment to government
- PP – Person to person
- PS – Payment for goods and services (general)
- TU – Top-Up for enhanced prepaid loads
- WT – Wallet transfer
- B1 – Utility Payment, Brazil domestic transactions
- B2 – Government Services, Brazil domestic transactions
- B3 – Mobile phone top-ups, Brazil domestic transactions

· B4 – Coupon		
sourceOfFunds	**registerP2P** (as **params[]**)	N...2
The source of the funds used by the sender and applicable for all remittance transactions: · 1 — Credit card · 2 — Debt card · 3 — Prepaid card · 4 — Deposit/Debit account · 5 — Credit account · 6 — Mobile money account · 7 — Cash · 8 — Check · 9 — Other		
digitalCommerceDomainIdentifier	**epg/api/v1/mastercard/scof/payment/decrypted** (as **additionalParameters[]**or **params[]**)	n ..1
Entity responsible for populating the digital commerce solutions indicators defined by the **digitalCommerceProgramType** parameter according to Mastercard requirements: · 2 – Acquirer (applicable for EPG payments) · 3 – Issuer		
digitalCommerceProgramType	**epg/api/v1/mastercard/scof/payment/decrypted** (as **additionalParameters[]**or **params[]**)	n ..1

Digital commerce program type specified when processing in-store biometric initiated transactions:

- 1 – SQR — refers to Square, a platform that enables individuals to create a free, reusable digital identity for secure and efficient online transactions
- 2 – Click to Pay — a type of digital commerce program focused on simplifying the online checkout process and enhancing security
- 3 – Biometrics Checkout Program — a type of E-Commerce program that utilizes unique physical characteristics, such as fingerprints or facial recognition, to authenticate payments at checkout
- 4 – Gen POI — refers to a generic or broad category of Point of Interest that encompasses various types of businesses and facilities

Mastercard updates in-store biometric-initiated transactions certified against the Mastercard Biometric Checkout requirements to reflect the combination of the **digitalCommerceDomainIdentifier** and the **digitalCommerceProgramType** parameter values.

Note: If values of the **digitalCommerceDomainIdentifier** and **digitalCommerceProgramType** parameters are submitted in the **epg/api/v1/mastercard/scof/payment/decrypted** request, the values submitted during the registration of the order (using the **register.do** or **registerPreAuth.do method**) are overwritten.

The **params[]** group also passes the 3DS authentication result (see Passing 3DS authentication result from external MPI or 3DSS).

The **params[]** group can also be used to submit customer-specific parameters. These parameters are sent in the authorization request to the online transaction processing system (OLTP) if configured as SV ISO fields in the EPG web interface.

The information from the descriptor fields (**descriptorName**, **descriptorLocation** and **descriptorCountryCode**) is used by EPG to generate field **DE43** (**Card acceptor name/Location**) which is sent to SVFE. EPG generates **DE43** only if all three fields are filled.

A merchant sends the descriptor information for every order if a dynamic descriptor is necessary. Otherwise, the default data from SVFE is sent. A Dynamic Descriptor includes information about the merchant to be printed in the cardholder's bank statement.

Note: Parameters starting with _internal_ are prohibited from use in the **params** block in **/registerP2P**.

5.3 3-D Secure 2 parameter list

The following parameters may be transferred in the **threeDS2Params** block in a method used to send a payment request (**paymentorder.do** or **paymentotherway.do**).

Name	Type	Mandatory
browserAcceptHeader	ANS..2048	No
Exact content of the HTTP accept headers as it is sent to the 3DS Requestor from the cardholder's browser. If the total length of the accept header sent by the browser exceeds 2048 characters, the 3DS Server truncates the excess portion. Note: The following is mandatory content for the **browserAcceptHeader** property in all requests: application/json, text/javascript, */*;		
browserJavaEnabled	Boolean	No

Specifies whether the cardholder's browser can execute Java: · true · false This value is returned from the **navigator.javaEnabled** property.		
browserLanguage	ANS..8	No
Browser language as defined in IETF BCP47. This value is returned from the **navigator.language** property.		
browserColorDepth	N..2	No
Bit depth of the colour palette for displaying images, in bits per pixel. This value is obtained from the cardholder's browser using the **screen.colorDepth** property.		
browserScreenHeight	N..6	No
Total height of the cardholder's screen in pixels. This value is returned from the **screen.height** property.		
browserScreenWidth	N..6	No
Total width of the cardholder's screen in pixels.		

This value is returned from the **screen.width** property.		
browserTZ	N..5	No
Time difference between UTC and the cardholder's browser local time, in minutes.		
browserUserAgent	ANS..2048	No
Exact content of the HTTP **User-Agent** header.		
deviceChannel	N2	Yes
Type of channel interface that is used to initiate the transaction: · 01 — app-based (APP) · 02 — browser (BRW) · 03 — 3DS Requestor Initiated (3RI) · 04–79 — reserved for EMVCo future use (values invalid until defined by EMVCo) · 80–99 — reserved for DS use		
deviceRenderOptions[]		Yes

Information about the rendering types and interface that the device supports. This parameter defines the SDK UI types that the device supports for displaying specific challenge user interfaces within the SDK. Note: All device rendering options must be supported by the SDK and ACS components. Data will be formatted into a JSON object before being placed into the Device Rendering Options Supported field of the message. A detailed description of the elements of the deviceRenderOptions block is provided below.		
sdkAppID	ANS36	Yes
Universally unique identifier created after all installations and updates of the 3DS Requestor App on a Consumer Device are done. This identifier will be newly generated and stored by the 3DS SDK for each installation or update. The identifier format is standard as defined in IETF RFC 4122.		
sdkEphemPubKey	ANS..256	Yes
Public key component of the ephemeral key pair generated by the 3DS SDK and used to establish session keys between the 3DS SDK and ACS. In AReq, this data element is present as its own object. In ARes, this data element is contained within the ACS Signed Content JWS Object.		
sdkMaxTimeout	N2	Yes

Maximum time interval (in minutes) for all exchanges.		
sdkReferenceNumber	ANS32	Yes
Vendor and version for the 3DS SDK that is integrated in a 3DS Requestor App, assigned by EMVCo when the 3DS SDK is approved.		
sdkTransID	ANS36	Yes
Universally unique transaction identifier assigned by the 3DS SDK to identify a single transaction.		
cardholderAccount[]		No
Optional information about the cardholder account. A detailed description of the elements of the cardholderAccount block is provided below.		
merchantRiskIndicator[]		No
Optional information about the specific purchase made by the cardholder. A detailed description of the elements of the merchantRiskIndicator block is provided below.		
threeDSRequestorAuthentication[]		No

Optional information about how the cardholder was authenticated during the process of logging in to their 3DS Requestor account. A detaileddescription of the elements of the threeDSRequestorAuthentication block is provided below.		
threeDSRequestorPriorTransactionAuthentication[]		No
Optional information about the 3DS authentication of a cardholder that occurred before the current transaction. A detailed description of the elements of the threeDSRequestorPriorTransactionAuthentication block is provided below.		

The deviceRenderOptions block parameters

Name	Type	Mandatory
sdkInterface	N2	Yes
This parameter lists all of the SDK Interface types that the device supports for displaying specific challenge user interfaces within the SDK. The accepted values are as follows: · 01 — native · 02 — HTML · 03 — both		

sdkUiType	N2	Yes
This parameter lists all UI types that the device supports for displaying specific challenge user interfaces within the SDK. The accepted values are as follows: · 01 — text · 02 — single select · 03 — multi select · 04 — OOB · 05 — HTML other (valid only for HTML UI) The valid values for each interface are as follows: · Native UI — 01 to 04 · HTML UI — 01 to 05 Note: All SDKs must support all UI types.		

The cardholderAccount block parameters

Name	Type	Mandatory
chAccAgeInd	N2	No

Time passed since the cardholder account was created at the 3DS Requestor: · 01 — no account (guest check-out) · 02 — created during this transaction · 03 — less than 30 days · 04 — 30 to 60 days · 05 — more than 60 days		
chAccChange	N8	No
Date when the cardholder account at the 3DS Requestor was last changed, including changing the billing or shipping address, creating a new payment account, or new user (or users). The date format is *YYYYMMDD*.		
chAccChangeInd	N2	No
Time passed since the cardholder information has been changed last in the cardholder's 3DS Requestor account (including the change of billing or shipping address, creating a new payment account, or new user (or users)): · 01 — changed during this transaction · 02 — less than 30 days · 03 — 30 to 60 days · 04 — more than 60 days		
chAccDate	N8	No
Date when the cardholder opened the account at the 3DS Requestor.		

The date format is *YYYYMMDD*.		
chAccPwChange	N8	No
Date when the password for the cardholder account at the 3DS Requestor was changed or the account was reset. The date format is *YYYYMMDD*.		
chAccPwChangeInd	N2	No
Time passed since the password for the cardholder account at the 3DS Requestor has been changed or the account has been reset: · 01 — no change · 02 — changed during this transaction · 03 — less than 30 days · 04 — 30 to 60 days · 05 — more than 60 days		
nbPurchaseAccount	N..4	No
Number of purchases made through this cardholder account during the previous six months.		
provisionAttemptsDay	N..3	No
Number of Add Card attempts during the last 24 hours.		

txnActivityDay	N..3	No
Number of transactions (successful and abandoned) for this cardholder account at the 3DS Requestor across all payment accounts in the previous 24 hours.		
txnActivityYear	N..3	No
Number of transactions (successful and abandoned) for this cardholder account at the 3DS Requestor across all payment accounts in the previous year.		
paymentAccAge	N8	No
Date when the payment account was enrolled in the cardholder's account at the 3DS Requestor. The date format is *YYYYMMDD*.		
paymentAccInd	N2	No
Time passed since the payment account has been enrolled in the cardholder's account at the 3DS Requestor: · 01 — no account (guest check-out) · 02 — during this transaction · 03 — less than 30 days · 04 — 30 to 60 days · 05 — more than 60 days		

shipAddressUsage	N8	No
Date when the shipping address used for the current transaction was first used at the 3DS Requestor. The date format is *YYYYMMDD*.		
shipAddressUsageInd	N2	No
Time passed since the shipping address used for the current transaction has first been used at the 3DS Requestor: · 01 — this transaction · 02 — less than 30 days · 03 — 30 to 60 days · 04 — more than 60 days		
shipNameIndicator	N2	No
Specifies whether the **Cardholder Name** in the account is identical to the shipping **Name** used for the current transaction: · 01 — account Name is identical to shipping Name · 02 — account Name is different than shipping Name		
suspiciousAccActivity	N2	No

Specifies whether the 3DS Requestor has previously experienced any suspicious activity (including fraud) in the cardholder account: · 01 — no suspicious activity has been observed · 02 — suspicious activity has been observed

The merchantRiskIndicator block parameters

Name	Type	Mandatory
deliveryEmailAddress	N..254	No
For electronic delivery, the email address to which the purchase was delivered.		
deliveryTimeframe	N2	No
Purchase delivery time frame: · 01 — electronic delivery · 02 — same day shipping · 03 — overnight shipping · 04 — two-day or more shipping		
giftCardAmount	N..15	No
For a prepaid or gift card, the total amount of a purchase made by the card, in major currency units. The amount part in minor units is truncated: for example, for USD 123.45 the gift card amount will be 123.		

giftCardCount	N2	No
For a prepaid or gift cards, the total count of individual prepaid or gift cards (or codes) purchased.		
giftCardCurr	N3	No
For a prepaid or gift card purchase, the currency code of the card as defined in ISO 4217 other than those listed in the table of excluded ISO currencies in the *EMV 3-D Secure Protocol and Core Functions Specification, Version 2.1.0*.		
preOrderDate	N8	No
For a pre-ordered purchase, the expected date when the merchandise will be available. The date format is *YYYYMMDD*.		
preOrderPurchaseInd	N2	No
Specifies whether the cardholder is placing an order for a merchandise already in stock or a merchandise that will be available in the future: · 01 — merchandise available · 02 — future availability		
reorderItemsInd	N2	No

Specifies whether the cardholder is repeatedly ordering a previously purchased merchandise: · 01 — first time ordered · 02 — reordered		
shipIndicator	N2	No
Shipping method selected for the transaction. Merchants must select the **Shipping Indicator** code that most accurately describes the cardholder's specific transaction. In case of mixed goods in the cart (electronic and physical), the **Shipping Indicator** code for the physical goods is used. If all items are electronic, the **Shipping Indicator** code that describes the most expensive item is used. The accepted values are as follows: · 01 — ship to cardholder's billing address · 02 — ship to another verified address on file with merchant · 03 — ship to address that is different than the cardholder's billing address · 04 — "Ship to Store" / Pick-up at local store (Store address shall be populated in shipping address fields) · 05 — digital goods (includes online services, electronic gift cards and redemption codes) · 06 — travel and Event tickets, not shipped · 07 — other (for example, Gaming, digital services not shipped, media subscriptions, and so forth.)		

The threeDSRequestorAuthentication block parameters

Name	Type	Mandatory

threeDSReqAuthData	ANS..2048	No
Data that documents and supports a specific authentication process. In the current version of the *EMV 3-D Secure Protocol and Core Functions Specification, Version 2.1.0*., this data element is not defined in detail. The planned usage is that for each 3DS Requestor Authentication Method, this field carries data which the ACS can use to verify the authentication process. An example is provided below: · 02 — the field can carry generic 3DS Requestor authentication information. · 03 — the field can carry information about the provider of the federated ID and related information. · 04 — the field can carry the FIDO attestation data (including the signature). In future versions of the specification, these details are expected to be included.		
threeDSReqAuthMethod	N2	No

Mechanism used by the cardholder to authenticate at the 3DS Requestor: · 01 — no 3DS Requestor authentication occurred (for example, cardholder “logged in” as guest) · 02 — login to the cardholder account at the 3DS Requestor system using 3DS Requestor’s own credentials · 03 — login to the cardholder account at the 3DS Requestor system using federated ID · 04 — login to the cardholder account at the 3DS Requestor system using issuer credentials · 05 — login to the cardholder account at the 3DS Requestor system using third-party authentication · 06 — login to the cardholder account at the 3DS Requestor system using FIDO Authenticator · 07–79 — reserved for EMVCo future use (values invalid until defined by EMVCo) · 80–99 — reserved for DS use		
threeDSReqAuthTimestamp	N12	No
Date and time in UTC of the cardholder authentication. The date format is *YYYYMMDD*.		

The threeDSRequestorPriorTransactionAuthentication block parameters

Name	Type	Mandatory
threeDSReqPriorAuthData	ANS..2048	No

Data that documents and supports a specific authentication process. In the current version of the *EMV 3-D Secure Protocol and Core Functions Specification, Version 2.1.0*., this data element is not defined in detail. The planned usage is for each 3DS Requestor Authentication Method, this field carries data that the ACS can use to verify the authentication process. In future versions of the specification, these details are expected to be included.		
threeDSReqPriorAuthMethod	N2	No
Mechanism used by the cardholder previously to authenticate at the 3DS Requestor: · 01 — frictionless authentication occurred by ACS · 02 — cardholder challenge occurred by ACS · 03 — AVS verified · 04 — other issuer methods · 05–79 — reserved for EMVCo future use (values invalid until defined by EMVCo) · 80–99 — reserved for DS use		
threeDSReqPriorAuthTimestamp	N12	No
Date and time in UTC of the prior cardholder authentication. The date format is *YYYYMMDD*.		
threeDSReqPriorRef	N36	No

Additional information provided to the ACS to determine the best approach for handing a request.

This data element contains an ACS Transaction ID for a previously authenticated transaction (for example, the first recurring transaction that was authenticated by the cardholder).

5.4 Customizing payment methods

A method for processing a customer's payment is defined by the **paymentWay** parameter of a request (a registration, preauthorization registration request, or a request for an alternative payment method). The parameter value is selected from a predefined list of values that reflects available payment methods:

- CARD — payment made by entering the card details
- CARD_BINDING — payment made using a binding
- CARD_MOTO — payment made using the call center
- UPOP_MOTO (CUP UPOP MOTO) — payment made China UnionPay Express Pay
- UPOP — payment made China UnionPay Secure Pay
- FILE_BINDING — payment made using a binding uploaded in a file
- P2P — payment made when transferring funds from one card to another
- APPLE_PAY — payment made using the Apple Pay service
- MASTERPASS — payment made using a Masterpass e-wallet
- OTHER — payment used for orders processed outside EPG
- GOOGLE_PAY — payment made using the Google Pay service
- SAMSUNG_PAY — payment made using the Samsung Pay service
- NSPK_E_CERT — payment made using the NSPK Electronic Certificate
- FASTPAYMENT_QR — payment made using the FastPayment QR code
- MPU — payment via Myanmar Payment Union (MPU)
- WAVE — payment via WavePay

If you want to use custom payment methods (for example, a bank transfer), the following configuration must be applied:

- The **paymentWay** parameter that is passed as a separate parameter in a request must be set to Alternative (OTHER).
.

Note: During the registration of a payment, specifying the payment method is not mandatory but it must be specified at payment completion. If it is not specified, the appropriate payment method is automatically defined by EPG.

5.5 Response codes

An action code is a digital code of a result received after the merchant's user addressed to EPG. The codes used in the system are listed in the table below.

Action code	Error message	Description
-20010	BLOCKED_BY_LIMIT	Transaction is rejected because the amount exceeds the limit specified by the issuing bank.
-3003	Unknown	Unknown error.

-2020	Invalid ECI received	Invalid Electronic Commerce Indicator — the ECI received in PaRes is not valid for the payments network. The rule applies only to the following payments networks: · Mastercard — possible values are 01 and 02 · Visa — possible values are 05 and 06
-2018	Declined. DS connection timeout	There is no access to the Directory Server (DS) for Visa or Mastercard, or a connection error has occurred after a card involvement request (VeReq). This is an error of interaction between the SmartVista E-Commerce Payment Gateway and the payments network's servers due to technical problems with the payments network's servers.

-2017	Declined. The PARes status is not Y	Transaction is rejected. The PARes status is not set to Y.
-2016	Declined. VERes status is \"U\"	VERes status is unknown. The issuing bank could not determine if the card is enrolled in 3-D Secure.
-2013	No attempts left	All attempts allowed to perform a payment have been used.
-2011	PARes status is \"U\"	PARes status is unknown. The issuing bank was not able to perform the 3-D Secure card authorization.
-2007	Session time limit	Period of entering the payment card details has expired (by default the timeout is 20 minutes). Note: The session duration may be specified while registering an order — if the merchant has the Alternative **session timeout** permission, then the timeout duration is

		specified in the merchant settings.
-2006	TDS_AUTH_FAILED	Issuing bank rejected authentication (3-D Secure authorization has failed).
-2005	TDS_AUTH_FAILED	Issuing bank signature could not be checked (that is, the PARes was readable but the signature was incorrect); or the authentication status in PaRes was *N*.
-2003	BLOCKED_BY_PORT	Result of the fraud validation.
-2002	BLOCKED_BY_AMOUNT	Transaction was rejected because the payment amount exceeded the specified limits. Note: It may be the daily withdrawal limit specified by the acquiring bank, the limit of transactions for one card specified by the merchant, or the limit for one transaction specified by a merchant.

-2001	BLOCKED_BY_IP	Transaction is rejected because the customer's IP address is in the blacklist.
-2000	BLOCKED_BY_PAN	Transaction is rejected because the card number is in the blacklist.
-100	No payment attempted yet	There were no payment attempts.
-1	Processing not available	Timer used to wait for a processing response has expired.
0	Request processed successfully	Payment has been performed successfully.
5	Network refused transaction	Code received from SVFE in case of a failed payment.
100	Card limits exceeded	Transaction is declined because of card limits — the issuing bank has prohibited Internet transactions on the card.

101	Wrong expiry date	Transaction is declined. The card has expired.
103	Contact issuer	No connection to the issuing bank. The merchant must contact the issuing bank.
106	PIN attempts exceeded. Card is blocked.	Maximum number of PIN input attempts is exceeded. The card may be blocked temporarily.
111	Decline. Wrong PAN	Card number is incorrect.
116	Not enough money.	Transaction is declined because the amount exceeds the available balance of the selected account.
119	Security violation	Code 37 in accordance with the IBANK protocol.

120	Transaction was refused.	Transaction is not allowed by the issuing bank. The response code of the payments network is 57. The reason for rejecting the transaction should be specified by the issuing bank.
121	Limit exceeded.	Attempt has been made to perform a transaction with an amount exceeding the daily limit specified by the issuing bank.
125	Invalid card number.	Card number is incorrect. This error may have several meanings: · An attempt has been made to perform a refund of the amount exceeding the hold amount. · An attempt has been made to refund a zero amount. · The expiration date is specified incorrectly

		(for American Express only).
903	Limit exceeded.	Attempt has been made to perform a transaction with an amount that exceeds the issuing bank's limit.
904	Invalid message format	Internal system error.
910	Cannot contact bank.	Host of the issuing bank is not available.
914	Original transaction not found	Transaction is not found when sending a completion, reversal, or refund request.
999	Transaction refused by fraud.	Beginning of the transaction authorization has been missed. The transaction is declined due to fraud.
1001	EMPTY	This code is specified at the moment of the transaction's authorization, when the

		card details have not been entered yet.
1023	P2P_VISA_ALIAS_IS_NOT_ENABLED	This code is returned if operations with Visa Payment Network Alias are not enabled for EPG (the **p2p.VISA.alias.enabled** parameter in the **System settings** is not enabled).
1024	P2P_VISA_ALIAS_URL_IS_NOT_DEFINED	This code is returned if the **p2p.VISA.alias.url** parameter in the **System settings** is not configured.
1025	INTERNAL_ERROR	P2P transfer is not available for the sending country. This code is returned if the recipient payments networks supported by the merchant are not specified on the **P2P by VISA ADS parameters** tab in the merchant configuration.

1026	INTERNAL_ERROR	The P2P transfer is not available for the recipient country. This code is returned if the sending countries supported by the merchant are not specified on the **P2P by VISA ADS parameters** tab in the merchant configuration.
1027	INTERNAL_ERROR	The P2P transfer is not available for the recipient payments network. This code is returned if the recipient countries supported by the merchant are not specified on the **P2P by VISA ADS parameters** tab in the merchant configuration.
1028	INTERNAL_ERROR	Only the single P2P transfer type is applicable (either by card or by alias). This code is returned if both card and alias details are specified for the P2P transfer.

2002	Invalid operation	Incorrect transaction.
2003	SSL_FORBIDDEN	SSL (not 3-D Secure or SecureCode) transactions are prohibited for the merchant.
2004	SSL without CVC forbidden	Payment via SSL without the CVC2 is prohibited.
2005	3DS rule failed	Payment does not meet the 3-D Secure validation requirements.
2009	Refund more than deposited error	Refund amount exceeds the deposited amount.
2014	3DS rule error	3-D Secure rule execution error has occurred.
2015	Terminal select rule error	Terminal selection rule is incorrectly configured.
2016	3DS forbidden	3-D Secure authorization is impossible for the specified card and merchant.

2020	NO_ERROR	3DS2 authentication has been passed successfully.
8204	Duplicate order	Order is duplicated.
151018	CANNOT_SEND_REQUEST	Processing timeout. Sending the request has failed.
151019	RESPONSE_TIMEOUT	Processing timeout. Sending of the request is successful, however a response from the bank has not been received.
341014	GENERAL_ERROR	General error.
341015	Masterpass checkout failed	Checkout procedure has failed for a Masterpass payment.
341016	TDS2_DECLINED_BY_ARES	3DS2 authentication is declined by the Issuer's ACS in the Authentication Response (ARes).

341017	TDS2_UNKNOWN_STATUS_IN_ARES	3DS2 authentication status in ARes is unknown.
341018	TDS2_CHALLENGE_CANCELLED	3DS2 CReq (Challenge Request) has been canceled.
341019	TDS2_CHALLENGE_FAILED	3DS2 CReq failed.
341020	TDS2_UNKNOWN_STATUS_IN_RREQ	3DS2 unknown status in RReq (Result Request). The RReq message with the authentication result is sent by ACS through DS to the 3DS Server.
34031	NO_ERROR	3DS2 authentication has been passed successfully.
341021	TDS2_APP_NEED_DEVICE_INFO	3DS2 app need device info.
341022	TDS2_TRN_TIMEOUT_AT_ACS_OTHER	3DS2 timeout at ACS.
341023	TDS2_TRN_TIMEOUT_AT_ACS_CREQ_NOT_RECEIVED	3DS2 timeout at ACS, CReq not received.

341024	TDS2_ACS_MAX_CHALLENGE_EXCEEDED	3DS2 ACS maximum challenge attempts exceeded.
341025	TDS2_CARD_AUTH_FAILED	3DS2 authentication failed.
341026	TDS2_REJECTED_STATUS_IN_RREQ	3DS2 RReq status is Rejected.
341030	TDS2_AUTHENTICATION_SUCCESS	3DS2 authentication successful.
71015	DATA_INPUT_ERROR	Generic response code for wrong data that have been entered. Currently this code is returned only when PAN passed to the **paymentorder.do** API is different from PAN passed previously in the **/api/v1/3ds2/checkPreliminary** API.

AVS response codes

The following Address Verification System (AVS) Service response codes can be received from payments networks:

Code	Description

-1	Both address and postal code match.
1	Address matches but the postal code does not.
2	Postal code matches but the address does not.
3	Neither the address nor the postal code match.
50	Address verification is desired but is not available.
51	Transaction improperly requests verification.

5.6 Specifying reccurenceFrequency

The Quartz cron is used to configure the recurring payments frequency. More information is available at the following URL: http://www.quartz-scheduler.org/documentation/quartz-2.2.2/tutorials/crontrigger.html.

The value format is <* * * * * *> — a line consisting of six required time and date fields and one optional field separated by a white space. The fields are as follows:

Field position	Unit of measurement	Allowed values	Allowed special characters
First	Seconds	0-59	, - * /
Second	Minutes	0-59	, - * /

Third	Hours	0-23	, - * /
Fourth	Day-of-month	1-31	, - * ? / L W
Fifths	Month	1-12 or JAN-DEC	, - * /
Sixth	Day-of-Week	1-7 or SUN-SAT	, - * ? / L #
Seventh (optional)	Year (Optional)	empty, 1970-2199	, - * /

5.7 P2P response codes

The following response codes may be received when processing a P2P transaction:

Code	Message	Description
0	Success.	The request has been processed successfully.
12	**clientId** not equals **clientId** for this binding.	There is a mismatch of the **Client ID** value specified in the request and **Client ID** required for the binding.
14	System error.	Software or hardware issue or malfunction.

14	Session timeout.	A user's session has been finished automatically as no activity was detected after a certain period.
14	Order not found.	System is unable to find the order with the specified **MDorder** value
14	The **orderId** parameter value is wrong.	The **orderId** parameter value in the request is specified incorrectly.
14	Request IP is wrong.	An issue occurred with the IP address used in a network request.
14	Email address is wrong.	The email in the request is specified incorrectly.
14	Order is in the wrong state.	P2P transaction cannot be performed for the order in this state.
14	Card numbers are equal.	The same card number was specified for the sender and the recipient.

14	Sender’s card number is empty.	The sender’s card number was not specified in the request.
14	Sender’s card number is wrong.	The sender’s card number was specified incorrectly.
14	Sender’s PAN payment system is unknown.	The system cannot identify the payments network associated with the sender’s PAN.
14	Sender’s CVC is empty.	The sender’s CVC was not specified in the request.
14	Sender’s CVC is wrong.	The sender’s CVC was specified incorrectly.
14	Sender’s card expiration month is empty.	The sender’s card expiration month (**expirationMonth**) was not specified in the request.
14	Sender’s card expiration month is wrong.	The sender’s card expiration month (**expirationMonth**) was specified incorrectly.
14	Sender’s card expiration year is empty.	The sender’s card expiration year (**expirationYear**) was not specified in the request.

14	Sender's card expiration year is wrong.	The sender's card expiration year (**expirationYear**) was specified incorrectly.
14	Sender's name empty.	The sender's name was not specified in the request.
14	Sender's name wrong.	The sender's name was specified incorrectly.
14	Recipient's card number is empty.	The recipient's PAN was not specified in the request.
14	Recipient's card number is wrong.	The recipient's PAN was specified incorrectly.
14	Transfer amount is empty.	An error occurred during the transfer process, or the transaction was incomplete and not processed, or the **amountInput** parameter value was not specified n the request.
14	Cannot set the amount.	The **amountInput** parameter value cannot be entered as it is out of the predefined range, of funds is insufficient, or the application need to be updated.

14	Transfer amount is wrong.	The **amountInput** parameter value is incorrect.
14	Transfer currency is empty.	The **currency** parameter value was not specified in the request.
14	Transfer currency is wrong.	The **currency** parameter value is incorrect.
14	Language is wrong.	The **language** parameter value specified in the request is incorrect, or the browser or the website's settings are incorrectly configured.
14	Order number is empty.	An error occurred during the transfer process, or the transaction was incomplete and not processed, or the order ID was not specified n the request.
14	Order number is wrong.	The order ID in the request was specified incorrectly.
14	Return URL is empty.	The **returnUrl** parameter value was not specified in the request.

14	Return URL is wrong.	The **returnUrl** parameter value was specified incorrectly.
14	Transfer request parameters field **params** is wrong.	Values of some parameters in the **params** block are incorrect.
14	Session timeout seconds field is wrong.	The **sessionTimeoutSecs** parameter value was not specified in the request.
14	Payment attempts are over.	There are no more payment attempts left.
14	Verify request required.	A confirmation that the request meets certain criteria is required.
14	PAN does not match the verified one.	PAN specified in the request and the verified PAN are different.
14	The **orderId** (**mdOrder**) parameter must be specified.	The **orderId** (**mdOrder**) parameter value need to be specified in the request.
14	Order description is wrong.	The **orderDescription** parameter value is incorrect.

14	Fail URL is wrong.	The **failUrl** parameter value was specified incorrectly.
14	Wrong operation type for mandatory verification.	An issue occurred with the type of operation being performed during the verification process, often due to changes in the account or payment profile.
14	Wrong operation type or no card data.	An issue occurred with the type of operation being performed during the verification processor an issue with the card details entered or the card's ability to support the transaction.
14	Operation failed.	Transaction was unsuccessful.
14	Recipient's name field is empty.	The recipient's name was not specified in the request.
14	Recipient's name field is missing.	The recipient's name was specified incorrectly.
14	Recipient's name field length exceeded.	The specified **recipientName** parameter value exceeds this field length.

14	Recipient's name field is empty.	The recipient's name was not specified in the request.
14	Recipient's name field is missing.	The recipient's name was specified incorrectly.
14	Recipient's name field length exceeded.	The specified **recipientName** parameter value exceeds this field length.
14	Payer address field is empty.	The **payerAddress** parameter value was not specified in the request.
14	Payer address field is missing.	The **payerAddress** parameter is not available in the request.
14	Payer address field length exceeded.	The specified **payerAddress** parameter value exceeds the field length.
14	Payer city field is empty.	The **payerCity** parameter value was not specified in the request.
14	Payer city field is missing.	The **payerCity** parameter is not available in the request.

14	Payer city field length exceeded.	The specified **payerCity** parameter value exceeds the field length.
14	Payer country field is empty.	The **payerCountry** parameter value was not specified in the request.
14	Payer country field is missing.	The **payerCountry** parameter is not available in the request.
14	Payer country field length exceeded.	The specified **payerCountry** parameter value exceeds the field length.
14	Payer postal code field is empty.	The **payerPostalCode** parameter value was not specified in the request.
14	Payer postal code field is missing.	The **payerPostalCode** parameter is not available in the request.
14	Payer postal code field length exceeded.	The specified **payerPostalCode** parameter value exceeds the field length.
14	Payer state field is empty.	The **payerState** parameter value was not specified in the request.

14	Payer state field is missing.	The **payerState** parameter is not available in the request.
14	Payer state field length exceeded.	The specified **payerState** parameter value exceeds the field length.
14	Unsupported operation.	The transaction is not supported.
14	Impossible operation.	The transaction is not allowed.
14	Destination binding client id differs from client id or source binding client id.	The specified destination binding or source binding is not associated with the relevant **clientId**.
14	Source and destination bindings are the same.	The **sourceBindingId** and **destinationBindingId** parameter values are the same.
14	Destination binding is not allowed for DEBIT type.	The specified **destinationBindingId** parameter value cannot be set for the DEBIT transfer type.

14	Source binding is not allowed for CREDIT type.	The specified **sourceBindingId** parameter value cannot be set for the CREDIT transfer type.
14	Card numbers are equal.	Card numbers in the **fromCard[]**and **toCard[]**blocks are the same.
1023	P2P Visa alias is not enabled.	This code is returned if operations with Visa Payment Network Alias are not enabled for EPG (the **p2p.VISA.alias.enabled** parameter on the **System settings** page is not enabled).
1024	P2P Visa alias URL is not defined.	This code is returned if the **p2p.VISA.alias.url** parameter in the **System settings** is not configured.
1025	The P2P transfer is not available for the sending country.	This code is returned if the recipient payments networks supported by the merchant are not specified on the **P2P by VISA ADS parameters** tab in the merchant configuration.

1026	The P2P transfer is not available for the recipient country.	This code is returned if the sending countries supported by the merchant are not specified on the **P2P by VISA ADS parameters** tab in the merchant configuration.
1027	The P2P transfer is not available for the recipient payment network.	This code is returned if the recipient countries supported by the merchant are not specified on the **P2P by VISA ADS parameters** tab in the merchant configuration.
1028	Only the single P2P transfer type is applicable (either by card or by alias).	This code is returned if both card and alias details are specified for the P2P transfer.

5.8 Gathering browser information

To implement cardholder authentication based on the 3-D Secure 2 protocol, the browser information is obtained by EPG from the AReq message for an ACS to determine the ability to support authentication on a specific cardholder's browser for each transaction.

This data may be obtained by the 3-D Secure software provided to the 3DS Requestor or through remote JavaScript calls. The 3DS Server ensures that the data is not altered or hard-coded, and that it is unique for each transaction. The following specific fields are captured from the cardholder browser for each transaction:

- Browser Accept Headers
- Browser IP Address
- Browser Java Enabled

- Browser Language
- Browser Screen Color Depth
- Browser Screen Height
- Browser Screen Width
- Browser Time Zone
- Browser User-Agent

An additional method introduced in EPG to support the 3-D Secure 2 protocol is **get3DS2Urls**. The method is transferred using the payment.js script delivered to the cardholder as a standard JavaScript handling payment page.

Browser data flow

The browser data flow is depicted in the diagram below.

The data processing sequence is as follows:

1. The cardholder's browser through the **get3DS2Urls** method obtains **threeDSMethodURL** and **threeDSMethodURLServer** from EPG.
2. The **get3DS2Urls** method creates two iframes to support gathering the browser information:
   - An iframe to submit data to 3DSS using **threeDSServerTransID**
   - An iframe to submit data to ACS using **threeDSMethodData**
3. Through the iframes, data is exchanged with 3DSS and ACS, which send scripts to collect the cardholder browser information:
   - From 3DSS, the **/api/v1/client/gather** method sends an HTML page to the cardholder browser with a script (cardholderinfo.html). After that the information is processed with the **/api/v1/client** method.
   - From ACS, the **/3ds2/threeDSMethod** method sends threeDSMethod.html to the cardholder browser to gather the browser information. The ACS receives browser information from the **/3ds2/storeClientInfo** endpoint.
4. The ACS sends a notification to 3DSS that the browser information has been is obtained. To notify the 3DS Server that the ACS has collected information about

the browser, 3DS Server is required to provide **threeDSServerTransID** to **/acs/notification** API.

Note: The timeout for 3DS Server to wait to receive the cardholder data is specified by the **dataCollectionTimeout** parameter.

Model

The collected browser information includes the following parameters:

Name	Type	Mandatory
userAgent	String	Yes
Information the cardholder's web browser sends in the **User-Agent** HTTP header when making requests to websites. It is a string containing information about the cardholder's browser, operating system, device type, and other details. The **User-Agent** format varies for different browsers.		
fingerprint	String	No
Information collected from the device browser for subsequent identification.		
OS	String	No
Operating system used by the cardholder's device.		
OSVersion	String	No

Version of the operating system used by the cardholder’s device.		
device	String	No
Information about the cardholder’s device (model, version, and so on).		
deviceType	String	No
Type of device on which the browser is running (mobile phone, desktop, tablet, and so on).		
isMobile	Boolean	No
Specifies whether the cardholder’s device is mobile: · true · false		
screenPrint	String	No
Information about the screen resolution of the cardholder’s device.		
colorDepth	String	Yes
Bit depth of the color palette for displaying images on the screen of the cardholder’s device.		

screenHeight	Integer	Yes
Height of the cardholder's device screen.		
screenWidth	Integer	Yes
Width of the cardholder's device screen.		
plugins	String	No
List of plug-ins installed on the cardholder's device browser.		
javaEnabled	Boolean	Yes
Specifies whether Java support is enabled for the cardholder's browser. · true · false		
browserLanguage	String	Yes
Language of the cardholder's browser.		
browserTimeZone	String	Yes
Time zone of the cardholder's browser.		

browserTimeZoneOffset	Integer	No
Difference between UTC time and the cardholder's browser local time, in minutes.		
browserAcceptHeader	String	Yes
Specifies the following: · The browser to which a request must be sent. · File formats (MIME-types) accepted by the server as a response to the browser that sends a request. **browserAcceptHeader**: application/json, text/javascript, */*;		
browserIpAddress	String	Yes
IP address of the cardholder's browser.		

API integration

API integration is allowed for merchants. In this case, it is necessary to obtain the browser information from the merchant site by any possible method and provide it to 3DS Server through the **/api/v1/client** API method.

Note: To generate AReq data and prepare it to be sent, it is necessary to have the browser information gathered and the **acsMethodAvailable** values be set to Y or U.

```
public boolean isDataCollected() {
    return cardholderInfo != null && (acsMethodAvailable == Y || acsMethodAvailable == U);
}
```

Example of an iframe

```
getUrlsAction: "../../../epg/rest/threeds2/getUrls.do",

 get3DS2Urls: function() {
            if (!properties.tryGet3DS2Urls) {
            return;
            }
            $.ajax({
            url: settings.getUrlsAction,
            type: 'POST',
            cache: false,
            data: ({mdOrder: properties.orderId}),
            dataType: 'json',
            error: function(){
            return true;
            },
            success: function(data) {
            if (data['completed'] !=
                  null && !JSON.parse(data['completed'])) {
              setTimeout(methods.get3DS2Urls, 500);
            } else {
            if (data['is3Ds2Eligible'] !=
            null && JSON.parse(data['is3Ds2Eligible'])) {
                        var tdssFrame = document.createElement('iframe');
                  tdssFrame.name = 'tdss_' + Date.now();
                  tdssFrame.src = '#';
                  tdssFrame.style.display = 'none';
                  tdssFrame = document.body.appendChild(tdssFrame);
                        var tdssForm = document.createElement('form');
                  tdssForm.target = tdssFrame.name;
                  tdssForm.method = 'post';
                  tdssForm.action = data['threeDSMethodURLServer'];
                  tdssForm = tdssFrame.appendChild(tdssForm);
                  tdssForm.submit();

                        if (data['threeDSMethodURL'] !=
                  null && data['threeDSMethodURL'] !=='null') {
                     var iframe = document.createElement('iframe');
                     iframe.name = 'acs_' + Date.now();
```

```
                iframe.src = '#';
                iframe.style.display = 'none';
                iframe = document.body.appendChild(iframe);
                var form = document.createElement('form');
                form.target = iframe.name;
                form.method = 'post';
                form.action = data['threeDSMethodURL'];
                var param = document.createElement('input');
                param.type = 'hidden';
                param.name = 'threeDSMethodData';
                param.value = data['threeDSMethodDataPacked'];
                param = form.appendChild(param);
                form = iframe.appendChild(form);
                form.submit();
                  }
        }
        }
        }
        });
},
```

Example of the HTML script for gathering browser information

```
<!DOCTYPE html>
 <html lang="en">
 <head>
        <meta charset="UTF-8">
        <script>
        // browser info gathering script
        </script>
        <script>
      window.addEventListener('load', function () {
        function encodeForm(data) {
          let urlEncodedDataPairs = [];

        for (let name in data) {
          urlEncodedDataPairs.push(encodeURIComponent(name) +
  '=' + encodeURIComponent(data[name]))
        }
```

```
        return urlEncodedDataPairs.join('&').replace(/%20/g, '+')
        }

        const threeDSServerTransID = "${threeDSServerTransID}";
        const client = new ClientJS();
        const screen = client.getCurrentResolution().split("x");
        const info = {
        userAgent: client.getUserAgent(),
          //fingerprint: client.getFingerprint(),
        OS: client.getOS(),
          OSVersion: client.getOSVersion(),
        device: client.getDevice(),
          deviceType: client.getDeviceType(),
          isMobile: client.isMobile(),
          screenPrint: client.getScreenPrint(),
          colorDepth: client.getColorDepth(),
          screenHeight: screen && screen[1] ? screen[1] : 0,
          screenWidth: screen && screen[0] ? screen[0] : 0,
          plugins: client.getPlugins(),
          javaEnabled: client.isJava(),
          browserLanguage: client.getLanguage(),
          browserTimeZone:
          Intl.DateTimeFormat().resolvedOptions().timeZone,
          browserTimeZoneOffset:
        new Date().getTimezoneOffset()
        };

        const dataForm = encodeForm({
          threeDSServerTransID: threeDSServerTransID,
          clientInfo: JSON.stringify(info, null, 2)
        });
        const request = new XMLHttpRequest();
       request.open("POST", "${storeInfoURL}");
       request.setRequestHeader('Content-Type',
      'application/x-www-form-urlencoded');
       request.send(dataForm)
       })
       </script>
</head>
<body>
```

</body>
</html>

5.9 Business application identifiers

The following business application identifiers are used for Visa:

Value	Application type
AA	Account to Account
BB	Business to Business
BI	Money transfer (bank-initiated)
CP	Card Bill Payment
FD	Funds Disbursement (general)
FT	Funds Transfer
GD	Government Disbursement
GP	Gambling Payment (other than online gambling)
LO	Loyalty and Offers

BP	Non-card Bill Payment
MI	Merchant Initiated Money Transfer
CI	Cash In
CO	Cash Out
MP	Face-to-face Merchant Payment
MD	Merchant Disbursement
OG	Online Gambling Payout
PD	Payroll/Pension Disbursement
PP	Person to Person
TU	Prepaid Reload/Top Up
WT	Wallet Transfer
PS	Payment for Goods and Services

CD	Cash Deposit

5.10 Passing 3DS authentication result from external MPI or 3DSS

The following parameters must be configured if it is necessary to pass the 3DS authentication results from an external MPI or 3DSS to EPG. Merchants can pass these authentication results if they use a third-party MPI or 3DSS.

ATTENTION: The **Use external MPI** option in the merchant's configuration must be enabled to allow a merchant to use an external MPI.

Parameter	Description	Mandatory
EXTERNAL_MPI_RES	Possible values: · true — authentication results are passed. · false — authentication results are not passed. If this parameter is missing or any other value is specified, all other parameters of 3DS authentication result are not processed and the authentication results are not passed.	Yes
3ds_protocol	3DS protocol version: · 1.0.2 · 2.1.0 · 2.2.0	Yes

eci	ECI version: · 01 · 02 · 05 · 06 · 07 · N2	Yes
cavv	Authentication value received from the external MPI or 3DSS.	Parameter must be present for the following **eci** values: · 02 · 05 · N2
xid	XID value for 3DS1 transactions.	Parameter must be present if **3ds_protocol** = 1.0.2.
3ds2_ds_trans_id	**dsTransID** value for a 3DS2 transaction.	Parameter must be present for a 3DS2 transaction if **3ds_protocol** = 2.1.0 or 2.2.0.
3ds2_3dss_trans_id	**threeDSServerTransID** value for a 3DS2 transaction.	Parameter must be present for a 3DS2 transaction

		if **3ds_protocol** = 2.1.0 or 2.2.0.

5.11 ClientBrowserInfo structure

The **clientBrowserInfo** parameter contains the following structure:

Name	Type	Mandatory
userAgent	String	Yes
Information the cardholder's web browser sends in the **User-Agent** HTTP header when making requests to websites. It is a string containing information about the cardholder's browser, operating system, device type and other details. The **User-Agent** format varies for different browsers.		
colorDepth	String	Yes
Bit depth of the color palette for displaying images on the cardholder's device screen.		
screenHeight	Integer	Yes
Height of the cardholder's device screen.		
screenWidth	Integer	Yes

Width of the cardholder's device screen.		
javaEnabled	Boolean	Yes
Specifies whether supporting Java is enabled for the cardholder's browser: · true · false		
browserLanguage	String	Yes
Language of the cardholder's browser.		
browserTimeZoneOffset	Integer	Yes
Difference between UTC time and the cardholder's browser local time, in minutes.		
browserAcceptHeader	String	Yes
Specifies the following: · The browser to which a request must be sent. · File formats (MIME-types) accepted by the server as a response to the browser that sends a request.		
browserIpAddress	String	Yes

IP address of the cardholder’s browser.		
fingerprint	String	No
Information collected from the device browser for subsequent identification.		
OS	String	No
Operating system used by the cardholder’s device.		
OSVersion	String	No
Version of the operating system used by the cardholder’s device.		
device	String	No
Information about the cardholder’s device (model, version, and so on).		
deviceType	String	No
Type of device on which the browser is running (mobile phone, desktop, tablet, and so on).		
isMobile	Boolean	No

Specifies whether the cardholder's device is mobile: · true · false		
screenPrint	String	No
Information about the cardholder's device screen resolution.		
plugins	String	No
List of plug-ins installed on the cardholder's device browser.		
browserTimeZone	String	No
Time zone of the cardholder's browser.		
javascriptEnabled	Boolean	Yes
Specifies whether JavaScript is enabled for the cardholder's browser: · true · false		

An example of the browser data is provided below:

```
{
 "userAgent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
            Chrome/69.0.3497.100 Safari/537.36",
 "fingerprint": 2577441658,
 "OS": "Windows",
 "OSVersion": "10",
 "isMobile": false,
 "screenPrint": "Current Resolution: 1536x864, Available Resolution: 1536x834,
  Color Depth: 24, Device XDPI: undefined, Device YDPI: undefined",
 "plugins": "Chrome PDF Plugin, Chrome PDF Viewer, Native Client",
 "javaEnabled": false,
 “javascriptEnabled": true,
 "browserLanguage": "en-US",
 "browserTimeZone": "Europe/Moscow"
 }
```

5.12 AReqFieldsOverride structure

The **aReqFieldsOverride** parameter contains the following structure:

Name	Type	Mandatory
mcc	AN..4	Yes
A four-digit merchant category code to be passed to 3DSS in **AReq.mcc**.		
threeDSRequestorURL	AN 1..512	Yes

URL to be passed to 3DSS in **AReq.threeDSRequestorURL**. This data element provides additional information to the receiving 3-D Secure system if a problem arises and should provide contact information. In practice, payments networks expect a merchant URL.		
acquirerMerchantID	N..35	Yes
Acquirer merchant identifier to be passed to 3DSS in **AReq.acquirerMerchantID**. It is a string whose maximum length is 35 characters.		
threeDSRequestorName	AN..40	Yes
3DS Requestor name to be passed to 3DSS in **AReq.threeDSRequestorName**. It is a string whose maximum length is 40 characters.		
merchantName	AN..40	Yes
Merchant name to be passed to 3DSS in **AReq.merchantName**. It is a string whose maximum length is 40 characters.		
merchantCountryCode	AN..3	Yes
Three-digit merchant country code according to ISO 3166. This code is to be passed to 3DSS in **AReq.merchantCountryCode**.		

5.13 ParamNames parameters list

The description of the parameters that can be passed in the **paramNames** block of the **getSessionStatus.do** method is provided in the table below.

currency	
Description	Currency code in the ISO 4217 format.
currentState	
Description	Code of the current state of the transaction.
date	
Description	Date when the transaction payment was completed, in the following format: *dd.MM.yyyy HH:mm:ss*.
depositedAmount	
Description	Deposited amount.
depositFlag	
Description	Specifies whether this is a two-phase transaction.
eci	

Description	ECI (Electronic Commerce Indicator), as defined by 3-D Secure. The card network may add an ECI to the card data. This indicator is then included in the payment token. If you receive an ECI, you must pass it on to your payment processor; otherwise, the transaction fails.
fraud	
Description	Informs the customer about a potentially fraudulent transactions (FRAUD or POSSIBLE_FRAUD).
ip	
Description	IP address of the customer.
ipCountryCode	
Description	Country code of the customer.
maskedPan	
Description	Masked number of the card (according to the specified masking rules).
mdOrder or **(mdorder)**	

Description	Number of the order in the payments system.
merchantFullName	
Description	Full name of the merchant.
merchantLogin	
Description	Login of the merchant.
orderDescription	
Description	Description of the order.
orderNumber	
Description	Number of the order in the merchant’s system.
panCountryCode	
Description	Country code of the customer’s card.
paymentRefNum	

Description	Authorization reference number.
paymentState	
Description	Transaction status: · Started · Payment_approved · Payment_declined · Payment_void · Payment_deposited · Refunded
paymentWay	
Description	Payment method.
processingId	
Description	Identifier of the merchant in the processing center.
refNum or **(refnum)**	
Description	Payment reference number.
terminalId	

Description	Terminal identifier.
merchantUrl	
Description	URL of the merchant site (online store).
amountFormatted	
Description	Formatted registration amount.
approvedAmountFormatted	
Description	Confirmed formatted amount to charge the customer.
depositedAmountFormatted	
Description	Formatted amount to be charged.
currencyName	
Description	Name of the currency in the ISO 4217 format.
cardholderName	

Description	Name of the cardholder.
expiry	
Description	Payment card expiration date, in the following format: *YYYYMM*.
pan	
Description	Number of the card.
creditPan	
Description	Recipient card number for a P2P money transfer.
debitPan	
Description	Source card number for a P2P money transfer.
creditBankName	
Description	Bank name for a recipient card for a P2P money transfer.
creditPanCountryCode	

Description	Country code of the recipient card for a P2P money transfer.
recipientData	
Description	Data on the recipient of a P2P money transfer.

5.14 CustomerBillingAddress parameters

The description of the parameters that can be passed in the **customerBillingAddress** block is provided in the table below.

Name	Type	Mandatory
billAddrCity	AN..40	No
City of the customer's billing address associated with the card used for the purchase.		
billAddrCountry	AN..3	No
Country of the customer's billing address associated with the card used for the purchase. It is an ISO 3166-1 numeric three-digit country code.		
billAddrLine1	ANS..99	No

First line of the street address or equivalent local portion of the customer's billing address associated with the card used for the purchase.		
billAddrLine2	ANS..99	No
Second line of the street address or equivalent local portion of the customer's billing address associated with the card used for the purchase.		
billAddrLine3	ANS..99	No
Third line of the street address or equivalent local portion of the customer's billing address associated with the card used for the purchase.		
billAddrPostCode	ANS..10	No
Zip or other postal code of the customer's billing address associated with the card used for the purchase.		
billAddrState	ANS..99	No
Sate or province of the customer's billing address associated with the card used for the purchase.		

5.15 AirlineData parameters

The description of the parameters that can be passed in the **airlineData** block of the **register.do** method is provided in the table below.

Name	Type	Mandatory
ticketNumber	ans 15	Yes
Ticket number.		
numberOfLegs	n 1 (1-4)	Yes
Number of travel legs. Information about each leg is provided in the **legs[]** block.		
travelAgencyCode	ans...8	No
Code of the travel agency where the ticket was purchased.		
travelAgencyName	ans...25	No
Name of the travel agency where the ticket was purchased.		
passengerName	ans...29	Yes
Passenger’s name.		
ticketIssueDate	n 8	No

Ticket issue date in the following format: *DDMMYYYY*.		
ticketIssueAddress	ans 16	No
Address of the agency where the ticket was issued.		
totalFare	n 12	No
Total amount of fares.		
totalFees	n 12	No
Total amount of fees.		
totalTaxes	n 12	No
Total amount of taxes.		
restrictedTicketIndicator	n 1	No
Restricted ticket Indicator: · 0 — unrestricted ticket · 1 — restricted ticket		

bookingReferenceNumber	ans 20	Yes
Booking Reference Number.		
issuingCarrierCode	ans 4	No
Issuing carrier code.		
carrierName	an 19	No
Air carrier name.		
planNumber	an 2	No
Plain number.		
invoiceNumber	an 6	No
Invoice number.		
originalCurrencyCode	n 3	No
Original currency code.		

originalInvoiceAmount	n 12	No
Original invoice amount.		
legs[]		Yes
Information about trip legs. See the description of the parameters in the table below.		

The following fields are available for each leg in the **legs[]** block:

Name	Type	Mandatory
departureAirportCode	an 3	Yes
Departure airport code.		
carrierCode	an 2	Yes
Air carrier code. It is specific to the airline on which the passenger is flying.		
flightNumber	ans 5	No
Leg flight number.		

fareBasis	an 6	No
Core identifying a fare type. It is specific to the airline on which the passenger is flying.		
serviceClassCode	an 2	No
Leg service class code. It is specific to the airline on which the passenger is flying		
stopOverCode	an 1	No
Flight stop-over code. It is specific to the airline on which the passenger is flying.		
destinationAirportCode	an 3	Yes
Destination city/airport code.		
departureDate	n 8	Yes
Leg departure date in the following format: *DDMMYYYY*.		
departureTime	n 6	No
Leg departure time in the following format: *HHmmss*.		

conjunctionTicketNumber	an 15	No
Leg conjunction ticket number.		
exchangeTicketNumber	an 15	No
Leg exchange ticket number.		
couponNumber	ans 1	No
Leg coupon number.		
arrivalTime	n 6	No
Leg arrival time in the following format: *HHmmss*.		
fare	n 12	No
Leg fare amount.		
fees	n 12	No
Leg fee amount.		

taxes	n 12	No
Leg tax amount.		
endorsementsRestrictions	ans 20	No
Leg endorsements or restrictions.		
departureTax	n 12	No
Departure tax.		

5.16 PaymentToken block

The following parameters are transferred in the **paymentToken** block.

Name	Type	Mandatory
protocolVersion	ANS	Yes
Encryption or signing scheme under which the message is created. It enables the protocol to evolve over time, if required.		
signature	ANS	Yes

Verifies that the message came from Google. It is created with ECDSA by the intermediate signing key.		
intermediateSigningKey[]		Yes
JSON object that contains the intermediate signing key from Google. It contains the **signedKey** with **keyValue**, **keyExpiration**, and signatures. It is serialized to simplify the intermediate signing key signature verification process. See the intermediateSigningKey block description.		
signedMessage		Yes
JSON object serialized as a string that contains the **encryptedMessage**, **ephemeralPublicKey**, and tag. It is serialized to simplify the signature verification process.		

5.17 AdditionalParameters block

Parameters of the **additionalParameters** block have the following format:.

Name	Type	Mandatory
name	ANS..255	Yes
Name of an additional parameter.		
value	ANS..1024	Yes

Value of the additional parameter.

5.18 OrderStatus block

The following parameters are transferred in the **orderStatus** block.

Name	Type	Mandatory
errorCode	N 1..3	No
Response code: · 0 — a successful transaction · Any other number — an error occurred when processing the request		
errorMessage	AN 1..512	No
Description of the error in the language that was sent in the **language** parameter of the request.		
orderNumber	ANS 1..32	Yes
Number (identifier) of the order in the merchant's online store system. It is unique for every store in the system and is generated on the order registration. If the **Require system to generate order numbers** permission is enabled for the merchant, the order number is generated automatically. Otherwise, the order number data is sent in the API.		

orderStatus	N2	No
Order status in SmartVista E-Commerce Payment Gateway. The value is selected from the variants listed below. It is absent if a matching order was not found. The possible values of the field are listed in the orderStatus values table below.		
actionCode	N3	Yes
Processing system authorization code.		
actionCodeDescription	AN..512	Yes
Action code description in the language specified in the **Language** parameter.		
amount	N 1..19	Yes
Order amount in the minor denomination (for example, cents).		
currency	N3	Yes
Payment currency code in the ISO 4217 format.		
date	ANS	Yes

Date of order registration.		
merchantOrderParams[]		No
Tag containing attributes that contain additional merchant parameters. See the merchantOrderParams block.		
attributes[]	Not applicable	Yes
Attributes of the order in the payments system (order number). See the attributes block.		
cardAuthInfo[]	Not applicable	No
Tag containing the payment attributes. See the cardAuthInfo block.		

5.19 Attributes block

Parameters of the **attributes** block have the following format.

Name	Type	Mandatory
name	AN7	Yes

Attribute name is **mdOrder**.		
value	ANS36	Yes
Order number in the payments system (it is unique in the system).		

5.20 MerchantOrderParams block

The format of the **merchantOrderParams** block parameters is described below.

Name	Type	Mandatory
name	AN..20	Yes
Name of the additional merchant parameter.		
value	AN..1024	Yes
Value of the additional merchant parameter.		

5.21 CardAuthInfo block

The following parameters are transferred in the **cardAuthInfo** block.

Name	Type	Mandatory

pan	N 13...19	No
Masked number of the card that has been used for the payment. It is only specified for paid orders.		
expiration	N6	No
Card expiration date in the following format: *YYYYMM*. It is only specified for paid orders.		
cardholderName	ANS 2..26	No
Name of the cardholder. This parameter is verified according to the following criteria: · Acceptable characters are: Latin letters, 0-9, $,), (, -, . , a space · Cardholder name must start with a letter · Minimum length: 2 characters · Maximum length: 26 characters · Null is valid · Uppercase and lowercase are acceptable		
authorizationResponseId (the deprecated name is **approvalCode**)	AN6	No
Payments network authorization code. The field has a fixed length of six characters; it can contain both numbers and letters.		

secureAuthInfo[]	Not applicable	No
Tag containing information about secure authentication. The parameters of the secureAuthInfo[] block are detailed below.		
authenticationIndicator	String	No
3DS authentication indicator that specifies the type of 3DS authentication used for the transaction. Possible values: · THREEDS_1_Y — SCA Cardholder authentication with 3DS 1.x · THREEDS_1_A — Cardholder authentication attempt with 3DS 1.x · THREEDS_2_Y — SCA Cardholder authentication with 3DS 2.x · THREEDS_2_F — RBA Cardholder authentication with 3DS 2.x · THREEDS_2_A — Cardholder authentication attempt with 3DS 2.x Note: This parameter is only applicable for successfully 3DS authenticated transactions, and is used for **Merchant.OrderStatusExtendedVersion.VERSION_13** or a higher version.		

secureAuthInfo[] block parameters

Note: To include the **secureAuthInfo** block in the method response, enable the **Receive 3DS requisites of transactions** option for the merchant.

Name	Type	Mandatory

eci	AN2	No
Electronic Commerce Indicator.		
cavv	ANS..200	No
Cardholder Authentication Verification Value.		
xid	ANS..80	No
Electronic Commerce Transaction Identifier.		
transStatus	String	No
Value of **ARes.transStatus** (for frictionless authentication) or **RReq.transStatus** (for challenge authentication). For possible values, see the EMVCo specification. Note: This parameter is used for **Merchant.OrderStatusExtendedVersion.VERSION_13** or a higher version.		
transStatusReason	String	No
Value of **ARes.transStatusReason** (for frictionless authentication) or **RReq.transStatusReason** (for challenge authentication). For possible values, see the EMVCo specification.		

Note: This parameter is used for **Merchant.OrderStatusExtendedVersion.VERSION_13** or higher version.

5.22 IntermediateSigningKey block

The following parameters are transferred in the **intermediateSigningKey** block.

Name	Type	Mandatory
signedKey	Object	Yes
JSON object that contains the intermediate signing key from Google. It is serialized to simplify the intermediate signing key signature verification.		
signatures	Object	Yes
Signatures for the intermediate signing key.		

5.23 PaymentMethodDetails block

The following parameters are transferred in the **paymentMethodDetails** block.

Name	Type	Mandatory
authMethod	String	Yes

Mechanism to authenticate the cardholder at the 3DS Requestor.		
pan	N 13...19	Yes
Card number.		
expirationMonth	N2	Yes
Month when the card validity period expires, in the following format: *MM*.		
expirationYear	N4	Yes
Year when the card validity period expires, in the following format: *YYYY*.		
eciIndicator	N2	No
ECI indicator. This parameter is presents if **`authMethod=`**`CRYPTOGRAM_3DS`.		
cryptogram	ANS28	No
Cryptogram. This parameter is present if **authMethod=CRYPTOGRAM_3DS**.		

5.24 Available merchant options

The following options can be enabled for merchants:

Name	System name
Airline data allowed	AIRLINE_DATA_ALLOWED
Alternative session timeout	ALTERNATIVE_SESSION_TIMEOUT
Bindings deactivation on a payment page is allowed	BINDING_DEACTIVATION_ALLOWED
Can create and update bindings without payment	BINDING_WITHOUT_PAYMENT_ALLOWED
Can create/update zero-amount bindings without CVV2/CVC2	BIND_WITHOUT_CVC
Can get bindings by card number	GET_BINDINGS_BY_CARD_NUMBER_OR_BINDING_ID

Can pass fiscal data	FISCAL_DATA_ALLOWED
Can pay by binding without CVV2/CVC2	BINDING_PAY_WITHOUT_CVC
Can pay by card without CVV2/CVC2	PAY_WITHOUT_CVC
Can pay with Apple Pay	USE_APPLEPAY
Can pay with Apple Pay decrypted	USE_APPLEPAY_DECRYPTED
Can pay with external fee	EXTERNAL_FEE_ALLOWED
Can pay with Google Pay by card	USE_GOOGLEPAY_CARD
Can pay with Google Pay by token	USE_GOOGLEPAY_TOKEN

Can pay with Google Pay decrypted by card	USE_GOOGLEPAY_DECRYPTED_CARD
Can pay with Google Pay decrypted by token	USE_GOOGLEPAY_DECRYPTED_TOKEN
Can pay with Mastercard SCoF by token decrypted	USE_MASTERCARD_SCOF_TOKEN_DECRYPTED
Can pay with Mir Pay by token	USE_MIRPAY_TOKEN
Can pay with Samsung Pay	USE_SAMSUNGPAY
Can pay with Samsung Pay decrypted	USE_SAMSUNGPAY_DECRYPTED
Can pay with Yandex Pay by card	USE_YANDEXPAY_CARD

Can pay with Yandex Pay by card decrypted	USE_YANDEXPAY_CARD_DECRYPTED
Can pay with Yandex Pay by token	USE_YANDEXPAY_TOKEN
Can pay with Yandex Pay by token decrypted	USE_YANDEXPAY_TOKEN_DECRYPTED
Can perform ordinary payment operations	PURCHASE_ALLOWED
Can register and pay orders in console	REGISTER_AND_PAY_IN_CONSOLE
Can register in Mir In-App	CAN_REGISTER_IN_MIR_IN_APP
Can register orders in console	REGISTER_IN_CONSOLE

Can send a link to the payment/binding form via email	SEND_PAYMENT_FORM
Can send a link to the payment/binding form via SMS	SEND_PAYMENT_FORM_SMS
Can use 'standard' P2P transfers	P2P_ALLOWED
Can use both external and internal fees in one transaction	CAN_USE_INTERNAL_AND_EXTERNAL_FEE_SIMULTANEOUSLY
Can use CUP UPOP SecurePay	UPOP_PAYMENT_ALLOWED
Can use DCC	DCC_ALLOWED
Can use Delayed-Charge payments	DELAYED_CHARGE_PAYMENT_ALLOWED

Can use enrollment verify service	VERIFY_ENROLLMENT
Can use Meeza RTP payments	MEEZA_RTP_ALLOWED
Can use MNO GENERIC payments	MNO_GENERIC_ALLOWED
Can use MPU Payments	MPU_ALLOWED
Can use NSPK E-Cert	NSPK_E_CERT_ALLOWED
Can use P2P Credit operations	P2P_CREDIT_ALLOWED
Can use P2P Debit operations	P2P_DEBIT_ALLOWED
Can use QR Fast Payments (SBP)	QR_FAST_PAYMENT_SPB_BRS_ALLOWED

Can use SV IPS Account/Wallet number Payments	SV_IPS_ACCOUNT_PAYMENT_ALLOWED
Can use SV IPS Alias Payments	SV_IPS_ALIAS_PAYMENT_ALLOWED
Can use SV IPS QR payments	SV_IPS_QR_PAYMENT_ALLOWED
Can use WAVE Payments	WAVE_ALLOWED
Configure and Transfer ISO fields to OLTP system (SVFE)	TRANSFERRING_PARAMS_TO_SV_ALLOWED
Credit Account verification allowed	CREDIT_ACCOUNT_VERIFICATION_ALLOWED
Debit Account verification allowed	DEBIT_ACCOUNT_VERIFICATION_ALLOWED

Deposit with amount exceeding registration amount	DEPOSIT_CAN_BE_EXCEEDED
Display Acquirer and Issuer fees for P2P payments	P2P_DISPLAY_ACQUIRER_AND_ISSUER_FEE
Enable address verification service (AVS)	AVS_ENABLED
Enable iReq-SSL mode for iReq errors	IREQ_SSL_ENABLED
Enable P2P by Visa Alias Directory Service (ADS)	ENABLE_P2P_BY_VISA_ADS
Fee calculation allowed	FEE_CALCULATION_ALLOWED
Handle 3DS2 method in EPG	HANDLE_3DS2_METHOD_IN_EPG

Inherit parent's lists	INHERIT_PARENT_LIST
Installment payments allowed	INSTALLMENT_ALLOWED
MasterPass checkout allowed	MASTERPASS_ALLOWED
Merchant allow to process 2-phase payment	TWO_PHASE_PAYMENT_ALLOWED
Merchant is allowed to process payment with delayed clearing	DELAYED_CLEARING_ALLOWED
Merchant is allowed to use alternative payment methods	PAY_BY_OTHER_WAY

Merchant is allowed to use bindings	BINDING_ALLOWED
Only Full 3DS payments allowed	ACCEPT_ONLY_ECI_Y
Overwrite data when parameter names are equal	REMAP_REQUEST_PARAMS
Payment form on merchant side allowed	MERCHANT_SIDE_PAYMENT_FORM_ALLOWED
Payment if ARes finished with status=U for CUP	TRY_SSL_IF_U_IN_ARES_UPI
Payment if RReq finished with status=U for CUP	TRY_SSL_IF_U_IN_RREQ_UPI

Payment with Tips NSPK allowed	NSPK_PAYMENTS_WITH_TIPS
Receive 3DS requisites of transactions	REPLY_3DS_DATA
Recurrent payment with fluctuated amount allowed	RECURRENT_WITH_FLUCTUATE_AMOUNT_ALLOWED
Recurrent payments allowed	RECURRENT_ALLOWED
Redirect client to custom payment page	CUSTOM_PAYMENT_PAGE
Refunds allowed	REFUNDS_ALLOWED
Require system to generate order numbers	GENERATE_ORDERNUMBER

Rules can be applied to the merchant	RULES_APPLIED
Sending callback notification is allowed	CALLBACK_OPERATIONS
Sending order description to SV is allowed	SEND_ORDER_DESC_TO_SV
Sending order number to SV is allowed	SEND_ORDER_NUMBER_TO_SV
Show finish payment page	USE_GENERIC_FINISH_PAYMENT_PAGE
Submerchants allowed	SUBMERCHANTS_ALLOWED
Use account updater	USE_ACCOUNT_UPDATER
Use external MPI	USE_EXTERNAL_MPI

Use NSPK decision system	USE_NSPK_DECISION_SYSTEM
Use TDSecAuthorizer	USE_TDS_AUTHORIZER
Wallet payments without order registration allowed	WALLET_PAYMENT_WITHOUT_ORDER_REGISTRATION

6. Glossary

Term	Description
3-D Secure (Verified by Visa)	Technology developed by Visa that enables additional authorization of a customer on the side of the issuing bank.
ACS	Access Control Server. It is an element of 3-D Secure infrastructure that ensures the validation of the payer by the issuing bank.
Bank	Acquiring bank that implements and runs the SmartVista E-Commerce Payment Gateway.

BIN	Bank Identification Number. The first six digits of the card number.
Merchant	Trade and Service Company (TSC) that sells goods or provides services via the Internet.
Mobile payment page	The functionality of the mobile and desktop payment pages is the same. The mobile version differs in the layout that takes into account the requirement for a compact and high-quality display of the page on the screens of mobile devices.
One-phase payment	Transaction for paying for goods or services performed with a card via the Internet. It does not require additional confirmation.
Order	Basic system entity that describes an order in an Internet store. Each order is characterized by an amount.
PAN	Card number (13 to 19 digits).
Payment details	Details used by a customer to pay for an order. Typically, details include card number, expiration date, CVC, and so on.
Payment form	HTML page that customers use to specify payment properties.

Payment gate of Bank-acquirer (PG)	Automated system that enables a merchant to receive payments and a customer to send payments via the Internet.
PCI SSF	Payment Card Industry Software Security Framework, a collection of security standards and associated validation and listing programs. PCI SSF includes the following standards: · Secure Software Standard (SSS) — a set of requirements outlined in the PCI SSF Framework that required Payment Software Vendors to validate in order to qualify as a Validated Payment Software or Listed Payment Software by the PCI Security Standard Council. Validating the Payment Software ensures a secure development of the application as per the industry best standards and practices. The validation assures that the Payment Software is securely developed to protect the integrity of the software and the confidentiality of sensitive data it stores, processes, and transmits. · Secure Software Lifecycle (Secure SLC) Standard — a standard developed and focused on ensuring that the software vendor's software development process, methodologies, and practices are secure. Evaluating the software lifecycle management practices against the Secure SLC Standard demonstrates the organization has mature software development practices and can develop secure payment software.

Refund	Partial or full refund to the customer's card in case of a refusal of requested goods (or services) or a refund of goods (or services). This transaction is possible after a deposit.
Reversal	Unlocking of funds on the payer's account. This feature is available for a limited time based on the terms specified by the bank.
SecureCode	Technology developed by Mastercard to enable additional authorization of a customer on the side of the issuing bank. It is practically the same as the 3-D Secure technology.
SSL payment	Payment for goods or services performed without the use of 3-D Secure technology.
Two-phase payment	Payment for goods or services performed via the Internet using a bank card. This type of payment requires additional confirmation. The two-phase mechanism splits the process into checking whether the card is capable of paying (authorization) and debiting the money from the account (financial confirmation). During the first step of a two-phase payment, the bank card paying capacity is checked and money on the customer's account are put on hold. The second step is a confirmation for of the funds transfer.

EPG - Merchant Integration Guide