**SPECIFICATION**

**PaySpot web payment service**

**for payment transactions**

**from eCommerce retailers**

**Document versions**

|  |  |  |  |
| --- | --- | --- | --- |
| Date | Author | Version | Description |
| 10.01.2017. |  | 1.0 | Initial version of the description of the WEB service for receiving payment order data from an external partner, a third party. |
| 01.09.2017. |  | 2.0 | Documents harmonized with the implemented methods. |
| 17.09.2017. |  | 3.0 | The method that returns the status of the realization of the order has been extended so that it also returns the data on the payment order.  Due to possible changes in the data on the external payment order in the Payment Transactions application. |
| 20.11.2020. |  | 4.0 | Version adapted to import payment orders from eCommerce merchants and web marketplaces. |
| 24.12.2021. |  | 4.1 | The structure of the MT110 message expanded with data from the authorization message for card payment or IPS payment. The fields are filled in if the existence of a successful authorization message is a prerequisite for creating MT110. Such a case is the payment of a group of payment orders in the PaySpot web or mobile application. |
| 25.03.2022. |  | 4.2 | Changed request and response examples for msg type messages. |
| 14.10.2022. |  | 5.0 | Added IPS. |
| 13.12.2022. |  | 5.1 | Introduced ReturnURL for mbanking IPS payments. Added merchantOrderID into merchant IPS callback hashing. |
| 6.1.2023. |  | 6.0 | Update in accordance with changes due to AML control |
| 11.7.2023. |  | 7.0 | Added PaySpot emoney APIs |
| 24.11.2023 |  | 8.0 | Added filtering payment orders |
| 01.12.2023 |  | 9.0 | Added 5.2,5.3,5.4,5.5,5.6 sections |
| 15.03.2024 |  | 10.0 | Mit transactions, 5.8 section |
| 15.01.2025 |  | 11.0 | Added UPC Integration |
| 25.02.2025 |  | 12.0 | Updated emoney payment, aded voucher list endpoint, added Emoney payment callback url and added AccountValidate (MsgType 102) |
| 26.03.2025 |  | 13.0 | MIT transactions, 6.8 section |
| 04.04.2025 |  | 14.0 | PayByLink/PayPerLink (PPL) OTP bank added, 5.9 section |

**The content**

[**1** **Introduction** 8](#_Toc194654304)

[**1.1** **Web service structure** 9](#_Toc194654305)

[**1.1.1** **The structure of the request created by the external system** 9](#_Toc194654306)

[**1.1.2** **The structure of the response created by the PaySpot system** 10](#_Toc194654307)

[**1.1.3** **Security framework** 10](#_Toc194654308)

[**1.1.4** **Communication prerequisites** 11](#_Toc194654309)

[**2** **Functionalities and description of the web wervice** 12](#_Toc194654310)

[**2.1** **Web service call rule** 12](#_Toc194654311)

[**2.2** **Web service header section of a request body** 12](#_Toc194654312)

[**2.2.1** **Header request body** 12](#_Toc194654313)

[**2.2.2** **Generate a Hash for client authentication** 12](#_Toc194654314)

[**2.2.3** **Header response body** 13](#_Toc194654315)

[**2.2.4** **Status response body** 13](#_Toc194654316)

[**3** **Orders** 13](#_Toc194654317)

[**3.1** **The course of receipt and processing of orders** 13](#_Toc194654318)

[**3.1.1** **Real-time order realization** 14](#_Toc194654319)

[**3.2** **Batch order realization** 15](#_Toc194654320)

[**3.2.1** **Realization of a payment order in the Payment System** 15](#_Toc194654321)

[**3.2.2** **Revocation of a payment order in a commercial bank** 15](#_Toc194654322)

[**3.2.3** **Confirmation for early realization of the payment order** 16](#_Toc194654323)

[**3.2.4** **Request for revocation of payment order** 16](#_Toc194654324)

[**3.3** **Input order** 16](#_Toc194654325)

[**3.3.1** **MsgType=100 - PaymentOrderValidate type request** 16](#_Toc194654326)

[**3.3.1.1** **PaymentOrderValidate input parameters** 16](#_Toc194654327)

[**3.3.1.2** **PaymentOrderValidate output parameters** 19](#_Toc194654328)

[**3.3.2** **MsgType=101 - PaymentOrderInsert type request** 20](#_Toc194654329)

[**3.3.2.1** **PaymentOrderInsert input parameters** 20](#_Toc194654330)

[**3.3.2.2** **PaymentOrderInsert output parameters** 23](#_Toc194654331)

[**3.3.3** **Order status MsgType=104 - PaymentOrderStatus** 25](#_Toc194654332)

[**3.3.3.1** **PaymentOrderStatus input parameters** 25](#_Toc194654333)

[**3.3.3.2** **PaymentOrderStatus output parameters** 26](#_Toc194654334)

[**3.3.4** **Cancel payment order MsgType=105 – PaymentOrderVoid** 28](#_Toc194654335)

[**3.3.4.1** **PaymentOrderVoid input parameters** 28](#_Toc194654336)

[**3.3.4.2** **PaymentOrderVoid output parameters** 29](#_Toc194654337)

[**3.3.5** **Filter payment order MsgType=106 – PaymentOrderFilter** 30](#_Toc194654338)

[**3.3.5.1** **PaymentOrderFilter input parameters** 30](#_Toc194654339)

[**3.3.5.2** **PaymentOrderFilter output parameters** 31](#_Toc194654340)

[**3.3.6** **Payment order confirmation MsgType=110 – PaymentOrderConfirm** 33](#_Toc194654341)

[**3.3.6.1** **PaymentOrderConfirm input parameters** 34](#_Toc194654342)

[**3.3.6.2** **PaymentOrderConfirm output parameters** 35](#_Toc194654343)

[**3.3.7 Validation of Account MsgType = 102 - AccountValidate** 36](#_Toc194654344)

[**3.3.7.1 Account validate input parameters** 36](#_Toc194654345)

[**3.3.7.2 Account validate output parameters** 37](#_Toc194654346)

[**3.4** **Order cancellation and refund to the payer** 38](#_Toc194654347)

[**3.4.1** **Order cancellation before settlement without refund** 38](#_Toc194654348)

[**3.4.2** **Partial refund of the purchase amount before settlement** 39](#_Toc194654349)

[**3.4.3** **Full refund of the purchase amount before settlement** 40](#_Toc194654350)

[**3.5** **Codebooks** 42](#_Toc194654351)

[**3.5.1** **StatusTrans** 42](#_Toc194654352)

[**4** **IPS** 43](#_Toc194654353)

[**4.1** **IPS messages** 43](#_Toc194654354)

[**4.1.1** **GetBanks MsgType=40** 43](#_Toc194654355)

[**4.1.1.1** **GetBanks input parameters** 43](#_Toc194654356)

[**4.1.1.2** **GetBanks output parameters** 43](#_Toc194654357)

[**4.1.2** **GenerateIPSData MsgType=41** 45](#_Toc194654358)

[**4.1.2.1** **GenerateIPSData input parameters** 45](#_Toc194654359)

[**4.1.2.2** **GenerateIPSData output parameters** 46](#_Toc194654360)

[**4.1.3** **PaymentReturn MsgType=43** 47](#_Toc194654361)

[**4.1.3.1** **PaymentReturn input parameters** 47](#_Toc194654362)

[**4.1.3.2** **PaymentReturn output parameters** 48](#_Toc194654363)

[**4.1.4** **CallbackURL** 49](#_Toc194654364)

[**4.1.3.1** **CallbackURL** **Hash** 49](#_Toc194654365)

[**4.2** **Integration with PaySpot web IPS page from eCommerce web site** 50](#_Toc194654366)

[**4.2.1** **Form submit** 50](#_Toc194654367)

[**4.2.1.1** **Form submit input parameters** 50](#_Toc194654368)

[**4.2.2.2** **Hash calculation** 51](#_Toc194654369)

[**4.3** **Rules and recommendations for eCommerce** 51](#_Toc194654370)

[**4.3.1** **Mandatory elements for eCommerce application/web page** 51](#_Toc194654371)

[**4.3.2** **Mandatory confirmation email elements** 52](#_Toc194654372)

[**5. Integration with PaySpot Payment methods web page from eCommerce web site (SIA)** 53](#_Toc194654373)

[**5.1** **Form submit** 53](#_Toc194654374)

[**5.1.1** **Form submit input parameters** 53](#_Toc194654375)

[**5.1.2** **Hash calculation** 54](#_Toc194654376)

[**5.2** **Callback URL Data** 55](#_Toc194654377)

[**5.2.1** **Hash Calculation for Payment by card** 56](#_Toc194654378)

[**5.2.2** **Hash Calculation for Payment by IPS** 56](#_Toc194654379)

[**5.3** **API Completion Request** 56](#_Toc194654380)

[**5.3.1** **Hash calculation** 57](#_Toc194654381)

[**5.3.2** **API Completion Response** 58](#_Toc194654382)

[**5.4** **Response Result Field** 59](#_Toc194654383)

[**5.4.1** **Card Payment** 59](#_Toc194654384)

[**5.4.2 IPS Payment** 60](#_Toc194654385)

[**5.5 Get Order Status** 60](#_Toc194654386)

[**5.5.1** **Hash calculation** 61](#_Toc194654387)

[**5.5.2** **Get Order Status response** 61](#_Toc194654388)

[**5.5.3.1 Transaction Status codes** 62](#_Toc194654389)

[**5.5.3.2 Transaction Result codes** 62](#_Toc194654390)

[**5.6 Refund / Void Transaction** 63](#_Toc194654391)

[**5.6.1** **Hash calculation** 64](#_Toc194654392)

[**5.6.2** **Void / Refund Transaction Response** 64](#_Toc194654393)

[**5.7 Client Delete Saved Card Data** 65](#_Toc194654394)

[**5.7.1** **Hash calculation** 66](#_Toc194654395)

[**5.7.2** **Client Delete Saved Card Data Response** 67](#_Toc194654396)

[**5.7.3** **Client Delete Saved Card Callback URL** 68](#_Toc194654397)

[**5.7.3.1** **Hash Calculation for Client Delete saved Card Callback** 69](#_Toc194654398)

[**5.8 Merchant Initiate Transactions (MIT)** 69](#_Toc194654399)

[**5.8.1 Registration card Request** 69](#_Toc194654400)

[**5.8.1.1 Registration Card Request** 70](#_Toc194654401)

[**5.8.1.2 RegistrationCard Response** 70](#_Toc194654402)

[**5.8.1.3 Hash calculation** 71](#_Toc194654403)

[**5.8.1.4 Form Example** 72](#_Toc194654404)

[**5.8.2 Authorize Request** 73](#_Toc194654405)

[**5.8.2.1 Authorize Request** 73](#_Toc194654406)

[**5.8.2.2 Authorize response** 74](#_Toc194654407)

[**5.8.2.3 Transaction Status codes** 75](#_Toc194654408)

[**5.8.2.4 Transaction Result codes** 75](#_Toc194654409)

[**5.8.2.5 Result codes** 76](#_Toc194654410)

[**5.8.2.6 Hash calculation** 77](#_Toc194654411)

[**5.9 Pay By Link / Pay Per Link (PPL)** 78](#_Toc194654412)

[**5.9.1 Generate Link** 78](#_Toc194654413)

[**5.9.1.1 Generate Link Request** 78](#_Toc194654414)

[**5.9.1.2 Generate Link Response** 79](#_Toc194654415)

[**5.9.1.3 Hash calculation** 80](#_Toc194654416)

[**5.9.1.4 Result codes** 80](#_Toc194654417)

[**5.9.2 List All Links** 81](#_Toc194654418)

[**5.9.2.1 List Links Request** 81](#_Toc194654419)

[**5.9.2.2 List Links Response** 82](#_Toc194654420)

[**5.9.2.3 Hash calculation** 83](#_Toc194654421)

[**5.9.2.4 Result codes** 83](#_Toc194654422)

[**5.9.3 Revoke Link** 84](#_Toc194654423)

[**5.9.3.1 Revoke Link Request** 84](#_Toc194654424)

[**5.9.3.2 Revoke Link Response** 85](#_Toc194654425)

[**5.9.3.3 Hash calculation** 85](#_Toc194654426)

[**5.9.2.4 Result codes** 86](#_Toc194654427)

[**6. Integration with PaySpot Payment methods web page from eCommerce web site (UPC)** 87](#_Toc194654428)

[**6.1** **Form submit** 87](#_Toc194654429)

[**6.1.1** **Form submit input parameters** 87](#_Toc194654430)

[**6.1.2 Hash calculation** 88](#_Toc194654431)

[**6.2** **Callback URL Data** 89](#_Toc194654432)

[**6.2.1** **Hash Calculation for Payment by card** 89](#_Toc194654433)

[**6.2.2** **Hash Calculation for Payment by IPS** 90](#_Toc194654434)

[**6.3** **API Completion Request** 90](#_Toc194654435)

[**6.3.1** **Hash calculation** 91](#_Toc194654436)

[**6.3.2** **API Completion Response** 92](#_Toc194654437)

[**6.4** **Response Result Field** 93](#_Toc194654438)

[**6.4.1 Card Payment** 93](#_Toc194654439)

[**6.4.2 IPS Payment** 95](#_Toc194654440)

[**6.5 Get Order Status** 95](#_Toc194654441)

[**6.5.1** **Hash calculation** 96](#_Toc194654442)

[**6.5.2** **Get Order Status response** 97](#_Toc194654443)

[**6.5.3.1 Transaction Status codes** 97](#_Toc194654444)

[**6.5.3.2 Transaction Result codes** 98](#_Toc194654445)

[**6.6 Refund / Void Transaction** 100](#_Toc194654446)

[**6.6.1** **Hash calculation** 101](#_Toc194654447)

[**6.6.2** **Void / Refund Transaction Response** 102](#_Toc194654448)

[**6.7 Client Delete Saved Card Data** 102](#_Toc194654449)

[**6.7.1** **Hash calculation** 103](#_Toc194654450)

[**6.7.2** **Client Delete Saved Card Data Response** 104](#_Toc194654451)

[**6.7.3** **Client Delete Saved Card Callback URL** 105](#_Toc194654452)

[**6.7.3.1 Hash Calculation for Client Delete saved Card Callback** 106](#_Toc194654453)

[**6.8 Merchant Initiate Transactions (MIT)** 106](#_Toc194654454)

[**6.8.1 Registration card Request** 106](#_Toc194654455)

[**6.8.1.1 Registration Card Request** 107](#_Toc194654456)

[**6.8.1.2 RegistrationCard Response** 107](#_Toc194654457)

[**6.8.1.3 Hash calculation** 108](#_Toc194654458)

[**6.8.1.4 Form Example** 109](#_Toc194654459)

[**6.8.2 Authorize Request** 109](#_Toc194654460)

[**6.8.2.1 Authorize Request** 110](#_Toc194654461)

[**6.8.2.2 Authorize response** 110](#_Toc194654462)

[**6.8.2.3 Transaction Status codes** 111](#_Toc194654463)

[**6.8.2.4 Transaction Result codes** 112](#_Toc194654464)

[**6.8.2.5 Result codes** 112](#_Toc194654465)

[**6.8.2.6 Hash calculation** 113](#_Toc194654466)

[**7** **Card payments** 115](#_Toc194654467)

[**7.1** **Card payments messages** 115](#_Toc194654468)

[**7.1.1** **Marketplace card transaction (msgType 62)** 115](#_Toc194654469)

[**7.1.1.1** **MarketplaceTransactions input parameters** 115](#_Toc194654470)

[**7.1.1.2** **MarketplaceTransactions output parameters** 117](#_Toc194654471)

[**8** **PaySpot e-money payments** 118](#_Toc194654472)

[**8.1** **PaySpot e-money messages** 118](#_Toc194654473)

[**8.1.1** **E-money payment authorization (msgType 72)** 118](#_Toc194654474)

[**8.1.1.1** **Emoney payment authorization input parameters** 118](#_Toc194654475)

[**8.1.1.2** **Emoney payment authorization output parameters** 119](#_Toc194654476)

[**8.1.1.3** **Response code** 120](#_Toc194654477)

[**8.1.2** **E-money details (msgType 74)** 121](#_Toc194654478)

[**8.1.2.1** **Emoney details input parameters** 121](#_Toc194654479)

[**8.1.2.2** **Emoney details output parameters** 121](#_Toc194654480)

[**8.1.2.3** **Card status codes** 122](#_Toc194654481)

[**8.1.3** **Emoney voucher list (msgType 73)** 123](#_Toc194654482)

[**8.1.3.1 Emoney voucher list input parameters** 123](#_Toc194654483)

[**8.1.3.2** **Emoney voucher list output parameters** 124](#_Toc194654484)

[**8.2** **Emoney Purchase Callback URL Data** 125](#_Toc194654485)

[**8.2.1** **Hash Calculation** 125](#_Toc194654486)

[**8.3 Emoney Payment Callback URL Data** 126](#_Toc194654487)

[**8.3.1** **Hash Calculation for Emoney Payment** 127](#_Toc194654488)

[**9.** **Appendix** 128](#_Toc194654489)

[**9.1 Workflows** 128](#_Toc194654490)

[**9.1.1 Payment order request** 128](#_Toc194654491)

# **Introduction**

This document describes the process of data exchange necessary for creating payment orders and their implementation in the NBS Payment System (PP), based on transactions created in the third party IT system - external partner and forwarded / imported into the IT system of PaySpot payment institution. The exchange of data required for the creation of payment orders is done between the IT system of the external partner and the PaySpot IT system by calling the web service that PaySpot exposed to the external partner for that purpose.

After receiving the data necessary to create a payment order, PaySpot controls them, as well as the necessary additional processing needed to create a payment order and prepare it for sending through the IT system of the partner bank, in accordance with established procedures for payment orders. The created order and its sending for realization in PP-NBS is done on the basis of instructions received from the external partner, including the value date for the realization of the order.

PaySpot and the external partner exchange unique references that identify the transaction / order in each of the systems. Further access to the account by the external partner is initiated by quoting the transaction reference or order in the external partner's system and the order reference obtained from the PaySpot system.

The exchange of data between the IT system of the partner and PaySpot is performed in accordance with the specifications of the content of web service messages described in this document. Data exchange is being done:

* by calling the appropriate PaySpot Web service method
* by sending a message of the appropriate message type (MsgType) in the web service method call

**Note:**

This is the initial document that describes the proposed data exchange process between the IT systems of external partners and PaySpot, as well as the structure of data exchange methods. PaySpot forwards sales orders to the commercial bank, under its business policy, and follows the technical specification for data exchange with the commercial bank.

## **Web service structure**

Data exchange with the PaySpot system is done by calling the universal PaySpot **REST web service.**

The service's base address is represented in this document by the variable {resource\_service\_base}, and its value will be provided later.

Rest web service uses the JSON message structure to send the data necessary to create payment orders in the PaySpot system. The exposed web service contains only one method, which is called with the appropriate message/request type. The global structure of the web service method is:

### **The structure of the request created by the external system**

**REQUEST:**

**{**

**"data": {**

**"header": {**

**"companyID": xxxxxxxxxxx,**

**"externalRequestID": ”nnnnnnnn”,**

**"requestDateTime":"date and time of request",**

**"msgType":  Message Type ID,**

**"hash":  “Authentication Hash string”,**

**"rnd": “Random string”,**

**"language": 1,**

**},**

**"body": {**

**Structure defined by type of a message**

**}**

**}**

**}**

The structure of the request header is always the same.

### **The structure of the response created by the PaySpot system**

**RESPONSE:**

**{**

**"data": {**

**"header": {**

**"companyID": xxxxxxxxxxx,**

**"externalRequestID": ”nnnnnnnn”,**

**"** r**esponseDateTime ":"date and time of request",**

**"msgType":  Message Type ID,**

**"language": 1,**

**},**

**"body": {**

**Structure defined by type of a message**

**},**

**"status": {**

**"errorCode": 0,**

**"errorMessage": "OK",**

**"dateTime": "request date time"**

**}**

**}**

**}**

### **Security framework**

PaySpot REST service is available only to external companies with the authorization provided by PaySpot DOO. Security elements are:

* All communication will use SSL / TLS v1.3 encrypted HTTP channels
* CORS (Cross-Origin Resource Sharing) Policy used to restrict domains that can access the PaySpot REST service
* API Rate Limiting will prevent any DoS-Type attacks, known users will be on the white list, in case they violate the set restrictions, we will evaluate them as a form of precaution.
* External partners will have to send the hashed data according to the specification, and our service will check the hash to make sure it has not changed on the way to our servers.
* Recording and tracking are used as a form of auditing and storing transaction details in case of need for incident reports.
* All API requests are checked to prevent SQL Injection and sending large amounts of data that could shut down the system.

### **Communication prerequisites**

Communication with PaySpot REST service will be available via:

1. Private VPN connections via the Internet with encrypted data
2. Using the Internet with encrypted protocol SSL / TLS v1.3.

# **Functionalities and description of the web wervice**

## **Web service call rule**

If the external system does not receive a response from the PaySpot web service (time out, communication interruption, etc.) or is in the received response **ErrorCode=(-1)** (error in PaySpot web service due to which the order could not be processed), in which case the external system should repeat sending the same request by calling the PaySpot service with the same request ID and with the same order reference from its system. The external system should set up the sending service so that the number of repeated sending attempts for one order does not exceed 3 attempts.

If the PaySpot service determines that an account with the same reference obtained from the external partner's system already exists in the PaySpot database, the PaySpot service will return the current status of the order and the PaySpot reference of that order in response.

## **Web service header section of a request body**

### **Header request body**

PaySpotweb service is a REST web service with a JSON message structure. Message type determines the structure of data in the body, Body {}, JSON messages. The structure of the JSON message header is shown in the table:

|  |  |  |  |
| --- | --- | --- | --- |
| **Parameter** | **Type** | **Mandatory** | **Description** |
| **Header {** |  |  | **A tag that includes all the data from the JSON request header in the web service call** |
| companyID | Int | M | Unambiguous identification of the external system that creates the PaySpot web service call |
| externalRequestID | String | O | Unique reference/call ID of the web service generated in the external partner system. Must be greater than the reference, previous call ID (in ascending order) |
| requestDateTime | String | M | Date and time of the web service call in the caller's system, format: YYYY-MM-DD hh:mm:ss (ISO) |
| msgType | Int | M | The type of message/request that is also forwarded to the PaySpot web service call |
| hash | String | M | A hash string used to authenticate a customer who called the PaySpot web service |
| rnd | String | M | Random string of 20 characters |
| language | Int | O | Language ID (1=default (sr), 2=eng ... ) |

### **Generate a Hash for client authentication**

This section explains how to generate a hash for successful authentication in the PaySpot system.

**Hash – SHA 512 Algorithm**

The hash is created by doing the SHA512 algorithm from the given string, then doing base64 encoding. Parameters in the hash string must be separated by “|”.

**plaintext = companyID + | + msgType + | + rnd + | + secretKey**

\*Parameters within the hash string must not contain backslash ("\") and pipeline ("|") characters.

\*secretKey will be delivered by the PaySpot system.

**Example:**

plaintext= 12356|101|BrHklmcUE67kV0oK2oQT|Test123456!

hash = Base64(SHA512(plaintext))

### **Header response body**

PaySpot web service creates a response in JSON format. The structure and content of the response are determined by the type of message from the web service call. The structure of the header in the PaySpot service response is shown in the table:

|  |  |  |  |
| --- | --- | --- | --- |
| **Parameter** | **Type** | **Mandatory** | **Description** |
| **Header {** |  |  | **A tag that includes all the data from the JSON request header in the web service call** |
| companyID | Int | M | Unambiguous identification of the external system that created the web service call and to which the response is returned |
| externalRequestID | String | O | Unique web service call reference / ID generated in the system of the external partner for which the web service response is returned |
| responseDateTime | String | M | Date and time of web service response creation in PaySpot system, format YYYY-MM-DD hh: mm: ss (ISO) |
| msgType | Int | M | The type of message/request for which the response was created |
| language | Int | O | Language ID (1=default (sr), 2=eng ... ) |

### **Status response body**

The PaySpot web service also contains a Status block in the created response, which contains information on the status of receiving the request from the external system. The structure of this tag is shown in the table:

|  |  |  |  |
| --- | --- | --- | --- |
| **Parameter** | **Type** | **Mandatory** | **Description** |
| **Status {** |  |  | **A tag that contains information about the processing status of the received request** |
| errorCode | Int | M | In case of an error that occurred in the process of receiving and processing the received request  Status error codes::  (-1) – error in the format of the message  (-2) – wrong MsgType  (-3) – wrong companyID  (-4) – non-authorized  (0) – the message is successfully received |
| errorMessage | String | M | Description corresponding to the obtained ResponseCode |
| requestDateTime | String | M | Date and time of creating the answer in the PaySpot web service, format: YYYY-MM-DD hh:mm:ss (ISO) |

# **Orders**

## **The course of receipt and processing of orders**

After creating an order in its system, the external partner initiates the sending of payment orders to the PaySpot system by calling the web service with the appropriate message type. From the external system, it is possible to send a group of orders that are covered by one payment transaction. One group can contain one or more payment instructions / orders. Orders from one group are sent in one web service call, as a group/package of payment orders.

For orders that have been successfully imported into the PaySpot application, a unique reference is generated in the PaySpot system and the data is returned in the response of the invited web service. A combination of data is used to later identify each order: ***order reference in the external system*** or ***order reference in the PaySpot system.***

It is possible to establish two models of realization of payment orders created in the IT system of an external partner:

* Real-time realization of orders – the client pays for the created orders in the external partner system. After receiving and controlling the payment order, PaySpot sends the order for realization to the commercial bank and returns the parameters on the success of the execution of the payment order to the external system. The external system in the process of daily settlement calculates liabilities to PaySpot and transfers funds for realized orders and related commissions to the PaySpot account.
* Batch realization of the order – the group of orders were accepted by PaySpot, but was not forwarded for final realization to the Payment System. One of the preconditions needs to be met in order to be sent to the final realization:
  1. The Client needs to make a payment at the PaySpot counter. Then the appropriate action of confirming the payment of the group of accounts through the user interface of the PaySpot application is initiated, in the PaySpot branch
  2. The Second precondition is that External System, after a certain time, initiates the realization of the order by calling the appropriate method of web service
  3. At last, the order is due for execution, an event defined by the value date determined in the external system

**Note 1:**

PaySpot will determine a unique identifier for each external partner in its system. This unique identifier will be used by the external partner to "present" itself to the PaySpot web service when calling methods from that web service.

**Note 2:**

The external system sends:

* Unique number/reference of the order group and unique number/reference of each order, as an integral part of the body data of the request for creating or changing payment orders

PaySpot in the intern system generates

* unique number of the group of payment orders and
* the unique number of each payment order

Created orders in the Paypot system will have a status determined according to the model of sending orders from an external system. If the orders are sent in the Batch mode of order execution, the created group of orders and the created orders receive the so-called "Pending" status, i.e. remain in the status of "created" orders until some of the events described above confirm the realization of the created payment order, after which the payment order is forwarded for final realization in a commercial bank, i.e. in Clearing, RTGS or IPS system NBS.

### **Real-time order realization**

For real-time execution of orders, the external system initiates the so-called two-stage sending of orders in the PaySpot IT system - sending of orders through 2 cycles:

* Account control and commission calculation - each order of a new payment group, after being created in the system of an external partner, is sent for control to PaySpot. After successful control, PaySpot calculates the total commission and related costs, and returns the response information for each account to the external system. If an error is detected during the logical control of the order, an error code and a message describing the error are returned in response to the method call. The external system is obliged to make the necessary and possible logical controls of the correctness of the order in its system, before sending the order, as well as to correct incorrect data before re-sending the order to the PaySpot system.
* Sending orders for execution - after all, orders belonging to one group of payment orders have been created and after the external partner has collected the total amount from the client, the external system initiates sending a package/group of confirmed orders for execution to PaySpot. The PaySpot system performs another control of the correctness of the data in the orders, calculates the corresponding commission and costs, and initiates the sending of a group of orders to the commercial bank for the final realization in the Payment Operations. In response to the called method, it returns to the external system information on the success of the Order Receipt. For orders that have an error, the error information is returned, and a note that the order was not executed. After receiving the confirmation of the execution of all orders from the group of orders, the external partner completes the process of recording paid orders in its system.

The real-time method of realization of payment orders will be used most often when payment orders are entered through a self-service payment terminal or kiosk. In this case, the external system will be financially responsible for all realized orders.

## **Batch order realization**

This way of realizing a group of payment orders will be applied in the processes in which the realization of payment orders created in the PaySpot system, based on imported data from the external system, is done after an event that initiates the execution of orders. Examples.

* payment of imported orders by the client at the counter of the PaySpot branch,
* subsequently received confirmation from the external partner that the orders have been collected from the client
* instruction of the external partner to forward the orders for execution
* the expiration of the pending order period, for orders entered with a future value date

The execution of orders are initiated by the defined integration process and the procedure for the execution of payment orders with an external system.

The batch model for the realization of payment orders will be used in the realization of payment operations services for eCommerce systems. For the needs of these solutions, the payment system will be adapted to the need to transfer funds to eCommerce merchants or sub-merchants. The transfer of funds will be executed under the instructions prepared by the eCommerce system.

### **Realization of a payment order in the Payment System**

Realization of created payment orders in the PaySpot system is performed from the dedicated PaySpot account specified in the payment order sent for execution to the commercial bank. The commercial bank will inform PaySpot about the status of order realization through regular daily statements, as well as through the so-called inquiries about the status of the realization of the order forwarded through the call of the appropriate web service method. PaySpot will provide information on the status of the realization of the payment system for the needs of the external system. The information on the status of the payment order will be forwarded to the external system in response to the call of the web service method.

### **Revocation of a payment order in a commercial bank**

The commercial bank, as well as PaySpot, can perform appropriate actions on the payment order before or during the processing process, by the rules of national payment system and payment rules in the commercial bank through which payment orders created in the PaySpot system are executed. PaySpot has the option to initiate the revocation/withdrawal of a payment order before the order enters the execution process in the national payment system of the bank. At the request of PaySpot, the bank may withdraw the order or cancel it before the final realization, in the case of external orders, or cancel it during the working day in the case of internal orders, i.e. orders executed within the bank, and following the business rules of the bank.

If the order is executed in the bank's system or the national payment system, and the client requests that the order be revoked or that a certain correction of data is performed on it, such requests are resolved by the PaySpot Complaints Resolution Procedure. Through the complaints process in the PaySpot system, only complaints related to the realized order in domestic payment transactions are resolved. This complaint process does not resolve issues with the delivery of goods, services, or other obligations paid by this order.

### **Confirmation for early realization of the payment order**

If the external system imported orders into the PaySpot application that have a value date in the future, it is possible that the external system sends a request to change the value date of the order or to realize a payment order with a new currency date, which can not be longer than the initial date order currency. The reason is that the client was given a confirmation of payment which states the initially entered date for payment of the order.

### **Request for revocation of payment order**

The external system may initiate the revocation/withdrawal of a given payment order until the order is forwarded to the commercial bank for execution. The revocation of a previously imported and confirmed payment order is initiated by the external system under the complaint procedure of the PaySpot Payment Institution, as well as the contract determined by the procedure for revoking the payment order. In addition to the request for revocation of the payment order, the external system must provide the reason for the revocation of the order. PaySpot has the right to request original confirmations of payments made, in the process of resolving requests for revocation of given payment orders.

## **Input order**

Entering a new order is done by calling the web service with the message type ***MsgType = 101***, which performs logical control/verification of orders from the package/group of orders and creation of orders for the transfer of funds to the merchant, if the orders are valid.

In one call of the web service, data for one group of accounts and all related orders created in the external system are forwarded. The external system generates a unique reference of the order group and a unique reference for each order, which are unique in the external partner system. The sum of the amounts of orders from the group must be the sum of the amounts of individual orders. An account group obtained from an external partner system can contain one or more accounts.

PaySpot creates a payment group in its system when it receives a new order group reference from the external partner system, after which it creates a unique order reference in the PaySpot system for each new order, according to the unique order reference from the external system.

### **MsgType=100 - PaymentOrderValidate type request**

**[POST]: /api/paymentordervalidate**

**REQUEST URL PRODUCTION:** [**https://www.nsgway.rs:50010/api/paymentordervalidate**](https://www.nsgway.rs:50010/api/paymentordervalidate)

**REQUEST URL TEST:** [**https://test.nsgway.rs:50009/api/paymentordervalidate**](https://test.nsgway.rs:50009/api/paymentordervalidate)

Message type ***MsgType=100 - PaymentOrderValidate*** is used for orders input validation (parameters input validation related to allowed characters, if mandatory fields are entered, etc.) **This message is optional.**

### **PaymentOrderValidate input parameters**

**PaymentOrderValidate** message input parameters

|  |  |  |  |
| --- | --- | --- | --- |
| **Parameter** | **Type** |  | **Description** |
| **Body {** |  |  | **Message body** |
| **Orders [ {** |  |  | **Orders** |
| sequenceNo | Int | M | The ordinal number of orders within the group |
| merchantOrderReference | String | M | Unique order reference generated by sender AN (30) |
| debtorAccount | String | O | Client's account number. |
| debtorName | String | M | Payer name. A (35) |
| debtorAddress | String | M | Payer address. A (35) (min. 4 characters) |
| debtorCity | String | M | Payer city. A (20) |
| debtorModul | String | O | Client model N (2) |
| debtorReference | String | O | Ordering number AN (23) |
| beneficiaryAccount | String | M | Payee account number AN (30) |
| beneficiaryCode | String | O | Payee registration number, natural person - AN (13) or legal entity - N (8) |
| beneficiaryName | String | M | Payee name A (35) |
| beneficiaryAddress | String | M | Payee address A(35) |
| beneficiaryCity | String | M | Payee city A(20) |
| beneficiaryModul | Int | O | Payee model N(2) |
| beneficiaryReference | String | O | Payee reference number AN(23) |
| amountTrans | Decimal | M | GROSS Payment order amount, includes decimal point and max. 2 decimal places. Example: 1346.23 or 3458.20. The amount is in the currency defined for the order group. D(15,2) |
| senderFeeAmount | Decimal | M | The amount of the fee for the sender of the order, external partner, or 0.00 if this fee does not exist. D(15,2) |
| paySpotFeeAmount | Decimal | O | The amount of the fee due to PaySpot for the performed order processing service, if this fee is calculated by the external partner.D(15,2) |
| beneficiaryAmount | Decimal | M | Amount to be paid to the payee. D(15,2) |
| beneficiaryCurrency | Int | M | Currency code in which the amount of the obligation to the payee is stated. AN (3), ISO Numeric code |
| purposeCode | Int | M | Payment code from the NBS codebook. AN(3) |
| paymentPurpose | String | M | Purpose of payment, description. A(80) |
| isUrgent | Int | M | Indication of whether the order is urgent (1 = YES, 0 = NOT URGENT order). For IPS account is always = 1 (urgent) |
| valueDate | String | M | Value date. Format: YYYY-MM-DD.  It cannot be older than the date of work (it cannot be a date from the past).  It could be a date in the future, for pending orders. |
| beneficiaryEmail | String | O | Recipient's email address A(35) |
| beneficiaryContactNumber | String | O | Payee's phone A(50) |
| beneficiaryContactPerson | String | O | Contact person of the payee A(100) |
| **} ] Orders** |  |  | **End of orders** |
| **}Body** |  |  | **End of message body** |

**PaymentOrderValidate** message request example

|  |
| --- |
| {  "data": {  "header": {  "companyID": 123456,  "requestDateTime": "2022-09-06 10:09:07",  "msgType": 100,  "rnd": "cz8W0DSlje17mtinrUYu",  "hash": "FDgTZInsnahVZyjbIId4RoR4+MPwJySlJ+dCuB6ZCVmhG0vD6I7jrtvWVgC5fTRZhssZ7FAgFSZ6YVqkBKp+Jg==",  "language": 2  },  "body": {  "orders": [  {  "sequenceNo": 1,  "merchantOrderReference": "1316",  "debtorAccount": null,  "debtorName": "Petar Petrovic",  "debtorAddress": "Ulica 18",  "debtorCity": "Valjevo",  "debtorModul": null,  "debtorReference": null,  "beneficiaryAccount": "160000000021612549",  "beneficiaryCode": null,  "beneficiaryName": "Vujica Vujic",  "beneficiaryAddress": "Prote Mateje",  "beneficiaryCity": "Valjevo",  "beneficiaryModul": null,  "beneficiaryReference": null,  "amountTrans": 50,  "senderFeeAmount": 23,  "paySpotFeeAmount": 0,  "beneficiaryAmount": 27,  "beneficiaryCurrency": 941,  "purposeCode": 289,  "paymentPurpose": "Baksis",  "isUrgent": 0,  "valueDate": "2022-10-04",  "beneficiaryEmail": null,  "beneficiaryContactNumber": null,  "beneficiaryContactPerson": null  }  ]  }  }  } |

### **PaymentOrderValidate output parameters**

**PaymentOrderValidate** message output parameters

|  |  |  |  |
| --- | --- | --- | --- |
| **Parameter** | **Type** |  | **Description** |
| **Body {** |  |  | **Message body** |
| **Orders[{** |  |  | **Orders** |
| merchantOrderReference | String | M | Unique payment order reference in the eCommerce or merchant system. AN(20) |
| errorCode | Int | M | Error code determined in the process of processing instructions for creating individual payment orders for eCommerce merchants |
| errorMsg | String | M | Error message text |
| **}]Orders** |  |  |  |
| **}Body** |  |  |  |

**PaymentOrderValidate** message response example

|  |
| --- |
| {  "data": {  "header": {  "companyID": 123456,  "responseDateTime": "2022-09-06T15:27:55.203194",  "msgType": 100,  "language": 2  },  "body": {  "orders": [  {  "merchantOrderReference": "1316",  "errorCode": 0,  "errorMessage": "OK"  }  ]  },  "status": {  "errorCode": 0,  "errorMessage": "Success",  "dateTime": "2022-09-06T15:27:55.203194"  }  }  } |

### **MsgType=101 - PaymentOrderInsert type request**

**[POST]: /api/paymentorderinsert**

**REQUEST URL PRODUCTION:** [**https://www.nsgway.rs:50010/api/paymentorderinsert**](https://www.nsgway.rs:50010/api/paymentorderinsert)

**REQUEST URL TEST:** [**https://test.nsgway.rs:50009/api/paymentorderinsert**](https://test.nsgway.rs:50009/api/paymentorderinsert)

Message type ***MsgType=101 - PaymentOrderInsert*** is used for order creation in the PaySpot system. This is mandatory initial message.

### **PaymentOrderInsert input parameters**

**PaymentOrderInsert** message input parameters

|  |  |  |  |
| --- | --- | --- | --- |
| **Parameter** | **Type** |  | **Description** |
| **Body {** |  |  | **Message body** |
| **PaymentOrderGroup {** |  |  | **Payment order group** |
| merchantOrderID | String | M | The unique reference of the purchase order for which payment orders are sent and is generated in the eCommerce or merchant system. AN (40). |
| merchantOrderAmount | Decimal | M | Total value of the purchase order/purchase for which the payment has been initiated. D (15.2)  Contains the total value of goods and/or services + the amount of additional fee that the external partner system may charge for a transaction. This is actually amount which will be paid during transaction. |
| merchantCurrencyCode | Int | M | Currency code for purchase order/order in + paid products and services. ISO numeric code. The default is the national currency (941 for RSD) |
| paymentType | Int | M | Payment method selected by the customer:   * ***1=Payment card*** * ***2= PaySpot branch*** * ***3=Transfer order*** * ***4=IPS payment*** * ***5=PaySpot e-money*** * ***6=Custom*** * ***7=Payment from Deposit*** |
| actionType | String | M | Action type:   * ***I=insert a new group of orders,*** * ***U=update the existing group of orders (allowed only after 105 message)*** |
| requestType | String | O | Order request type (if not set then common payment order request):   * ***xxxx=PaySpot voucher purchase,***   ***xxxx=Parking servis voucher purchase*** |
| sumOfOrders | Decimal | M | Sum of payment orders' gross amounts. D (15.2) |
| numberOfOrders | Int | M | Total number of orders in the group, forwarded in the request message. |
| terminalID | String | M (only for card payments) | POS terminal ID where the payment transaction was made with a payment card. AN (8) |
| transtype | String | M (only for card payments) | Auth/PreAuth/Info/Campaign/Invoice |
| **Orders [ {** |  |  | **A tag that includes a set of data for each individual order** |
| sequenceNo | Int | M | The ordinal number of orders within the group |
| merchantOrderReference | String | M | Unique order reference generated by sender AN (30) |
| debtorAccount | String | O | Client's account number. |
| debtorName | String | M | Payer name. A (35) |
| debtorAddress | String | M | Payer address. A (35) (min. 4 characters) |
| debtorCity | String | M | Payer city. A (20) |
| debtorModul | String | O | Client model N (2) |
| debtorReference | String | O | Ordering number AN (23) |
| beneficiaryAccount | String | M | Payee account number AN (30) |
| beneficiaryCode | String | O | Payee registration number, natural person - AN (13) or legal entity - N (8) |
| beneficiaryName | String | M | Payee name A (35) |
| beneficiaryAddress | String | M | Payee address A(35) |
| beneficiaryCity | String | M | Payee city A(20) |
| beneficiaryModul | Int | O | Payee model N(2) |
| beneficiaryReference | String | O | Payee reference number AN(23) |
| amountTrans | Decimal | M | GROSS Payment order amount, includes decimal point and max. 2 decimal places. Example: 1346.23 or 3458.20. The amount is in the currency defined for the order group. D(15,2) |
| senderFeeAmount | Decimal | M | The amount of the fee for the sender of the order, external partner, or 0.00 if this fee does not exist. D(15,2) |
| paySpotFeeAmount | Decimal | O | The amount of the fee due to PaySpot for the performed order processing service, if this fee is calculated by the external partner.D(15,2) |
| beneficiaryAmount | Decimal | M | Amount to be paid to the payee. D(15,2) |
| beneficiaryCurrency | Int | M | Currency code in which the amount of the obligation to the payee is stated. AN (3), ISO Numeric code |
| purposeCode | Int | M | Payment code from the NBS codebook. AN(3)  **Special purpose codes are used for purchasing of PaySpot emoney:**  **699 - Voucher** |
| paymentPurpose | String | M | Purpose of payment, description. A(80) |
| isUrgent | Int | M | Indication of whether the order is urgent (1 = YES, 0 = NOT URGENT order). For IPS account is always = 1 (urgent)  =2 - Each payment order will be created individually for payment to the merchant |
| valueDate | String | M | Value date. Format: YYYY-MM-DD.  It cannot be older than the date of work (it cannot be a date from the past).  It could be a date in the future, for pending orders. |
| beneficiaryEmail | String | O | Recipient's email address A(35) |
| beneficiaryContactNumber | String | O | Payee's phone A(50) |
| beneficiaryContactPerson | String | O | Contact person of the payee A(100) |
| **} ] Orders** |  |  | **End of orders** |
| **} PaymentOrderGroup** |  |  | **End of payment order group** |

|  |  |  |  |
| --- | --- | --- | --- |
| **Customer {** |  |  |  |
| customerEmail | String | M (when PaySpot emoney is purchased) | Email address where an email with voucher data will be sent. A(70) |
| **} Customer** |  |  | **End of data on the customer who pays the purchase order on the basis of which the group of payment orders for end-merchants was created** |

|  |  |  |  |
| --- | --- | --- | --- |
| **}Body** |  |  | **End of message body** |

**PaymentOrderInsert** message request example

|  |
| --- |
| {  "data": {  "header": {  "companyID": 123456,  "requestDateTime": "2022-09-06 10:09:07",  "msgType": 101,  "rnd": "cz8W0DSlje17mtinrUYu",  "hash": "FDgTZInsnahVZyjbIId4RoR4+MPwJySlJ+dCuB6ZCVmhG0vD6I7jrtvWVgC5fTRZhssZ7FAgFSZ6YVqkBKp+Jg==",  "language": 2  },  "body": {  "paymentOrderGroup": {  "merchantOrderID": "Order123",  "merchantOrderAmount": 50,  "merchantCurrencyCode": 941,  "paymentType": 1,  "actionType": "I",  "sumOfOrders": 50,  "numberOfOrders": 1,  "terminalID": "IN001807",  "transtype": "Auth",  "orders": [  {  "sequenceNo": 1,  "merchantOrderReference": "1316",  "debtorAccount": null,  "debtorName": "Petar Petrovic",  "debtorAddress": "Ulica 18",  "debtorCity": "Valjevo",  "debtorModul": null,  "debtorReference": null,  "beneficiaryAccount": "160000000021612549",  "beneficiaryCode": null,  "beneficiaryName": "Vujica Vujic",  "beneficiaryAddress": "Prote Mateje",  "beneficiaryCity": "Valjevo",  "beneficiaryModul": null,  "beneficiaryReference": null,  "amountTrans": 50,  "senderFeeAmount": 23,  "paySpotFeeAmount": 0,  "beneficiaryAmount": 27,  "beneficiaryCurrency": 941,  "purposeCode": 289,  "paymentPurpose": "Baksis",  "isUrgent": 0,  "valueDate": "2022-10-04",  "beneficiaryEmail": null,  "beneficiaryContactNumber": null,  "beneficiaryContactPerson": null  }  ]  }  }  }  } |

### **PaymentOrderInsert output parameters**

**PaymentOrderInsert** message output parameters

|  |  |  |  |
| --- | --- | --- | --- |
| **Parameter** | **Type** |  | **Description** |
| **Body {** |  |  |  |
| **PaymentOrderGroup {** |  |  |  |
| merchantOrderID | String | M | The unique purchase order reference for which payment orders are sent and is generated in the eCommerce or merchant system. AN(40). |
| payspotGroupID | String | M | The unique purchase order (payment orders group) created in the PaySpot system. AN (10) |
| groupStatus | String | O | Payment orders group status in the PaySpot system: 0 = created group, -1 = group not created due to incorrect data |
| groupStatusDescr | String | O | Payment orders group status description in the PaySpot system |
| **Orders[{** |  |  |  |
| sequenceNo | Int | M | Ordinal number of orders within the group |
| merchantOrderReference | String | M | Unique payment order reference in the eCommerce or merchant system. AN(20) |
| payspotTransactionID | String | M | Unique payment order reference generated by PaySpot. AN (10) |
| statusTrans | String | M | Payment order status in PaySpot system (1 = order created, -1 = order not created ) |
| statusTransDescr | String | M | Description of the payment order status in the PaySpot system |
| createDate | String | M | Date of payment order creation in the PaySpot system. Format YYYY-MM-DD, (2021-09-16) |
| createTime | String | M | Time of payment order creation in the PaySpot system., Format: hh:mm:ss (09:43:24) |
| paymentDate | String | O | Date of order realization, ie date of sending the order to the bank: YYYY-MM-DD, (2021-10-14) |
| paymentTime | String | O | Time of order realization, ie time of sending the order to the bank, Format: hh:mm:ss (09:43:24) |
| feeAmount | Decimal | M | Payment order commission. It is determined by the Payment institution, a decimal number with 2 decimal places |
| additionalFee | Decimal | O | The amount of additional cost for the execution of a payment order, e.g. payment card transaction cost. 0.00 if there is no additional cost |
| senderFee | Decimal | M | Commission due to the sender. It is determined by the Payment institution, decimal number with 2 decimal places 0.00 if it does not count and forward the partner (senderFeeAmount from request) |
| statusProcessing | String | M | Order processing status (status > 0 = Success, status < 0 = Fail) |
| statusProcessingMsg | String | M | Order processing status message |
| errorCode | Int | M | Error code determined in the process of processing instructions for creating individual payment orders for eCommerce merchants |
| errorMsg | String | M | Error message text |
| **}]Orders** |  |  |  |
| **} PaymentOrderGroup** |  |  |  |
| errorCode | Int | M | Error code determined in the process of processing instructions for creating individual payment orders for eCommerce merchants |
| errorMsg | String | M | Error message text |
| **}Body** |  |  |  |

**PaymentOrderInsert** message response example

|  |
| --- |
| {  "data": {  "header": {  "companyID": 123456,  "responseDateTime": "2022-09-06T15:27:55.203194",  "msgType": 101,  "language": 2  },  "body": {  "paymentOrderGroup": {  "merchantOrderID": "Order123",  "payspotGroupID": 1100,  "groupStatus": 0,  "groupStatusDescr": "Group Created",  "orders": [  {  "sequenceNo": 1,  "merchantOrderReference": "1316",  "payspotTransactionID": "1120",  "statusTrans": "1",  "statusTransDescr": "Correct instructions",  "createDate": "2022-09-06",  "createTime": "15:27:55",  "paymentDate": null,  "paymentTime": null,  "feeAmount": 0.00,  "additionalFee": 0.00,  "senderFee": 23.00,  "statusProcessing": "1",  "statusProcessingMsg": "Success",  "errorCode": 0,  "errorMessage": "OK"  }  ]  },  "errorCode": null,  "errorMessage": null  },  "status": {  "errorCode": 0,  "errorMessage": "Success",  "dateTime": "2022-09-06T15:27:55.203194"  }  }  } |

### **Order status MsgType=104 - PaymentOrderStatus**

**[POST]: /api/paymentorderstatus**

**REQUEST PRODUCTION URL:** [**https://www.nsgway.rs:50010/api/paymentorderstatus**](https://www.nsgway.rs:50010/api/paymentorderstatus)

**REQUEST TEST URL:** [**https://test.nsgway.rs:50009/api/paymentorderstatus**](https://test.nsgway.rs:50009/api/paymentorderstatus)

Message type ***MsgType=104 - PaymentOrderStatus*** is used for a query by which the external partner system receives information on the status of the payment order created in the PaySpot system.

#### **PaymentOrderStatus input parameters**

**PaymentOrderStatus** message input parameters

|  |  |  |  |
| --- | --- | --- | --- |
| **Parametar** | **Type** |  | **Description** |
| **Body {** |  |  |  |
| merchantOrderID | String | M | The unique reference of the purchase order for which payment orders are sent which is generated in the eCommerce or merchant system. AN(40). |
| merchantReference | String | M | Unique order reference generated in the eCommerce or merchant system. (MerchantOrderReference from Message 101) |
| payspotGroupID | String | M | The unique purchase order (group of payment orders) created in the PaySpot system. AN (10) |
| payspotTransactionID | String | M | Payment order reference generated in the PaySpot system N(10) |
| **}Body** |  |  |  |

**PaymentOrderStatus** message request example

|  |
| --- |
| {  "data": {  "header": {  "companyID": 123456,  "requestDateTime": "2022-09-06 12:25:15",  "msgType": 104,  "rnd": "AmbKvRLQBNqaniG1dkc2",  "hash": "KUYgWuNtRgkHbv7LXOWL6TcVWe/YTGwkUlTTdYo8GhX3lnUa8MInjzHOceiEqNnsU/9WT7FtOojogSD3UfPusw==",  "language": 2  },  "body": {  "merchantOrderID": "Order123",  "merchantReference": "1316",  "payspotGroupID": "1100",  "payspotTransactionID": "1120"  }  }  } |

#### **PaymentOrderStatus output parameters**

**PaymentOrderStatus** message output parameters

|  |  |  |
| --- | --- | --- |
| **Parameter** | **Type** | **Description** |
| **Body {** |  |  |
| merchantOrderID | String | The unique reference of the purchase order for which payment orders are sent which is generated in the eCommerce or merchant system. AN(40). |
| merchantReference | String | Unique order reference generated in the eCommerce or merchant system. |
| amountTrans | Decimal | GROSS Payment order amount, includes decimal point and max. 2 decimal places. Example: 1346.23 or 3458.20. The amount is in the currency defined for the order group. D(15,2) |
| beneficiaryAmount | Decimal | Amount to be paid to the payee. D(15,2) |
| payspotGroupID | String | The unique purchase order (group of payment orders) created in the PaySpot system. AN (10) |
| payspotTransactionID | String | Unique order reference generated by PaySpot. |
| statusTrans | String | Order status in PaySpot system (See chapter 3.5.1). |
| statusTransDescr | String | Order status description in the PaySpot system |
| createDate | String | Date of order creation in the PaySpot system. Format YYYY-MM-DD, (2021-09-16) |
| createTime | String | Time of order creation in the PaySpot system., Format: hh:mm:ss (09:43:24) |
| paymentDate | String | Date of order realization, ie the date of sending the order for realization to the bank: YYYY-MM-DD, (2023-12-12) |
| feeAmount | Decimal | Payment order commission. It is determined by the Payment institution, a decimal number with 2 decimal places |
| additionalFee | Decimal | The amount of additional cost for the execution of a payment order, e.g. payment card transaction cost. 0.00 if there is no additional cost |
| senderFee | Decimal | Commission due to the sender. It is determined by the Payment institution, decimal number with 2 decimal places 0.00 if it does not count and forward the Payment Institution |
| errorCode | Int | Error code determined in the process of processing instructions for creating individual payment orders for eCommerce merchants |
| errorMsg | String | Error message text |
| **}Body** |  |  |

**PaymentOrderStatus**message response example

|  |
| --- |
| {  "data": {  "header": {  "companyID": 123456,  "responseDateTime": "2022-09-06T16:06:34.524438",  "msgType": 104,  "language": 2  },  "body": {  "merchantOrderID": "Order123",  "merchantReference": "1316",  "payspotGroupID": "1100",  "payspotTransactionID": "1120",  "beneficiaryAmount": 77.00,  "amountTrans": 100.00,  "statusTrans": "-9",  "statusTransDescr": "Revoked (Void) / canceled instruction",  "createDate": "2022-09-06",  "createTime": "15:25:48",  "paymentDate": 2022-09-06,  "paymentTime": null,  "feeAmount": 0.00,  "additionalFee": 0,  "senderFee": 23,  "errorCode": 0,  "errorMessage": "OK"  },  "status": {  "errorCode": 0,  "errorMessage": "OK",  "dateTime": "2022-09-06T16:06:34.524438"  }  }  } |

### **Cancel payment order MsgType=105 – PaymentOrderVoid**

**[POST]: /api/paymentordervoid**

**REQUEST PRODUCTION URL:** [**https://www.nsgway.rs:50010/api/paymentordervoid**](https://www.nsgway.rs:50010/api/paymentordervoid)

**REQUEST TEST URL:** [**https://test.nsgway.rs:50009/api/paymentordervoid**](https://test.nsgway.rs:50009/api/paymentordervoid)

Message type ***MsgType=105 - PaymentOrderVoid*** is used for order canceling **only if funds are not transfered yet** to the merchant account (the funds are still available on the PaySpot account). Otherwise the message will do nothing.

The main purpose of this message is order cancelling if something is wrong with order data (for instance bad debtor or beneficiary data, wrong beneficiary reference number, etc.)

#### **PaymentOrderVoid input parameters**

**PaymentOrderVoid** message input parameters

|  |  |  |  |
| --- | --- | --- | --- |
| **Parameter** | **Type** |  | **Description** |
| **Body {** |  |  |  |
| merchantOrderID | String | M | The unique reference of the purchase order for which payment orders are sent which is generated in the eCommerce or merchant system. AN(40). |
| merchantReference | String | M | Unique order reference generated in the eCommerce or merchant system. (MerchantOrderReference from Message 101) |
| payspotGroupID | String | M | The unique purchase order (group of payment orders) created in the PaySpot system. AN (10) |
| payspotTransactionID | String | M | Order reference generated by PaySpot. |
| amountTrans | Decimal | M | Gross amount for void/refund |
| senderFeeAmount | Decimal | M | Void/refund fee |
| beneficiaryAmount | Decimal | M | Net amount for void/refund to payer |
| requestReason | String | M | Order deleting reason:   * ***'V'=Order cancel*** * ***'R'=Total refund to the customer*** * ***'PR'= Partial refund to the customer*** |
| requestReasonMsg | String | M | Order deleting reason description |
| **}Body** |  |  |  |

**PaymentOrderVoid** message request example

|  |
| --- |
| {  "data": {  "header": {  "companyID": 123456,  "requestDateTime": "2022-09-06 12:25:15",  "msgType": 105,  "rnd": "AmbKvRLQBNqaniG1dkc2",  "hash":  "+TbGfkTBzoANBi3804ZriyWoHKoGLk8ZfCpsdlvpRrUKPQjgr3SIqZQsY4L1kJiE3SK5S2J/3133iIS6/XZw==",  "language": 2  },  "body": {  "merchantOrderID": "Order123",  "merchantReference": "1316",  "payspotGroupID": "1100",  "payspotTransactionID": "1120",  "amountTrans": 100,  "senderFeeAmount": 26.20,  "beneficiaryAmount": 73.80,  "requestReasonCode": "V",  "requestReasonMsg": "Refund reason"  }  }  } |

#### 

#### **PaymentOrderVoid output parameters**

**PaymentOrderVoid** message output parameters

|  |  |  |  |
| --- | --- | --- | --- |
| **Parameter** | **Type** |  | **Description** |
| **Body {** |  |  |  |
| merchantOrderID | String | M | The unique reference of the purchase order for which payment orders are sent which is generated in the eCommerce or merchant system. AN(40). |
| merchantReference | String | M | Unique order reference generated in the eCommerce or merchant system. (MerchantOrderReference from Message 101) |
| payspotGroupID | String | M | The unique purchase order (group of payment orders) created in the PaySpot system. AN (10) |
| payspotTransactionID | String | M | Unique order reference generated by PaySpot. |
| statusTrans | String | M | Order status in the PaySpot system. (See chapter 3.5.1) |
| statusTransDescr | String | M | Order status description in the PaySpot system. |
| executionDate | String | O | Order execution / deletion date, Format: YYYY-MM-DD, (2021-10-14) |
| executionTime | String | O | Order execution / deletion time, Format: hh:mm:ss (09:43:24) |
| statusProcessing | String | M | Order processing status (status > 0 = Success, status < 0 = Fail) |
| statusProcessingMsg | String | M | Order processing status message |
| errorCode | Int | M | Error code determined in the process of processing instructions for creating individual payment orders for eCommerce merchants |
| errorMsg | String | M | Error message text |
| **}Body** |  |  |  |

**PaymentOrderVoid** messag response example

|  |
| --- |
| {  "data": {  "header": {  "companyID": 123456,  "responseDateTime": "2022-09-06T15:53:01.92447",  "msgType": 105,  "language": 2  },  "body": {  "merchantOrderID": "Order123",  "merchantReference": "1316",  "payspotGroupID": "1100",  "payspotTransactionID": "1120",  "statusTrans": "-9",  "statusTransDescr": "Revoked (Void) / canceled instruction",  "executionDate": "2022-09-06",  "executionTime": "15:53:01",  "statusProcessing": "1",  "statusProcessingMsg": "Success",  "errorCode": 0,  "errorMessage": "OK"  },  "status": {  "errorCode": 0,  "errorMessage": "OK",  "dateTime": "2022-09-06T15:53:01.92447"  }  }  } |

### **Filter payment order MsgType=106 – PaymentOrderFilter**

**[POST]: /api/** **paymentorderfilter**

**REQUEST PRODUCTION URL:** [**https://www.nsgway.rs:50010/api/paymentorder**](https://www.nsgway.rs:50010/api/paymentorder)**filter**

**REQUEST TEST URL:** [**https://test.nsgway.rs:50009/api/paymentorder**](https://test.nsgway.rs:50009/api/paymentorder)**filter**

Message type ***MsgType=106 - PaymentOrderFilter*** is used to preview payment orders payed with payment cards, based on given parameters.

#### **PaymentOrderFilter input parameters**

**PaymentOrderVoid** message input parameters

|  |  |  |  |
| --- | --- | --- | --- |
| **Parameter** | **Type** |  | **Description** |
| **Body {** |  |  |  |
| cid | String | M | Identity number of the company (MB) |
| dateFrom | String | M | The start date of the reporting period |
| dateTo | String | M | The end date of the reporting period |
| **}Body** |  |  |  |

**PaymentOrderFilter** message request example

|  |
| --- |
| {    "data": {      "header": {        "companyID": 123456,        "requestDateTime": "2023-11-24 13:38:43",        "msgType": 106,        "rnd": "B1kYlrl32lq",        "hash": "XEibAUiNxYnCjZXh6BYp3RHzh0x881n7BcPhN+AqDprtLeBmtnHDkpH/HAHaCbbEHw44zeMvMnpZUDoyS7TrvQ==",        "language": 2      },      "body": {         "mb": "4536778907654",         "dateFrom": "2023-11-24 13:38:43",         "dateTo": "2023-12-04 13:38:43"      }    }  } |

#### **PaymentOrderFilter output parameters**

**PaymentOrderFilter** message output parameters

|  |  |  |  |
| --- | --- | --- | --- |
| **Parameter** | **Type** |  | **Description** |
| **Body {** |  |  |  |
| **posTransInsurance [{** |  |  |  |
| orderNumber | String |  | Order number |
| costBearer | String |  | Cost bearer |
| orderType | String |  | Type of order |
| posOwnerAccountName | String |  | Name of POS account owner |
| posOwnerAccountPayment | String |  | Payment of POS account owner |
| posOwnerAccountDescription | String |  | Name of POS account owner |
| representativeName | String |  | Name of representative |
| orgUnit | String |  | Organization unit |
| orgUnitName | String |  | Name of organization unit |
| posTerminalId | String |  | Id of POS terminal |
| bankSpecificDate | String |  | Bank specification date |
| cardNumber | String |  | Card number |
| installmentNumber | String |  | Installment number |
| currency | String |  | Currency |
| posProcessingCommission | String |  | Commission of POS processing |
| orderAmount | String |  | Order amount |
| commissionPp | String |  | Commission of payment process |
| grossOrderAmount | String |  | Gross order amount |
| posOrderCommission | String |  | Commission of POS order |
| debtorName | String |  | Debtor name |
| debtorAddress | String |  | Debtor address |
| debtorCity | String |  | Debtor city |
| beneficiaryName | String |  | Beneficiary name |
| beneficiaryAddress | String |  | Beneficiary address |
| beneficiaryCity | String |  | Beneficiary city |
| purposeCode | String |  | Purpose code |
| purpose | String |  | Purpose of payment |
| accountPpInFavor | String |  | For the purpose of which account it is paid |
| callNumberPolicy | String |  | Call number policy |
| }] |  |  |  |
| **}Body** |  |  |  |

**PaymentOrderFilter** messag response example

|  |
| --- |
| {    "data": {      "header": {      "companyID": 123456,      "externalRequestID": **null**,      "responseDateTime": "2023-11-23T16:20:37.189055",      "msgType": 106,      "language": 2      },      "body": {        "posTransInsurance": [          {            "orderNumber": "1",            "costBearer": "John Doe",            "orderType": "Payment",            "posOwnerAccountName": "Company XYZ",            "posOwnerAccountPayment": "123456789",            "posOwnerAccountDescription": "Main Account",            "representativeName": "Agent ABC",            "orgUnit": "F001",            "orgUnitName": "OrgUnit 1",            "posTerminalId": "POS123",            "bankSpecificDate": "2023-11-15",            "cardNumber": "987654321",            "installmentNumber": "1",            "currency": "RSD",            "posProcessingCommission": "Processed",            "orderAmount": "1000.00",            "commissionPp": "10.00",            "grossOrderAmount": "1010.00",            "posOrderCommission": "5.00",            "debtorName": "Debtor ABC",            "debtorAddress": "123 Main St",            "debtorCity": "Cityville",            "beneficiaryName": "Beneficiary XYZ",            "beneficiaryAddress": "456 Oak St",            "beneficiaryCity": "Townsville",            "purposeCode": "PL001",            "purpose": "purpose",            "accountPpInFavor": "987654321",            "callNumberPolicy": "123456"          }        ]      },      "status": {        "errorCode": 0,        "errorMessage": "OK",        "dateTime": "2023-11-23T16:20:37.189055"      }    }  } |

### **Payment order confirmation MsgType=110 – PaymentOrderConfirm**

**[POST]: /api/paymentorderconfirm**

**REQUEST PRODUCTION URL:** [**https://www.nsgway.rs:50010/api/paymentorderconfirm**](https://www.nsgway.rs:50010/api/paymentorderconfirm)

**REQUEST TEST URL:** [**https://test.nsgway.rs:50009/api/paymentorderconfirm**](https://test.nsgway.rs:50009/api/paymentorderconfirm)

There are certain conditions that must be met in order for the Payment Institution to execute the transfer order to the final merchant, in accordance with the instructions received from the eCommerce platform, or from an external system managed by eCommerce or another type of eCommerce platform.

The basic condition under which the Payment Institution will make the transfer of funds to the merchant who sell the goods or services, for which the buyer made payment by one of the offered methods, is that the payment process by the buyer is completed, ie. that the funds paid by the buyer for the created purchase order have been transferred to a dedicated PaySpot account for the execution of payment orders to the seller of goods or services. Verification of this condition will be performed in the process of processing daily statements obtained from used banks, ie specifications of card transactions obtained from the acquirer bank.

For purchase orders for which the Payment Institution has received a confirmation of payment of funds to a dedicated account, the Payment Institution will place the corresponding transfer orders in the status of waiting for confirmation for final realization, ie refund to the buyer. If one of the conditions is met:

that the buyer has confirmed the delivery of the goods or services or

that the deadline for the buyer's complaint has expired, ie that the transfer order is due for payment - that the value date specified by the eCommerce organizer in the instructions for creating a transfer order to the seller is due.

In the request message MsgType = 110, the sender of instructions for creating and executing a payment order to the final merchant, sub-merchant, can, in addition to confirming the payment order, send instructions for correcting payment instructions on the initially created payment order:

* ***Payee's account number*** – it is possible to change the account number of the payee if there are technical reasons for changing the account, e.g. the initial account in a commercial bank has been closed for some reason, so the payment should be made to an account in another bank
* ***Order amount* –** if in the meantime partial refund has occurred it is possible to send 110 msg with lowered amount (remaining amount after refund)
* ***Value date*** – date of execution of the payment order. It may change if the buyer has confirmed receipt of the purchased goods or services before the payment deadline or the sender of the instruction has learned that the delivery has been made.

Confirming the created payment order changes the status of the created order in the PaySpot system from „***Created***“ to „***Confirmed***“ pay warrant.

PaySpot will initiate the process of executing payment orders to the final recipient of funds, i.e. to the merchant who sold the goods or services.

The request for the realization of the order for the transfer of funds to the seller will be initiated by the system organizer of the eCommerce platform, by calling the PaySpot service with the message:

* ***MsgType = 110*** – confirmation of the payment order to the merchant

### **PaymentOrderConfirm input parameters**

Input parameters of the ***PaymentOrderConfirm*** message type are the elements described in the table below. Within one request it is possible to send a confirmation for the execution of several payment orders, for different purchase orders, and for different merchants.

**PaymentOrderConfirm** message input parameters

|  |  |  |  |
| --- | --- | --- | --- |
| **Parameter** | **Type** |  | **Description** |
| **Body {** |  |  |  |
| **OrderConfirm [{** |  |  |  |
| merchantOrderID | String | M | The unique reference of the purchase order for which payment orders are sent which is generated in the eCommerce or merchant system. AN(40). |
| merchantReference | String | M | Unique order reference generated in the eCommerce or merchant system. (MerchantOrderReference from Message 101) |
| payspotGroupID | String | M | The unique purchase order (group of payment orders) created in the PaySpot system. AN (10) |
| payspotTransactionID | String | M | Unique order reference generated by PaySpot. |
| beneficiaryAccount | String | M | Payee's account number, AN(30) |
| beneficiaryAmount | Decimal(15,2) | M | Net amount of payment order, includes decimal point and max. 2 decimal places |
| valueDate | String | M | Value date (execution date) of the payment order. Format: YYYY-MM-DD. |
| }] |  |  |  |
| **}Body** |  |  |  |

**PaymentOrderConfirm** message request example

|  |
| --- |
| {  "data": {  "header": {  "companyID": 123456,  "requestDateTime": "2022-09-06 08:20:51",  "msgType": 110,  "rnd": "BrHklmcUE67kV0oK2oQT",  "hash":  "G+pxakdMbJrtqn8oKugQz62fc8vomaJJ1eVVC7Kxk5LG3AJQSBSFcLdm3lVNVteDQ5J1rofXehVfethkQQAyiQ==",  "language": 2  },  "body": {  "orderConfirm": [  {  "merchantOrderID": "Order123",  "merchantReference": "1316",  "payspotGroupID": "1100",  "payspotTransactionID": "1120",  "beneficiaryAccount": "160000000021612549",  "beneficiaryAmount": 73.80,  "valueDate": "2022-10-19"  }  ]  }  }  } |

### **PaymentOrderConfirm output parameters**

Output parameters for ***PaymentOrderConfirm*** the message type will contain information about the change in the status of the creation or modified payment order for making payments to the merchant who is the recipient of funds. Confirming the created payment order changes the status of the created order in the PaySpot system from „***Created***“ to „***Confirmed***“ pay warrant.

**PaymentOrderConfirm** message output parameters

|  |  |  |  |
| --- | --- | --- | --- |
| **Parameter** | **Type** |  | **Description** |
| **Body {** |  |  |  |
| **OrderConfirm [{** |  |  |  |
| merchantOrderID | String | M | The unique reference of the purchase order for which payment orders are sent which is generated in the eCommerce or merchant system. AN(40). |
| merchantReference | String | O | Unique order reference generated in the eCommerce or merchant system. (MerchantOrderReference from Message 101) |
| payspotGroupID | String | O | The unique purchase order (group of payment orders) created in the PaySpot system. AN (10) |
| payspotTransactionID | String | M | Unique order reference generated by PaySpot. |
| statusTrans | String | O | Order status in the PaySpot system |
| statusTransDescr | String | O | Order status description in the PaySpot system |
| statusProcessing | String | M | Order processing status (status > 0 = Success, status < 0 = Fail, for possible values see chapter 3.5.1) |
| statusProcessingMsg | String | M | Order processing status message |
| }] |  |  |  |
| errorCode | Int | O | Error code determined in the process of processing instructions for creating individual payment orders for eCommerce merchants |
| errorMsg | String | O | Error message text |
| **}Body** |  |  |  |

**PaymentOrderConfirm**message response example

|  |
| --- |
| {  "data": {  "header": {  "companyID": 123456,  "responseDateTime": "2022-09-06T15:49:42.810638",  "msgType": 110,  "language": 2  },  "body": {  "orderConfirm": [  {  "merchantOrderID": "Order123",  "merchantReference": "1316",  "payspotGroupID": "1100",  "payspotTransactionID": "1120",  "statusTrans": null,  "statusTransDescr": null,  "statusProcessing": "-2",  "statusProcessingMsg": "There is no confirmation from the bank about the paid transaction by payment card."  }  ],  "errorCode": 0,  "errorMessage": "OK"  },  "status": {  "errorCode": 0,  "errorMessage": "OK",  "dateTime": "2022-09-06T15:49:42.810638"  }  }  } |

### **3.3.7 Validation of Account MsgType = 102 - AccountValidate**

**[POST]: /api/AccountValidate**

**REQUEST PRODUCTION URL:** [**https://www.nsgway.rs:50010/api/AccountValidate**](https://www.nsgway.rs:50010/api/AccountValidate%20)

**REQUEST TEST URL:** [**https://test.nsgway.rs:50009/api/**](https://test.nsgway.rs:50009/api/paymentorderconfirm)**AccountValidate**

Call of this endpoint checks the entered account number.

First, it formats the entered account number into a standard format: 3 digits for the bank prefix + 13 digits for the account party number + 2 digits for the control number. After that, it performs validation of the formatted account by checking:

* Whether an account number has been entered
* The length of the formatted account number
* Modular validation

### **3.3.7.1 Account validate input parameters**

**Account validate** message input parameters

|  |  |  |  |
| --- | --- | --- | --- |
| **Parameter** | **Type** |  | **Description** |
| beneficiaryAccount | String | M | Account number for validation |

**Account validate** message request example

|  |
| --- |
| {      "Data": {          "Header": {              "CompanyID": "12345",              "ExternalRequestID": "",              "RequestDateTime": "2025-02-03 14:45:47",              "MsgType": "102",              "Rnd": "psd6vopmBerw8A97jmJE",              "Hash": "3Ar9OHxiUUcS/Z/fLaM8u0knpPzqGNDPv8oN6FHY9W71xZeJ1jLm97uEVkWXnsWnGQDo2vBBrkg59Tagqti/NQ==",              "Language": "1"          },          "Body": {              "beneficiaryAccount": "123-0000000123456-12"          }      }  } |

### **3.3.7.2 Account validate output parameters**

**Account validate** message output parameters

|  |  |  |  |
| --- | --- | --- | --- |
| **Parameter** | **Type** |  | **Description** |
| beneficiaryAccount | String | M | Returns formatted account number from request (removes "-" from account number if it has) |
| isValid | int | M | Returns 1 if it is valid, 0 if isn’t |
| errorCode | Integer | M | Error code |
| errorMessage | String | M | Description of errorCode |

**Account validate** message response example

|  |
| --- |
| {    "data": {      "body": {        "beneficiaryAccount": "123000000012345612",        "isValid": 1,        "errorCode": 0,        "errorMessage": "OK"      },      "status": {        "errorCode": 0,        "errorMessage": "OK",        "dateTime": "2025-02-26T13:32:40.313172"      },      "header": {        "companyID": 12345,        "externalRequestID": **null**,        "responseDateTime": "2025-02-26T13:32:40.313172",        "msgType": 102,        "language": 1      }    }  } |

## **Order cancellation and refund to the payer**

If the payer has complained to eCommerce about the delivered goods or services, eCommerce will, by its operating rules, check the delivery with the merchant and the payer. If the eCommerce platform, determines that it is necessary to return the funds to the payer, it is also required to send such instruction to PaySpot.

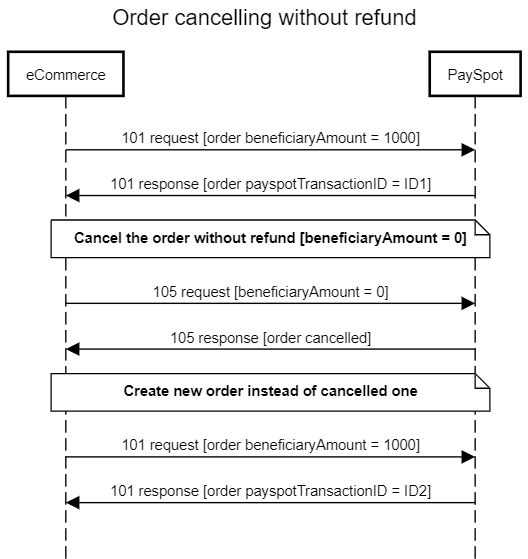
In addition eCommerce can cancel payment order and create new order(s) instead of the canceled one. This is only possible while payment order is not executed and funds are not tranferred yet to the merchant's account.

**Note: Card payment refunds will be executed through card payment system and eCommerce is responsible for initiating refund process. In the case of IPS payments PaySpot has to initiate refund to the payer account.**

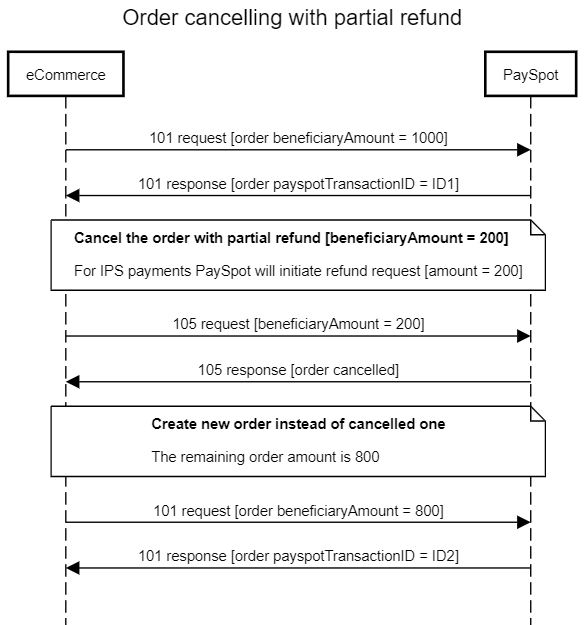
It is possible that eCommerce will perform order cancelation, optionally with full or partial refund to the payer:

* When the funds are not transferred yet to the merchant's account

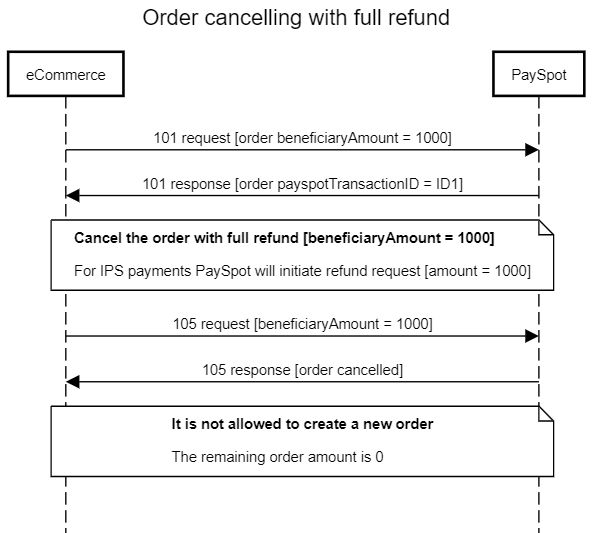
### **Order cancellation before settlement without refund**



### **Partial refund of the purchase amount before settlement**



### **Full refund of the purchase amount before settlement**



# **Codebooks**

## **StatusTrans**

|  |  |
| --- | --- |
| **Value** | **Description** |
| -1 | Payment order with error |
| -2 | There is no confirmation from the bank about paid card transaction |
| -3 | The amounts for the creation of orders for payment transactions in messages 101 and 110 differ |
| -5 | There is no confirmation from the bank about paid IPS transaction |
| -6 | Invalid value for paymentType field |
| -9 | Payment order deleted |
| 0 | Payment order loaded |
| 1 | Payment order created |
| 2 | Payment order confirmed and is waiting for execution |
| 3 | Payment order has already been marked as paid |
| 9 | Payment order processing in progress |

# **IPS**

### **IPS messages**

### **GetBanks MsgType=40**

**[POST]: /api/ips/getbanks/{EntityType}**

**REQUEST PRODUCTION URL:** [**https://www.nsgway.rs:50010/api/ips/getbanks/IE**](https://www.nsgway.rs:50010/api/ips/getbanks/IE)

**REQUEST TEST URL:** [**https://test.nsgway.rs:50009/api/ips/getbanks/IE**](https://test.nsgway.rs:50009/api/ips/getbanks/IE)

Message type ***MsgType=40- GetBanks*** is used to obtain a list of banks that have been verified for IPS within the payment system with the Central Bank. **{EntityType}** is payee type and may be legal (LE) or individual (IE). IPS output parameters depend on entity type.

#### **GetBanks input parameters**

**GetBanks** endpoint does not have Body parameters, it is required to send only Header parameters.

**GetBanks** message request example

|  |
| --- |
| {    "data": {      "header": {        "companyID": 123456,        "requestDateTime": "2022-10-14 08:38:19",        "msgType": 40,        "rnd": "5ZnUkpJKma0Mb3CQu9Ef",        "hash": "zhUWKYX2NYqvkFpPr5k6Y+ljMjrC5XIjpq9xnD2KP4NeIYIjNPCqBAwLVGOlswx+OFnBkfrkPY90Q41iym93qQ==",        "language": 2      }    }  } |

#### **GetBanks output parameters**

**GetBanks** message output parameters

|  |  |  |  |
| --- | --- | --- | --- |
| **Parameter** | **Type** |  | **Description** |
| **Body {** |  |  | **A tag that includes all data in the response body** |
| **BankList [{** |  |  | **Array of banks objects** |
| id | Int | M | Unique identification of the bank, id that will be sent in the request to generate a Deep Link. |
| name | String | M | Name of the bank |
| **}] BankList** |  |  |  |
| **} Body** |  |  |  |

**GetBanks**message response example

|  |
| --- |
| {      "data": {          "header": {              "companyID": 123456,              "responseDateTime": “2022-10-14T08:38:20.113981**”**,              "msgType": 40,              "language": 2          },          "body": {              "bankList": [  {                      "id": 160,                      "name": "BANCA INTESA "                  },                  {                      "id": 170,                      "name": "UNICREDIT BANK"                  },                  {                      "id": 105,                      "name": "AIK BANKА"                  },                  {                      "id": 200,                      "name": "BANKA POŠTANSKA ŠTEDIONICA"                  },                  {                      "id": 200,                      "name": "BANKA POŠTANSKA - NOVA APP"                  },                  {                      "id": 330,                      "name": "CREDIT AGRICOLE BANKA"                  },                  {                      "id": 150,                      "name": "DIREKTNA BANKA"                  },                  {                      "id": 250,                      "name": "EUROBANK"                  },                  {                      "id": 155,                      "name": "HALKBANK"                  },                  {                      "id": 205,                      "name": "KOMERCIJALNA BANKA"                  },                  {                      "id": 325,                      "name": "OTP BANKА"                  },                  {                      "id": 265,                      "name": "RAIFFEISEN BANKA"                  },                  {                      "id": 285,                      "name": "SBERBANK"                  }              ]          },          "status": {              "errorCode": 0,              "errorMessage": "OK",              "dateTime": "2022-10-17T12:04:36.426532"          }      }  } |

### **GenerateIPSData MsgType=41**

**[POST]: /api/ips/generateips/{EntityType}**

**REQUEST PRODUCTION URL:** [**https://www.nsgway.rs:50010/api/ips/generateips/IE**](https://www.nsgway.rs:50010/api/ips/generateips/IE)

**REQUEST TEST URL:** [**https://test.nsgway.rs:50009/api/ips/generateips/IE**](https://test.nsgway.rs:50009/api/ips/generateips/IE)

Message type ***MsgType=41 GenerateIPSData*** is used to generate IPS output data (deep link and/or QR code) for customer m-banking application where payment will be made.

***IPS output data expiration is 2 minutes in production enviroment and 3 minutes in test enviroment.***

#### **GenerateIPSData input parameters**

**GenerateIPSData**message input parameters

|  |  |  |  |
| --- | --- | --- | --- |
| **Parametar** | **Type** |  | **Description** |
| **Body {** |  |  | **A tag that includes all data in the request body** |
| merchantOrderID | String | M | The unique reference of the purchase order, generated in the eCommerce or merchant system. AN(40). |
| merchantOrderAmount | Decimal | M | Total value of the purchase order/purchase for which the payment was initiated. D (15.2)  Contains the total value of goods and/or services + the amount of additional fee that the external partner system may charge for a transaction |
| merchantCurrencyCode | Int | M | Currency code for purchase order. ISO numeric code. The default is the national currency (941 for RSD) |
| bankID | Int | M | ID from MsgType=40 |
| callbackURL | String | M | Merchant have to offer an URL endpoint where transaction status will be sent, using HTTP POST method (Described in 4.1.3 Section) |
| returnURL | String | O | Merchant could offer an URL endpoint where mbanking application will redirect the user after payment. If URL is not provided the user will be redirected to PaySpot payment confirmation page.  **NOTE: This parameter is relevant only for deep link payments. If URL is provided then the merchant is responsible for dsplaying transaction status indication (success/fail) on the provided URL, as requested by National Bank of Serbia.** |
| **}Body** |  |  |  |

**GenerateIPSData** message request example

|  |
| --- |
| {    "data": {      "header": {        "companyID": 123456,        "requestDateTime": "2022-10-17 08:38:19",        "msgType": 41,        "rnd": "5ZnUkpJKma0Mb3CQu9Ef",        "hash": "tcsgAXxCBSNAqRVa8PwQh/QwsR2YzK4tahKkZN8w67MVL5dysIkATRO80MPSXRtg8EGPxiHy6PiMHcgWMOq+Gw==",        "language": 2      },      "body": {        "merchantOrderID": "Order123",        "merchantOrderAmount": 31.32,        "merchantCurrencyCode": 941,        "bankID": 1,        "callbackURL": "https://callbackurl.com/api/statusTrans"      }    }  } |

#### **GenerateIPSData output parameters**

**GenerateIPSData** message output parameters

|  |  |  |  |
| --- | --- | --- | --- |
| **Parametar** | **Type** |  | **Description** |
| **Body {** |  |  | **A tag that includes all data in the response body** |
| deepLink | String | M | Deep Link that should open the customer's m-banking application |
| qrCode | String | M | QR Code |
| paySpotOrderID | String | M | Unique IPS reference |
| errorCode | Int | O | Error code determined in the process of processing instructions for creating individual payment orders for eCommerce merchants |
| errorMsg | String | O | Error message text |
| **}Body** |  |  |  |

**GenerateIPSData**message response example

|  |
| --- |
| {      "data": {          "header": {              "companyID": 123456,              "responseDateTime": "2022-10-17T12:39:00.113981",              "msgType": 41,              "language": 2          },          "body": {              "deepLink": "https://ipspos.bancaintesa.rs/ips/ek/fl/?data=SzpFS3xWOjAxfEM6MXxSOjE2MDYwMDAwMDEyNjY3ODg0OHxOOlBBWVNQT1QgRE9PIE5PVkkgU0FEfEk6UlNEMzEsMzJ8U0Y6MjIxfE06NjA2MnxSUDpJMDAwMDA5NTIyMjkwMDAwMDAx&callback=https%3a%2f%2ftest.nsgway.rs%3a50009%2fstatusTrans%2fI000009522290000001%2f31.32",              "qrCode": null,              "paySpotOrderID": "I000009522290000001",              "errorCode": 0,              "errorMessage": "OK"          },          "status": {              "errorCode": 0,              "errorMessage": "OK",              "dateTime": "2022-10-17T12:39:00.113981"          }      }  } |

### **PaymentReturn MsgType=43**

**[POST]: /api/ips/paymentreturn**

**REQUEST PRODUCTION URL:** [**https://www.nsgway.rs:50010/api/ips/paymentreturn**](https://www.nsgway.rs:50010/api/ips/paymentreturn)

**REQUEST TEST URL:** [**https://test.nsgway.rs:50009/api/ips/paymentreturn**](https://test.nsgway.rs:50009/api/ips/paymentreturn)

Message type ***MsgType=43 PaymentReturn*** is used for IPS payment return (refund).

#### **PaymentReturn input parameters**

**PaymentReturn**message input parameters

|  |  |  |  |
| --- | --- | --- | --- |
| **Parametar** | **Type** |  | **Description** |
| **Body {** |  |  | **A tag that includes all data in the request body** |
| payspotOrderID | String | M | Unique IPS reference (returned in msg 41). |
| returnAmount | Decimal | M | Amount to be returned (refunded). |
| terminalID | String | M | Merchant IPS terminal id ( delivered by PaySpot ) |
| **}Body** |  |  |  |

**PaymentReturn** message request example

|  |
| --- |
| {    "data": {      "header": {        "companyID": 123456,        "requestDateTime": "2022-10-17 08:38:19",        "msgType": 43,        "rnd": "5ZnUkpJKma0Mb3CQu9Ef",        "hash": "tcsgAXxCBSNAqRVa8PwQh/QwsR2YzK4tahKkZN8w67MVL5dysIkATRO80MPSXRtg8EGPxiHy6PiMHcgWMOq+Gw==",        "language": 2      },      "body": {        "payspotOrderID": "I000009522290000001",        "returnAmount": 31.32,        "terminalID": "I0000095",      }    }  } |

#### **PaymentReturn output parameters**

**PaymentReturn** message output parameters

|  |  |  |  |
| --- | --- | --- | --- |
| **Parametar** | **Type** |  | **Description** |
| **Body {** |  |  | **A tag that includes all the data in the response body** |
| statusCode | String | M | Refund transaction status, statusCode can be:  00 – successful transaction, 05 – unsuccessful transaction;  -1 – error in the IPS system (unable to execute transaction) |
| errorCode | Int | O | Error code determined in the process of processing instructions for creating individual payment orders for eCommerce merchants |
| errorMsg | String | O | Error message text |
| **}Body** |  |  |  |

**PaymentReturn**message response example

|  |
| --- |
| {      "data": {          "header": {              "companyID": 123456,              "responseDateTime": "2022-10-17T12:39:00.113981",              "msgType": 43,              "language": 2          },          "body": {              "statusCode": "00",              "errorCode": 0,              "errorMessage": "OK"          },          "status": {              "errorCode": 0,              "errorMessage": "OK",              "dateTime": "2022-10-17T12:39:00.113981"          }      }  } |

### **CallbackURL**

ECommerce needs to send an URL endpoint in **MsgType=41** where transaction status will be sent using HTTP POST method.

**CallbackURL** message input parameters

|  |  |  |  |
| --- | --- | --- | --- |
| **Parametar** | **Type** |  | **Description** |
| **{** |  |  |  |
| statusCode | String | M | Transaction status, statusCode can be:  00 – successful transaction, 05 – unsuccessful transaction, 82 – timeout;  -1 – error in the IPS system (unable to execute transaction) |
| paySpotOrderID | String | M | Unique identification of the created purchase order in the PaySpot system. |
| merchantOrderID | String | M | Unique identification of the created purchase order in the merchant system. |
| rnd | String | M | Random string of 20 characters |
| hash | String | M | A hash string used to authenticate a customer. (Described in 4.1.3.1 Section) |
| **}** |  |  |  |

**CallbackURL** messagerequest example

|  |
| --- |
| {     "statusCode": "00",     "paySpotOrderID": "I000009522290000001",  "merchantOrderID": "Order123",     "rnd": "BrHklmcUE67kV0oK2oQT",     "hash": "bMNSAO5qNHZUV8vhJNFmWeo26/n3mSBBGwPLBUGSDDqSCkT83q/LueYt5IWql1259HUn7CqsT5zluAalI30KSQ=="  } |

#### **4.1.3.1** **CallbackURL** **Hash**

This section explains how PaySpot will generate a hash for successful callback authentication.

**Hash – SHA 512 Algorithm**

The hash is created by doing the SHA512 algorithm from the given string, then doing base64 encoding. Parameters in the hash string must be separated by “|”.

**plaintext = rnd + | + paySpotOrderID + | + merchantOrderID + | + secretKeyCallback**

\*Parameters within the hash string must not contain backslash ("\") and pipeline ("|") characters.

\*secretKey will be delivered by the PaySpot system.

**Example:**

plaintext= BrHklmcUE67kV0oK2oQT|I000009522290000001|TestGenerate1|Test123456!

hash = Base64(SHA512(plaintext))

### **Integration with PaySpot web IPS page from eCommerce web site**

### **Form submit**

**[POST]: /api/PsWebIps/submit**

**REQUEST PRODUCTION URL:** [**https://www.nsgway.rs:50010/api/pswebips/submit**](https://www.nsgway.rs:50010/api/pswebips/submit)

**REQUEST TEST URL:** [**https://test.nsgway.rs:50009/api/pswebips/submit**](https://test.nsgway.rs:50009/api/pswebips/submit)

#### **Form submit input parameters**

Form submit input parameters

|  |  |  |  |
| --- | --- | --- | --- |
| **Parametar** | **Type** |  | **Description** |
| companyID | Int | M | Unambiguous identification of the external system that executes the PaySpot web service call |
| merchantOrderID | String | M | The unique reference of the purchase order, generated in the eCommerce or merchant system. AN(40). |
| merchantOrderAmount | Decimal | M | Total value of the purchase order/purchase for which the payment was initiated. D (15.2)  Contains the total value of goods and/or services + the amount of additional fee that the external partner system may charge for a transaction |
| merchantCurrencyCode | Int | M | Currency code for purchase order. ISO numeric code. The default is the national currency (941 for RSD) |
| language | Int | O | Language ID (1=default (sr), 2=eng ... ) |
| callbackURL | String | M | eCommerce has to provide an URL endpoint where transaction status will be sent, using HTTP POST method (Described in 4.1.3 Section) |
| successURL | String | M | eCommerce has to provide an URL endpoint where the customer browser will be redirected after successful payment |
| cancelURL | String | M | eCommerce has to provide an URL endpoint where the customer browser will be redirected if customer quits purchase before payment |
| errorURL | String | M | eCommerce has to provide an URL endpoint where the customer browser will be redirected in the case of unsuccessful payment |
| returnURL | String | O | Merchant could offer an URL endpoint where mbanking application will redirect the user after payment. If URL is not provided the user will be redirected to PaySpot payment confirmation page.  **NOTE: This parameter is relevant only for deep link payments. If URL is provided then the merchant is responsible for dsplaying transaction status indication (success/fail) on the provided URL, as requested by National Bank of Serbia.** |
| hash | String | M | Hash calculated as explained in the section 4.2.2.2. **It is mandatory to calculate the hash on the eCommerce server.** |
| rnd | String | M | Random string used for hash calculation |
| currentDate | String | M | Date and time of the submit request in the format: YYYY-MM-DD hh:mm:ss (ISO). **Must be sent in UTC timezone,** |

Html example for integration with PaySpot IPS web page:

      <form

        method="post"

        action='https://test.nsgway.rs:50009/api/PsWebIps/submit'

        target="output\_frame">

        <input type="hidden" name="companyId" value=123456 />

        <input type="hidden" name="merchantOrderID" value=Order123 />

        <input type="hidden" name="merchantOrderAmount" value=100.32 />

        <input type="hidden" name="merchantCurrencyCode" value=941 />

        <input type="hidden" name="language" value=1 />

        <input type="hidden" name="callbackURL" value='https://callbackUrl' />

        <input type="hidden" name="successURL" value='https://successUrl' />

        <input type="hidden" name="cancelURL" value='https://cancelUrl' />

        <input type="hidden" name="errorURL" value='https://failUrl' />

        <input type="hidden" name="hash" value= '4Oav8Pg9qfHWitVAk/BJOx/cujvdQfNV5tH07VvWPYxuT9nX3NuJAFrZCcdo7D28WeB92TzmB/R3xnFW43KkjQ==' />

        <input type="hidden" name="rnd" value='mIlyKRws6GHHi8mlJtg6' />

        <input type="hidden" name="currentDate" value=new Date().toISOString().replace('T', ' ').substring(0, 16) />

      </form>

Optional iframe (if you want to display PaySpot IPS page within iframe)

<iframe name="output\_frame" src="" id="output\_frame" width="100%" height="100%"></iframe>

#### **4.2.2.2** **Hash calculation**

This section explains how eCommerce needs to generate the hash for submit request from eCommerce web site.

**Hash – SHA 512 Algorithm**

The hash is created by doing the SHA512 algorithm from the given string, then doing base64 encoding. Parameters in the hash string must be separated by “|”.

**plaintext = rnd + | + currentDate + | + merchantOrderID + | + merchantOrderAmount + | + secretKey**

\*Parameters within the hash string must not contain backslash ("\") and pipeline ("|") characters.

\*secretKey will be delivered by the PaySpot system.

\*Difference between CurrentDate parameter (**in UTC**) and time when the request is called can't be more than **5min**

**Example:**

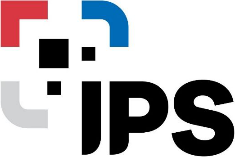
plaintext= BrHklmcUE67kV0oK2oQT|2022-10-17 08:38:19|TestGenerate1|100.32|Test123456!

hash = Base64(SHA512(plaintext))

### **Rules and recommendations for eCommerce**

### **Mandatory elements for eCommerce application/web page**

Following IPS logo should be visible in the page footer



Payments method selection should contain images below with an appropriate text explaining IPS payments. Text example: *Mbanking aplikacijom koju imate instaliranu na svom mobilnom uređaju skenirajte prikazani jednokratan IPS QR kôd i izvršite plaćanje u okruženju Vaše banke. Informacija o ishodu plaćanja biće Vam prikazana odmah po završetku obrade ali će Vam biti dostavljena i na e-mail adresu uz potvrdu o kupovini.*





**ECommerce Internet point of sale has to provide payments to all subjects, physical persons and legal entities, in accordance with Serbian law.**

### **Mandatory confirmation email elements**

ECommerce has to send a confirmation email for every successful payment transaction. Mandatory email elements are:

* Order reference number
* Total transaction amount
* Date and time of transaction
* IPS reference number

# **5. Integration with PaySpot Payment methods web page from eCommerce web site (SIA)**

### **Form submit**

**[POST]: /api/Ecommerce/submit**

**REQUEST PRODUCTION URL:** [**https://www.nsgway.rs:50010/api/ecommerce/submit**](https://www.nsgway.rs:50010/api/ecommerce/submit)

**REQUEST TEST URL:** [**https://test.nsgway.rs:50009/api/ecommerce/submit**](https://test.nsgway.rs:50009/api/ecommerce/submit)

#### **Form submit input parameters**

Form submit input parameters

|  |  |  |  |
| --- | --- | --- | --- |
| **Parametar** | **Type** |  | **Description** |
| companyID | Int | M | Unambiguous identification of the external system that executes the PaySpot web service call.  \*Will be delivered by the PaySpot system. |
| merchantOrderID | String | M | The unique reference of the purchase order, generated in the eCommerce or merchant system. AN(40). |
| merchantOrderAmount | Decimal | M | Total value of the purchase order/purchase for which the payment was initiated. D (15.2)  Contains the total value of goods and/or services + the amount of additional fee that the external partner system may charge for a transaction |
| merchantCurrencyCode | Int | M | Currency code for purchase order. ISO numeric code. The default is the national currency (941 for RSD) |
| language | Int | O | Language ID (1=default (sr), 2=eng ... ) |
| email | String | M | Customer’s e-mail address |
| customerId | String | O | ID which is an alternative to email for CIT transactions, can be used to store user's saved cards in case email is not provided. |
| successURL | String | M | eCommerce has to provide an URL endpoint where the customer browser will be redirected after successful payment |
| cancelURL | String | M | eCommerce has to provide an URL endpoint where the customer browser will be redirected if customer quits purchase before payment |
| errorURL | String | M | eCommerce has to provide an URL endpoint where the customer browser will be redirected in the case of unsuccessful payment |
| hash | String | M | Hash calculated as explained in the section 5.1.2. **It is mandatory to calculate the hash on the eCommerce server.** |
| rnd | String | M | Random string used for hash calculation |
| currentDate | String | M | Date and time of the submit request in the format: YYYY-MM-DD hh:mm:ss (ISO). **Must be sent in UTC timezone,** |
| requestType | Int | M | Request type in PaySpot System. ( Marketplace platform = 10, Web shop platform =11) |
| paymentPurpose | String | O | Payment purpose for IPS payment. AN(35) |
| timeout | Int | O | Redirect User on Cancel URL aftter timeout ( value in seconds ). Default value = 300 |

Html example for integration with PaySpot Payment Methods web page:

      <form

        method="post"

        action='https://test.nsgway.rs:50009/api/ecommerce/submit'

        target="output\_frame">

        <input type="hidden" name="companyId" value=123456 />

        <input type="hidden" name="merchantOrderID" value='Order123' />

        <input type="hidden" name="merchantOrderAmount" value=100.32 />

        <input type="hidden" name="merchantCurrencyCode" value=941 />

        <input type="hidden" name="language" value=1 />

        <input type="hidden" name="successURL" value='https://successUrl' />

        <input type="hidden" name="cancelURL" value='https://cancelUrl' />

        <input type="hidden" name="errorURL" value='https://failUrl' />

        <input type="hidden" name="email" value='test@gmail.com' />

<input type="hidden" name="customerId" value='123456789' />

        <input type="hidden" name="requestType" value=11/>

        <input type="hidden" name="hash" value= '4Oav8Pg9qfHWitVAk/BJOx/cujvdQfNV5tH07VvWPYxuT9nX3NuJAFrZCcdo7D28WeB92TzmB/R3xnFW43KkjQ==' />

        <input type="hidden" name="rnd" value='mIlyKRws6GHHi8mlJtg6' />

<input type="hidden" name="timeout" value='600' />

        <input type="hidden" name="currentDate" value=new Date().toISOString().replace('T', ' ').substring(0, 16) />

      </form>

Optional iframe (if you want to display PaySpot Payment Methods page within iframe)

<iframe name="output\_frame" src="" id="output\_frame" width="100%" height="100%"></iframe>

#### **5.1.2** **Hash calculation**

This section explains how eCommerce needs to generate the hash for submit request from eCommerce web site.

**Hash – SHA 512 Algorithm**

The hash is created by doing the SHA512 algorithm from the given string, then doing base64 encoding. Parameters in the hash string must be separated by “|”.

**plaintext = rnd + | + currentDate + | + merchantOrderID + | + merchantOrderAmount + | + secretKey**

\*Parameters within the hash string must not contain backslash ("\") and pipeline ("|") characters.

\*secretKey will be delivered by the PaySpot system.

\*Difference between CurrentDate parameter (**in UTC**) and time when the request is called can't be more than **20min**

**Example:**

plaintext= BrHklmcUE67kV0oK2oQT|2022-10-17 08:38:19|TestGenerate1|100.32|Test123456!

hash = Base64(SHA512(plaintext))

### **Callback URL Data**

ECommerce needs to predefined an URL endpoint where transaction status will be sent using HTTP POST method.

**CallbackURL** message input parameters

|  |  |  |  |
| --- | --- | --- | --- |
| **Parametar** | **Type** |  | **Description** |
| **{** |  |  |  |
| orderID | String | M | Merchant order from initial request |
| shopID | String | O(only for payment cards) | Identifier of the merchant’s shop assigned by Payment Gateway |
| authNumber | String | O(only for payment cards) | Authorization number.  The value is returned only in case of successful authorization. If authorization is not granted,  the field will be filled in with “NULL” |
| amount | String | M | Total value of the purchase order/purchase for which the payment was initiated |
| currency | String | M | Currency |
| transactionID | String | O(only for payment cards) | Identifier of transaction assigned by the PGW |
| result | String | M | Result of the transaction. Described below in section 5.4 |
| paySpotOrderID | String | O(only for IPS) | Unique IPS reference |
| rnd | String | O(only for IPS) | Random string used for hash calculation |
| hash | String | M | Hash calculated as explained in the sections below depending on payment method. **It is mandatory to calculate the hash on the eCommerce server.** |
| maskedPan | String | O(only for payment cards) | Masked Pan of card |
| expiryDate | String | O(only for payment cards) | Expiry date of card |
| cardBrand | String | O(only for payment cards) | Brand of card ( VISA,MASTERCARD,DINA, MAESTRO) |
| panAlias | String | O(only for payment cards) | Token of card |
| responseCode | String | O(only for eMoney) | Represents authorization code for eMoney transaction |
| responseMsg | String | O(only for eMoney) | Represents description of the authorization code for eMoney transaction |
| **}** |  |  |  |

#### **5.2.1 Hash Calculation for Payment by card**

This section explains how eCommerce needs to generate the hash to know that request come from PaySpot Payment Service.

**Hash – SHA 512 Algorithm**

The hash is created by doing the SHA512 algorithm from the given string, then doing base64 encoding. Parameters in the hash string must be separated by “|”.

**plaintext = orderID + | + shopID + | + amount + | + result + | + secretKeyCallback**

\*Parameters within the hash string must not contain backslash ("\") and pipeline ("|") characters.

\*secretKeyCallback will be delivered by the PaySpot system.

**Example:**

plaintext = order1234567890|shopIDTest|100.32|00|Test123456!

hash = Base64(SHA512(plaintext))

#### **5.2.2 Hash Calculation for Payment by IPS**

This section explains how eCommerce needs to generate the hash to know that request come from PaySpot Payment Service.

**Hash – SHA 512 Algorithm**

The hash is created by doing the SHA512 algorithm from the given string, then doing base64 encoding. Parameters in the hash string must be separated by “|”.

**plaintext = rnd + | + paySpotOrderID + | + orderID + | + secretKeyCallback**

\*Parameters within the hash string must not contain backslash ("\") and pipeline ("|") characters.

\*secretKey will be delivered by the PaySpot system.

**Example:**

plaintext= BrHklmcUE67kV0oK2oQT|I000009522290000001|TestGenerate1|Test123456!

hash = Base64(SHA512(plaintext))

### **API Completion Request**

**[POST]: /api/ECommerce/CompletionRequest**

**REQUEST PRODUCTION URL:**

**<https://www.nsgway.rs:50010/api/ECommerce/CompletionRequest>**

**REQUEST TEST URL:**

**<https://test.nsgway.rs:50009/api/ECommerce/CompletionRequest>**

Authorized (Pre-authorized) transactions can be completed using Web API service.  
If the authorization was successful, the values of parameters used for sending Completion request are  
obtained from the previous authorization step.  
If the authorization is not successful, it is impossible to make the  
completion.  
The transaction can be completed to the maximum extent of 30 days from the moment of  
purchase/pre-authorization. Beware that Voiding purchase/pre-authorization transaction is not available after 5 days have passed from the moment of purchase/pre-authorization, which means that every transaction that is not completed in that time period will only have Refund option and Void won't be available.

**AuthCompletionRequest** message input parameters

|  |  |  |  |
| --- | --- | --- | --- |
| **Parametar** | **Type** |  | **Description** |
| **{** |  |  |  |
| transactionId | String | M | Identifier of transaction assigned by the PGW |
| orderId | String | M | Order unique identifier |
| shopId | String | M | Identifier of the merchant’s shop assigned by Payment Gateway |
| amount | String | M | Total value of the purchase order/purchase for which the payment was initiated |
| currency | String | M | Currency |
| companyID | String | M | Unambiguous identification of the external system that executes the PaySpot web service call.  \*Will be delivered by the PaySpot system. |
| hash | String | M | Hash calculated as explained in the section 5.3.1. **It is mandatory to calculate the hash on the eCommerce server.** |
| rnd | String | M | Random string used for hash calculation |
| **}** |  |  |  |

**AuthCompletionRequest** message request example

|  |
| --- |
| {    "transactionId": "testTransactionID",    "orderId": "testOrderID",    "shopId": "testShopID",    "amount": "100.32",    "currency": "941",    "companyID": "123456",    "rnd": "mIlyKRws6GHHi8mlJtg6",    "hash": "4Oav8Pg9qfHWitVAk/BJOx/cujvdQfNV5tH07VvWPYxuT9nX3NuJAFrZCcdo7D28WeB92TzmB/R3xnFW43KkjQ==",    "currentDate": "2023-11-30 11:39"  } |

#### **5.3.1** **Hash calculation**

This section explains how eCommerce marketplace needs to generate the hash for submit request from eCommerce web site.

**Hash – SHA 512 Algorithm**

The hash is created by doing the SHA512 algorithm from the given string, then doing base64 encoding. Parameters in the hash string must be separated by “|”.

**plaintext = rnd + | + orderId + | + amount + | + secretKey**

\*Parameters within the hash string must not contain backslash ("\") and pipeline ("|") characters.

\*secretKey will be delivered by the PaySpot system.

**Example:**

plaintext= BrHklmcUE67kV0oK2oQT|1zd9k0o9|100.32|Test123456!

hash = Base64(SHA512(plaintext))

#### **5.3.2** **API Completion Response**

**AuthCompletionResponse** message response parameters

|  |  |  |  |
| --- | --- | --- | --- |
| **Parametar** | **Type** |  | **Description** |
| **{** |  |  |  |
| transactionId | String | M | Identifier of transaction assigned by the PGW |
| orderId | String | M | Order unique identifier |
| shopId | String | M | Identifier of the merchant’s shop assigned by Payment Gateway |
| amount | String | M | Total value of the purchase order/purchase for which the payment was initiated |
| currency | String | M | Currency |
| authNumber | String | M | Authorization number.  The value is returned only in case of successful authorization. If authorization is not granted,  the field will be filled in with “NULL” |
| result | String | M | Result of the transaction. Described below in section 5.4 |
| **}** |  |  |  |

**AuthCompletionResponse** message request example

|  |
| --- |
| {    "transactionId": "testTransactionID",    "orderId": "testOrderID",    "shopId": "testShopID",    "amount": "100.32",    "currency": "941",    "authNumber": "123456",    "result": "00"  } |

### **Response Result Field**

#### **5.4.1** **Card Payment**

The field **RESULT** can have the following values:

|  |  |
| --- | --- |
| Value | Description |
| 00 | Success |
| 01 | Denied by system |
| 02 | Denied due to store configuration issues |
| 03 | Denied due to communication issues with the authorization circuits |
| 04 | Denied by card issuer |
| 05 | Denied due to incorrect card number |
| 06 | Unforeseen error during processing of request |
| 07 | Duplicated order |
| 08 | Operator indicated not found |
| 09 | TRANSACTIONID indicated does not make reference to the entered ORDERID |
| 10 | Card not eligible for Installments |
| 11 | Incorrect status. Transaction not possible in the current status |
| 12 | Circuit disabled |
| 13 | Duplicated order |
| 16 | Currency not supported or not available for the merchant |
| 17 | Exponent not supported for the chosen currency |
| 20 | The card is VBV/SecureCode/SafeKey-enabled; the reply contains the data for  redirection to ACS website |
| 21 | Maximum time-limit for forwarding VBV request step 2 expired |
| 25 | A call to 3DS method must be performed by the Requestor |
| 26 | A challenge flow must be initiated by the Requestor |
| 35 | No payment instrument is acceptable |
| 37 | Missing CVV2: this is compulsory for the circuit selected |
| 38 | Pan alias not found or revoked |
| 40 | Empty Xml or missing 'data' parameter |
| 41 | Xml not parsable |
| 50 | Installments not available |
| 51 | Installment number out of bounds |

#### **5.4.2 IPS Payment**

The field **RESULT** can have the following values:

|  |  |
| --- | --- |
| Value | Description |
| 00 | Success |
| -1 | Denied |
| 82 | Timeout |

### **5.5 Get Order Status**

**[POST]: /api/ECommerce/CompletionRequest**

**REQUEST PRODUCTION URL:** [**https://www.nsgway.rs:50010/api/ECommerce/GetOrderStatus**](https://www.nsgway.rs:50010/api/ECommerce/GetOrderStatus)

**REQUEST TEST URL: <https://test.nsgway.rs:50009/api/ECommerce/GetOrderStatus>**

The service is provided for merchants who want to check the status of the transaction.

**GetOrderStatus** message input parameters

|  |  |  |  |
| --- | --- | --- | --- |
| **Parametar** | **Type** |  | **Description** |
| **{** |  |  |  |
| orderId | String | M | Order unique identifier |
| shopId | String | M | Identifier of the merchant’s shop assigned by Payment Gateway |
| companyID | String | M | Unambiguous identification of the external system that executes the PaySpot web service call.  \*Will be delivered by the PaySpot system. |
| hash | String | M | Hash calculated as explained in the section 5.5.1. **It is mandatory to calculate the hash on the eCommerce server.** |
| rnd | String | M | Random string used for hash calculation |
| **}** |  |  |  |

**GetOrderStatus** message request example

|  |
| --- |
| {    "orderId": "testOrderID",    "shopId": "testshopID",    "companyID": "123456",    "rnd": "mIlyKRws6GHHi8mlJtg6",    "hash": "4Oav8Pg9qfHWitVAk/BJOx/cujvdQfNV5tH07VvWPYxuT9nX3NuJAFrZCcdo7D28WeB92TzmB/R3xnFW43KkjQ=="  } |

#### **5.5.1** **Hash calculation**

This section explains how eCommerce marketplace needs to generate the hash for submit request from eCommerce web site.

**Hash – SHA 512 Algorithm**

The hash is created by doing the SHA512 algorithm from the given string, then doing base64 encoding. Parameters in the hash string must be separated by “|”.

**plaintext = rnd + | + orderId + | + secretKey**

\*Parameters within the hash string must not contain backslash ("\") and pipeline ("|") characters.

\*secretKey will be delivered by the PaySpot system.

**Example:**

plaintext= BrHklmcUE67kV0oK2oQT|1zd9k0o9|Test123456!

hash = Base64(SHA512(plaintext))

#### **5.5.2** **Get Order Status response**

**GetOrderStatus** message output parameters

|  |  |  |
| --- | --- | --- |
| **Parametar** | **Type** | **Description** |
| **{** |  |  |
| transactionId | String | Identifier of transaction assigned by the PGW |
| orderId | String | Merchant order unique identifier |
| shopId | String | Identifier of the merchant’s shop assigned by Payment Gateway |
| transactionAmount | String | The transaction amount in currency unit |
| authorizedAmount | String | The authorized amount in currency unit |
| accountedAmount | String | The accounted amount in currency unit |
| refundedAmount | String | The refunfed amount in currency unit |
| transactionStatus | String | Current status of the transaction . Described below in section 5.5.3.1 |
| transactionResult | String | Outcome of transaction. Described below in section 5.5.3.2 |
| authorizationNumber | String | Authorization number.  The value is returned only in case of successful authorization. If authorization is not granted,  the field will be filled in with “NULL” |
| transactionDate | String | Date and time of transaction in yyyy-mm-ggTHH:mm:ss format |
| currency | String | Currency |
| **}** |  |  |

**GetOrderStatus**message response example

|  |
| --- |
| {    "transactionId": "testTransactionID",    "orderId": "testOrderID",    "shopId": "testshopID",    "authorizedAmount": "110.32",    "transactionAmount": "110.32",    "accountedAmount": "110.32",    "refundedAmount": "0",    "transactionStatus": "03",    "transactionResult": "00",    "currency": "941",    "authorizationNumber": "123456",    "transactionDate": "2024-01-10T09:22:53"  } |

#### **5.5.3.1 Transaction Status codes**

The field **transactionStatus** can have the following values:

|  |  |
| --- | --- |
| Value | Description |
| 00 | Preauthorization |
| 01 | Preauthorization denied |
| 02 | Authorization to be processed |
| 03 | Authorization processed by clearing |
| 04 | Refund |
| 21 | Authorization to be reversed due to transaction error |
| 99 | Authorization underway with MyBank or  BancomatPay |

#### **5.5.3.2 Transaction Result codes**

The field **transactionResult** can have the following values:

|  |  |
| --- | --- |
| Value | Description |
| 00 | Success |
| 01 | Denied due to problems in the request message |
| 02 | Denied due to problems in the store registry |
| 03 | Denied due to communication problems with the authorization circuits |
| 04 | Denied by card issuer |
| 05 | Denied due to incorrect card number |
| 06 | Unforeseen error during processing of request |
| 10 | Card not eligible for Installments |
| 45 | Denied authorization due to failed antifraud check. |
| 51 | Installment number out of bounds (acquirer side) |
| 99 | Authorization underway with MyBank or BancomatPay |

### **5.6 Refund / Void Transaction**

**[POST]: /api/ECommerce/RefundTransaction**

**REQUEST PRODUCTION URL:** [**https://www.nsgway.rs:50010/api/ECommerce/RefundTransaction**](https://www.nsgway.rs:50010/api/ECommerce/GetOrderStatus)

**REQUEST TEST URL: [https://test.nsgway.rs:50009/api/ECommerce/RefundTransaction](https://test.nsgway.rs:50009/api/ECommerce/GetOrderStatus)**

The service is provided for merchants who want to void the transaction before the transaction was accounted (before financial transaction happened in bank). This window for Auth transactions (Immediate Accounting) is day of the transaction until midnight, while for PreAuthorized transaction (Deffered Accounting) the window for voiding the transaction is 5 days from the day since transaction happened. Transactions can be voided only for the full amount, there is no partial void therefore Amount field plays no role in Void transactions only in Refund. All transactions sent to this endpoint that can't be processed as Void will be saved in our system and processed as Refund transactions which are done through our Call Center. These Refund transactions can be partial and time window is declared by the issuing bank. Multiple refunds are allowed until the initial amount of the transaction has been fully refunded.

**RefundTransaction** message input parameters

|  |  |  |  |
| --- | --- | --- | --- |
| **Parametar** | **Type** |  | **Description** |
| **{** |  |  |  |
| orderId | String | M | Order unique identifier |
| shopId | String | M | Identifier of the merchant’s shop assigned by Payment Gateway |
| transactionId | String |  | Identifier of transaction assigned by the PGW |
| companyID | String | M | Unambiguous identification of the external system that executes the PaySpot web service call.  \*Will be delivered by the PaySpot system. |
| amount | String | M | Amount that is going to be refunded (in case of Void it will always be full amount of the transaction as there is no partial Void transaction). |
| hash | String | M | Hash calculated as explained in the section 5.5.1. **It is mandatory to calculate the hash on the eCommerce server.** |
| rnd | String | M | Random string used for hash calculation |
| **}** |  |  |  |

**RefundTransaction** message request example

|  |
| --- |
| {    "orderId": "testOrderID",  "transactionId": "testTransactionID",    "shopId": "testShopID",    "companyID": "123456",  "amount": "100.32",    "rnd": "mIlyKRws6GHHi8mlJtg6",    "hash": "4Oav8Pg9qfHWitVAk/BJOx/cujvdQfNV5tH07VvWPYxuT9nX3NuJAFrZCcdo7D28WeB92TzmB/R3xnFW43KkjQ=="  } |

#### **5.6.1** **Hash calculation**

This section explains how eCommerce marketplace needs to generate the hash for submit request from eCommerce web site.

**Hash – SHA 512 Algorithm**

The hash is created by doing the SHA512 algorithm from the given string, then doing base64 encoding. Parameters in the hash string must be separated by “|”.

**plaintext = rnd + | + orderId + | + amount + | + shopId + | + secretKey**

\*Parameters within the hash string must not contain backslash ("\") and pipeline ("|") characters.

\*secretKey will be delivered by the PaySpot system.

**Example:**

plaintext= BrHklmcUE67kV0oK2oQT|1zd9k0o9|100.32|TestShop123|Test123456!

hash = Base64(SHA512(plaintext))

#### **5.6.2** **Void / Refund Transaction Response**

**RefundTransaction** message output parameters

|  |  |  |
| --- | --- | --- |
| **Parametar** | **Type** | **Description** |
| **{** |  |  |
| transactionId | String | Identifier of transaction assigned by the PGW |
| orderId | String | Merchant order unique identifier |
| shopId | String | Identifier of the merchant’s shop assigned by Payment Gateway |
| transactionAmount | String | The transaction amount in currency unit |
| authorizedAmount | String | The authorized amount in currency unit |
| accountedAmount | String | The accounted amount in currency unit |
| refundedAmount | String | The refunfed amount in currency unit |
| transactionStatus | String | Current status of the transaction . Described below in section 4.3.3.1 |
| transactionResult | String | Outcome of transaction. Described below in section 4.3.3.2 |
| authorizationNumber | String | Authorization number.  The value is returned only in case of successful authorization. If authorization is not granted,  the field will be filled in with “NULL” |
| transactionDate | String | Date and time of transaction in yyyy-mm-ggTHH:mm:ss format |
| currency | String | Currency |
| **}** |  |  |

**RefundTransaction**message response example

|  |
| --- |
| {    "transactionId": "testTransactionID",    "orderId": "testOrderID",    "shopId": "testshopID",    "authorizedAmount": "110.32",    "transactionAmount": "110.32",    "accountedAmount": "110.32",    "refundedAmount": "0",    "transactionStatus": "03",    "transactionResult": "00",    "currency": "941",    "authorizationNumber": "123456",    "transactionDate": "2024-01-10T09:22:53"  } |

### **5.7 Client Delete Saved Card Data**

**[POST]: /api/ECommerce/ClientDeleteSavedCardData**

**REQUEST PRODUCTION URL:** [**https://www.nsgway.rs:50010/api/ECommerce/ClientDeleteSavedCardData**](https://www.nsgway.rs:50010/api/ECommerce/ClientDeleteSavedCardData)

**REQUEST TEST URL:**

**<https://test.nsgway.rs:50009/api/ECommerce/ClientDeleteSavedCardData>**

The service is provided for merchants who are using our CIT (Client Initiated Transaction) flow which includes saving the card token (tokenization) on the client side for further payments. This service deletes the tokenized data stored on our servers and removes the card from our frontend meaning that user can't use it for payments anymore.

**ClientDeleteSavedCardData** message input parameters

|  |  |  |  |
| --- | --- | --- | --- |
| **Parametar** | **Type** |  | **Description** |
| **{** |  |  |  |
| panAlias | String | M | Represents a card token |
| maskedPan | String | M | Represents a masked version of full card PAN, it includes first 6 and last 4 digits of the card PAN |
| shopId | String | M | Identifier of the merchant’s shop assigned by Payment Gateway |
| companyID | Int | M | Unambiguous identification of the external system that executes the PaySpot web service call.  \*Will be delivered by the PaySpot system. |
| customerId | String | O | ID which is an alternative to email for CIT transactions, can be used to store user's saved cards in case email is not provided. |
| customerEmail | String | O | User email which is used to store tokens, either customerId or customerEmail must be sent |
| amount | String | M | Amount that is going to be refunded (in case of Void it will always be full amount of the transaction as there is no partial Void transaction). |
| hash | String | M | Hash calculated as explained in the section 5.5.1. **It is mandatory to calculate the hash on the eCommerce server.** |
| rnd | String | M | Random string used for hash calculation |
| **}** |  |  |  |

**ClientDeleteSavedCardData** message request example

|  |
| --- |
| {    "PanAlias": "1234567890",  "maskedPan": "534223xxxxxx1234",    "shopId": "testShopID",    "companyID": 123456,  "clientId": "123456",    "rnd": "mIlyKRws6GHHi8mlJtg6",    "hash": "4Oav8Pg9qfHWitVAk/BJOx/cujvdQfNV5tH07VvWPYxuT9nX3NuJAFrZCcdo7D28WeB92TzmB/R3xnFW43KkjQ=="  } |

#### **5.7.1** **Hash calculation**

This section explains how eCommerce marketplace needs to generate the hash for submit request from eCommerce web site.

**Hash – SHA 512 Algorithm**

The hash is created by doing the SHA512 algorithm from the given string, then doing base64 encoding. Parameters in the hash string must be separated by “|”.

**plaintext = rnd + | + panAlias + | + secretKey**

\*Parameters within the hash string must not contain backslash ("\") and pipeline ("|") characters.

\*secretKey will be delivered by the PaySpot system.

**Example:**

plaintext= mIlyKRws6GHHi8mlJtg6|01231231231|Test123456!

hash = Base64(SHA512(plaintext))

#### **5.7.2** **Client Delete Saved Card Data Response**

**ClientDeleteSavedCardData** message output parameters

|  |  |  |
| --- | --- | --- |
| **Parametar** | **Type** | **Description** |
| **{** |  |  |
| panAlias | String | Represents a card token that was deleted |
| maskedPan | String | Represents a masked version of full card PAN, it includes first 6 and last 4 digits of the card PAN that was deleted |
| shopId | String | Identifier of the merchant’s shop assigned by Payment Gateway |
| companyId | String | Unambiguous identification of the external system that executes the PaySpot web service call.  \*Will be delivered by the PaySpot system. |
| customerId | String | ID which is an alternative to email for CIT transactions, can be used to store user's saved cards in case email is not provided. |
| customerEmail | String | User email which is used to store tokens, either customerId or customerEmail must be sent |
| errorCode | Int | Error code |
| errorMessage | String | Description of the Error code |
| statusAlias | String | The current status of the saved token |
| **savedCards [ {** |  | Array of saved cards |
| panAlias | String | Represents a card token |
| maskedPan | String | Represents a masked version of full card PAN, it includes first 6 and last 4 digits of the card PAN |
| panExpiryDate | String | Date of expiration of the tokenized card in format YYMM |
| shopId | String | Identifier of the merchant’s shop assigned by Payment Gateway |
| treCurr | String | Type of recurring transaction, **C**-Client Initiated Transaction, **U**-Unscheduled Merchant initiated recurring transaction and **R**-Recurring Merchant initiated transaction |
| creCurr | String | Identifier that connects initial recurring tansaction with following recurring transactions |
| **] }** |  |  |
| hash | String | Hash that is used to verify that the response is actually comming from our system, it's calculated according to specification in section 5.8.1 |

**ClientDeleteSavedCardData** message response example

|  |
| --- |
| {    "panAlias": "1234567890",    "maskedPan": "534223xxxxxx1234",  "companyId": 123456,  "customerId": "7890123",  "customerEmail": "test@gmail.com",    "shopId": "testshopID",    "errorCode": 0,    "errorMessage": "OK",    "statusAlias": 0,    "savedCards": [          {              "panAlias": "0000487552113990718",              "maskedPan": "534223xxxxxx1234",              "panExpiryDate": "2801",              "shopID": "80729SE00124302",              "treCurr": "C",              "creCurr": "COF0123456789"          },          {              "panAlias": "0000700670416298520",              "maskedPan": "534223xxxxxx1234",              "panExpiryDate": "3512",              "shopID": "80729SE00124302",              "treCurr": "C",              "creCurr": "COF0123456789"          }      ],  "hash": "aVdkL3zechCDz5FJJZKaP9cYTk7jVsLUWEdVBmw71zKbIBGM9pOsAGk1htg=="  } |

#### **5.7.3** **Client Delete Saved Card Callback URL**

ECommerce needs to predefined an URL endpoint where deleted credit card info will be sent using HTTP POST method.

**Client Delete Saved Card Callback** message input parameters

|  |  |  |  |
| --- | --- | --- | --- |
| **Parametar** | **Type** |  | **Description** |
| **{** |  |  |  |
| panAlias | String | M | Token of card |
| maskedPan | String | M | Masked PAN number of the card which represents first 6 and the last 4 digits of the card's PAN |
| companyID | String | M | Customer identifier that is used to link the PAN and Card to specific user |
| shopID | String | M | Identifier of the merchant’s shop assigned by Payment Gateway |
| customerId | String | O | ID which is an alternative to email for CIT transactions, can be used to store user's saved cards in case email is not provided. |
| customerEmail | String | O | User email which is used to store tokens, either customerId or customerEmail must be sent |
| errorCode | Int | M | Error code |
| errorMessage | String | M | Description of the Error code |
| statusAlias | String | M | The current status of the saved token |
| **savedCards [ {** |  | M | Array of saved cards |
| panAlias | String | M | Token of card |
| maskedPan | String | M | Represents a masked version of full card PAN, it includes first 6 and last 4 digits of the card PAN |
| panExpiryDate | String | M | Date of expiration of the tokenized card in format YYMM |
| shopId | String | M | Identifier of the merchant’s shop assigned by Payment Gateway for this specific token |
| treCurr | String | M | Type of recurring transaction, **C**-Client Initiated Transaction, **U**-Unscheduled Merchant initiated recurring transaction and **R**-Recurring Merchant initiated transaction |
| creCurr | String | M | Identifier that connects initial recurring transaction with following recurring transactions |
| **] }** |  |  |  |
| hash | String | M | Hash calculated as explained in the section 5.8.1. **It is mandatory to calculate the hash on the eCommerce server.** |

##### **5.7.3.1 Hash Calculation for Client Delete saved Card Callback**

This section explains how eCommerce needs to generate the hash to know that request come from PaySpot Payment Service.

**Hash – SHA 512 Algorithm**

The hash is created by doing the SHA512 algorithm from the given string, then doing base64 encoding. Parameters in the hash string must be separated by “|”.

**plaintext = shopID + | + panAlias + | + maskedPan + | + statusAlias + | + secretKeyCallback**

\*Parameters within the hash string must not contain backslash ("\") and pipeline ("|") characters.

\*secretKeyCallback will be delivered by the PaySpot system.

**Example:**

plaintext = shopIDTest|0000624648507414609|534223xxxxxx1234|0|Test123456!

hash = Base64(SHA512(plaintext))

### **5.8 Merchant Initiate Transactions (MIT)**

#### **5.8.1 Registration card Request**

**[POST]: /api/ECommerce/GetPaymentParams**

**REQUEST PRODUCTION URL:** [**https://www.nsgway.rs:50010/api/ECommerce/GetPaymentParams**](https://www.nsgway.rs:50010/api/ECommerce/GetPaymentParams)

**REQUEST TEST URL:**

**<https://test.nsgway.rs:50009/api/ECommerce/GetPaymentParams>**

**RegistrationCard** message input parameters

|  |  |  |  |
| --- | --- | --- | --- |
| **Parametar** | **Type** |  | **Description** |
| **{** |  |  |  |
| merchantOrderID | String | M | The unique reference of the purchase order, generated in the eCommerce or merchant system. AN(40). |
| merchantOrderAmount | Decimal | M | Total value of the purchase order/purchase for which the payment was initiated. D (15.2)  Contains the total value of goods and/or services + the amount of additional fee that the external partner system may charge for a transaction |
| merchantCurrencyCode | Int | M | Currency code for purchase order. ISO numeric code. The default is the national currency (941 for RSD) |
| language | Int | O | Language ID (1=default (sr), 2=eng ... ) |
| customerEmail | String | M | Customer’s e-mail address |
| customerId | String | O | ID which is an alternative to email for CIT/MIT transactions, can be used to store user's saved cards in case email is not provided. |
| panAlias | String | O | Merchant token for payment card |
| successURL | String | M | eCommerce has to provide an URL endpoint where the customer browser will be redirected after successful payment |
| cancelURL | String | M | eCommerce has to provide an URL endpoint where the customer browser will be redirected if customer quits purchase before payment |
| errorURL | String | M | eCommerce has to provide an URL endpoint where the customer browser will be redirected in the case of unsuccessful payment |
| requestType | Int | M | Request type in PaySpot System. ( Marketplace platform = 10, Web shop platform =11) |
| companyID | Int | M | Unambiguous identification of the external system that executes the PaySpot web service call.  \*Will be delivered by the PaySpot system. |
| hash | String | M | Hash calculated as explained in the section 5.8.1.3. **It is mandatory to calculate the hash on the eCommerce server.** |
| rnd | String | M | Random string used for hash calculation |
| currentDate | String | M | Date and time of the submit request in the format: YYYY-MM-DD hh:mm:ss (ISO). **Must be sent in UTC timezone,** |
| transactionType | String | M | Type of transaction .  -REDIRECT\_MIT |
| } |  |  |  |

##### **5.8.1.1 Registration Card Request**

**RegistrationCardRequest** message request example

|  |
| --- |
| {  "merchantOrderID":"testoid",           "merchantOrderAmount":1.00,           "merchantCurrencyCode":941,           "language":1,           "customerId":"123456",           "customerEmail":"test@gmail.com",           "successURL":"https://test.com",           "cancelURL":"https://test.com",           "errorUrl":"https://test.com",           "panAlias":"1234567890",           "requestType":10,           "companyID":123456,           "hash":"hashstring",           "transactionType":"REDIRECT\_MIT",           "rnd":"randomstring",           "currentDate":"2024-02-02 08:38:19"        } |

##### **5.8.1.2 RegistrationCard Response**

**RegistrationCardResponse** message response parameters

|  |  |  |  |
| --- | --- | --- | --- |
| **Parametar** | **Type** |  | **Description** |
| **{** |  |  |  |
| actionUrl | String | M | Url to call PGW |
| orderId | String | M | Order unique identifier |
| shopId | String | M | Identifier of the merchant’s shop assigned by Payment Gateway |
| amount | String | M | Total value of the purchase order/purchase for which the payment was initiated |
| currency | String | M | Currency |
| lang | String | M | Language of payment page |
| urlback | String | M | Complete URL to which the user is to be redirected to go to  the store in case  the payment process is cancelled. |
| urldone | String | M | Complete URL to which the customer’s browser is to be  redirected once the transaction has been successfully  completed. |
| urlms | String | M | Callback URL ( It will be PaySpot CallbackURL in that case) |
| page | String | M | For redirect value is „LAND“ |
| options | String | M | It contains the indicators of the additional options to be  activated for the payment. |
| mac | String | M | Hash string |
| email | String | M | Customer’s e-mail address |
| authormode | String | M | Will be „I“ |
| accountingmode | String | M | Type of booking to be used for this order:  - D deferred (Preauth)  - I immediate (Auth) |
| exponent | String | M | Number of decimals for the currency |
| trecurr | String | M | Value „U“ for MIT Unschedule |
| **}** |  |  |  |

**RegistrationCardResponse** message request example

|  |
| --- |
| {      "actionUrl": "https://virtualpostest.sia.eu/vpos/payments/main",      "lang": "EN",      "page": "LAND",      "amount": "100",      "currency": "941",      "orderid": "testoid",      "shopid": "80729SE00124301",      "urlback": "https://test.com",      "urldone": "https://test.com",      "urlms": "https://test.com",      "accountingmode": "D",      "authormode": "I",      "mac": "0da4ffed412c67d67ade613fd6c812d882e523c6f32386fbc46196513be97b13",      "email": "test@gmail.com",      "options": "M",      "exponent": "2",      "trecurr": "U",   } |

##### **5.8.1.3 Hash calculation**

This section explains how eCommerce needs to generate the hash for submit request from eCommerce web site.

**Hash – SHA 512 Algorithm**

The hash is created by doing the SHA512 algorithm from the given string, then doing base64 encoding. Parameters in the hash string must be separated by “|”.

**plaintext = rnd + | + currentDate + | + merchantOrderID + | + merchantOrderAmount + | + secretKey**

\*Parameters within the hash string must not contain backslash ("\") and pipeline ("|") characters.

\*secretKey will be delivered by the PaySpot system.

**Example:**

plaintext= BrHklmcUE67kV0oK2oQT|2022-10-17 08:38:19|TestGenerate1|100.32|Test123456!

hash = Base64(SHA512(plaintext))

##### **5.8.1.4 Form Example**

 <form

        method="post"

        data-testid="hidden-form"

        action="https://virtualpostest.sia.eu/vpos/payments/main">

        <input type="hidden" name="PAGE" value="LAND" />

        <input type="hidden" name="AMOUNT" value="100" />

        <input type="hidden" name="CURRENCY" value="941" />

        <input type="hidden" name="LANG" value="SR" />

        <input type="hidden" name="SHOPID" value="80729SE00124301" />

        <input type="hidden" name="ORDERID" value="testoid" />

        <input type="hidden" name="URLDONE" value="https://test.com" />

        <input type="hidden" name="URLBACK" value="https://test.com" />

        <input type="hidden" name="URLMS" value="https://test.nsgway.rs/api/Ecommerce/OtpResult" />

        <input type="hidden" name="ACCOUNTINGMODE" value="D" />

        <input type="hidden" name="AUTHORMODE" value="I" />

        <input type="hidden" name="OPTIONS" value="M" />

        <input type="hidden" name="EMAIL" value="test@gmail.com" />

        <input type="hidden" name="TRECURR" value="U" />

        <input type="hidden" name="EXPONENT" value="2" />

        <input

          type="hidden"

          name="MAC"

          value="0da4ffed412c67d67ade613fd6c812d882e523c6f32386fbc46196513be97b13"

        />

        <button

          type="submit"

          <p>Register card</p>

        </button>

      </form>

#### **5.8.2 Authorize Request**

**[POST]: /api/ECommerce/AuthorizeRequest**

**REQUEST PRODUCTION URL:** [**https://www.nsgway.rs:50010/api/ECommerce/AuthorizeRequest**](https://www.nsgway.rs:50010/api/ECommerce/AuthorizeRequest)

**REQUEST TEST URL:**

**<https://test.nsgway.rs:50009/api/ECommerce/AuthorizeRequest>**

**AuthorizeRequest** message input parameters

|  |  |  |  |
| --- | --- | --- | --- |
| **Parametar** | **Type** |  | **Description** |
| **{** |  |  |  |
| orderId | String | M | The unique reference of the purchase order, generated in the eCommerce or merchant system. AN(40). |
| amount | Decimal | M | Total value of the purchase order/purchase for which the payment was initiated. D (15.2)  Contains the total value of goods and/or services + the amount of additional fee that the external partner system may charge for a transaction |
| currency | String | M | Currency code for purchase order. ISO numeric code. The default is the national currency (941 for RSD) |
| customerEmail | String | M | Customer’s e-mail address |
| customerId | String | O | ID which is an alternative to email for CIT/MIT transactions, can be used to store user's saved cards in case email is not provided. |
| panAlias | String | O | Merchant token for payment card |
| shopId | String | M | Identifier of the merchant’s shop assigned by Payment Gateway |
| transactionType | String | M | Type of transaction ( REDIRECT/REDIRECT\_MIT/CIT/MIT\_UNSC ) |
| companyID | String | M | Unambiguous identification of the external system that executes the PaySpot web service call.  \*Will be delivered by the PaySpot system. |
| requestType | Int | M | Request type in PaySpot System. ( Marketplace platform = 10, Web shop platform =11) |
| hash | String | M | Hash calculated as explained in the section 5.8.2.6. **It is mandatory to calculate the hash on the eCommerce server.** |
| rnd | String | M | Random string used for hash calculation |
| currentDate | String | M | Date and time of the submit request in the format: YYYY-MM-DD hh:mm:ss (ISO). **Must be sent in UTC timezone,** |
| } |  |  |  |

##### **5.8.2.1 Authorize Request**

**AuthorizeRequest** message request example

|  |
| --- |
| {        "orderid": "testorder123",        "amount": 100.11,        "currency": "941",        "customerId": "12345",        "customerEmail": "test@gmail.com",        "panAlias": "testPanAlias",        "shopId": "80729SE00124303",  "requestType":11,        "companyID":123456,        "hash":"hashstring",        "rnd":"rnd",        "currentDate":"2024-02-02 08:38:19",        "transactionType": "MIT\_UNSC"      } |

##### **5.8.2.2 Authorize response**

**AuthorizeResponse** message output parameters

|  |  |  |
| --- | --- | --- |
| **Parametar** | **Type** | **Description** |
| **{** |  |  |
| transactionId | String | Identifier of transaction assigned by the PGW |
| orderId | String | Merchant order unique identifier |
| shopId | String | Identifier of the merchant’s shop assigned by Payment Gateway |
| transactionAmount | String | The transaction amount in currency unit |
| authorizedAmount | String | The authorized amount in currency unit |
| accountedAmount | String | The accounted amount in currency unit |
| refundedAmount | String | The refunfed amount in currency unit |
| transactionStatus | String | Current status of the transaction . Described below in section 5.8.2.3 |
| transactionResult | String | Outcome of transaction. Described below in section 5.8.2.4 |
| authorizationNumber | String | Authorization number.  The value is returned only in case of successful authorization. If authorization is not granted,  the field will be filled in with “NULL” |
| transactionDate | String | Date and time of transaction in yyyy-mm-ggTHH:mm:ss format |
| currency | String | Currency |
| result | String | Result of transaction |
| resultDescription | String | Description of result field. Described below in |
| **}** |  |  |

**AuthorizeResponse** message response example

|  |
| --- |
| {    "transactionId": "testTransactionID",    "orderId": "testOrderID",    "shopId": "testshopID",    "authorizedAmount": "110.32",    "transactionAmount": "110.32",    "accountedAmount": "110.32",    "refundedAmount": "0",    "transactionStatus": "03",    "transactionResult": "00",    "currency": "941",    "authorizationNumber": "123456",    "transactionDate": "2024-01-10T09:22:53",    "result": "00",    "resultDescription": "Success",  } |

##### **5.8.2.3 Transaction Status codes**

The field **transactionStatus** can have the following values:

|  |  |
| --- | --- |
| Value | Description |
| 00 | Preauthorization |
| 01 | Preauthorization denied |
| 02 | Authorization to be processed |
| 03 | Authorization processed by clearing |
| 04 | Refund |
| 21 | Authorization to be reversed due to transaction error |
| 99 | Authorization underway with MyBank or  BancomatPay |

##### **5.8.2.4 Transaction Result codes**

The field **transactionResult** can have the following values:

|  |  |
| --- | --- |
| Value | Description |
| 00 | Success |
| 01 | Denied due to problems in the request message |
| 02 | Denied due to problems in the store registry |
| 03 | Denied due to communication problems with the authorization circuits |
| 04 | Denied by card issuer |
| 05 | Denied due to incorrect card number |
| 06 | Unforeseen error during processing of request |
| 10 | Card not eligible for Installments |
| 45 | Denied authorization due to failed antifraud check. |
| 51 | Installment number out of bounds (acquirer side) |
| 99 | Authorization underway with MyBank or BancomatPay |

##### **5.8.2.5 Result codes**

The field **result** can have the following values:

|  |  |
| --- | --- |
| Value | Description |
| 00 | Success |
| 01 | Order or ReqRefNum not found |
| 02 | ReqRefNum duplicated or not valid |
| 03 | Incorrect message format, missing or incorrect field |
| 04 | Incorrect API authentication, incorrect MAC or timestamp exceeding the limit range |
| 05 | Incorrect date, or period indicated is empty |
| 06 | Unforeseen error during processing of request |
| 07 | TransactionID not found |
| 08 | Operator indicated not found |
| 09 | TRANSACTIONID indicated does not make reference to the entered ORDERID |
| 10 | Amount indicated exceeds maximum amount permitted |
| 11 | Incorrect status. Transaction not possible in the current status |
| 12 | Circuit disabled |
| 13 | Duplicated order |
| 16 | Currency not supported or not available for the merchant |
| 17 | Exponent not supported for the chosen currency |
| 20 | The card is VBV/SecureCode/SafeKey-enabled; the reply contains the data for redirection to ACS website |
| 21 | Maximum time-limit for forwarding VBV request step 2 expired |
| 25 | A call to 3DS method must be performed by the Requestor |
| 26 | A challenge flow must be initiated by the Requestor |
| 35 | No payment instrument is acceptable |
| 37 | Missing CVV2: this is compulsory for the circuit selected |
| 38 | Pan alias not found or revoked 40 Empty Xml or missing ‘data’ parameter |
| 41 | Xml not parsable |
| 50 | Installments not available |
| 51 | Installment number out of bounds (client side) |
| 98 | Application error |
| 99 | Transaction failed, see specific outcome attached to the element of the reply. |

##### **5.8.2.6 Hash calculation**

This section explains how eCommerce needs to generate the hash for submit request from eCommerce web site.

**Hash – SHA 512 Algorithm**

The hash is created by doing the SHA512 algorithm from the given string, then doing base64 encoding. Parameters in the hash string must be separated by “|”.

**plaintext = rnd + | + currentDate + | + orderId + | + amount + | + secretKey**

\*Parameters within the hash string must not contain backslash ("\") and pipeline ("|") characters.

\*secretKey will be delivered by the PaySpot system.

**Example:**

plaintext= BrHklmcUE67kV0oK2oQT|2022-10-17 08:38:19|TestGenerate1|100.32|Test123456!

hash = Base64(SHA512(plaintext))

### **5.9 Pay By Link / Pay Per Link (PPL)**

#### **5.9.1 Generate Link**

**[POST]: api/ECommerce/PayByLink/CreateLink**

**REQUEST PRODUCTION URL:** [**https://www.nsgway.rs:50010/**](https://www.nsgway.rs:50010/api/ECommerce/AuthorizeRequest)**api/ECommerce/PayByLink/CreateLink**

**REQUEST TEST URL:**

**<https://test.nsgway.rs:50009/api/ECommerce/PayByLink/CreateLink>**

**GenerateLink** message input parameters

|  |  |  |  |
| --- | --- | --- | --- |
| **Parametar** | **Type** |  | **Description** |
| **{** |  |  |  |
| sendEmail | String | M | This field indicates if the email should or shouldn't be sent by the payment gateway system. AN(1). |
| linkExpirationDate | String | O | Indication of the date of the end of validity of the link in the format yyyy-MM-ddTHH:mm:ss.SSS, if it has no value, the validity indicated in the store registered data will be used. AN(24) |
| linkAmount | Decimal | M | Link amount in specified currency. Decimal(15.2) |
| linkCurrency | String | M | Currency code for purchase order. ISO numeric code. The default is the national currency (941 for RSD) |
| linkOrderId | String | M | The unique reference of the purchase order for the specified link, generated in the eCommerce or merchant system. Once the link has been paid OrderID is no longer repeatable and can't be used again. AN(40). |
| linkUrlDone | String | M | Complete URL to which the client’s browser is to be redirected upon successful outcome of the transaction (it may include all of the parameters to be passed). AN(200) |
| linkLang | String | O | Language in which the messages of interaction with the final user must be shown. Options are 'EN' and 'IT', with 'EN' being default. AN(2) |
| linkShopEmail | String | O | It contains the email address to which the transaction outcome is to be sent. If it is not present, the one available in the store data registered at payment gateway will be used. AN(50) |
| linkEmail | String | M | Email address of client addressee of the link. AN(50) |
| linkName | String | M | Name of holder of payment instrument. AN(40) |
| linkSurname | String | M | Last name of holder of payment instrument. AN(40) |
| shopId | String | M | Identifier of the merchant’s shop assigned by Payment Gateway |
| companyID | String | M | Unambiguous identification of the external system that executes the PaySpot web service call.  \*Will be delivered by the PaySpot system. |
| hash | String | M | Hash calculated as explained in the section 5.9.1.3. **It is mandatory to calculate the hash on the eCommerce server.** |
| rnd | String | M | Random string used for hash calculation |
| } |  |  |  |

##### **5.9.1.1 Generate Link Request**

GenerateLink message request example

|  |
| --- |
| **{**    "sendEmail": "Y",    "linkExpirationDate": "2024-11-30T22:00:00.000",    "linkAmount": 9900.00,    "linkCurrency": "941",    "linkOrderId": "PBL17",    "linkUrlDone": "https://online-test.payspot.rs/login",    "linkLang": "EN",    "linkShopEmail": "nikolav54@gmail.com",    "linkEmail": "nikola.vukovic@payspot.co.rs",    "linkName": "Nikola",    "linkSurname": "Vukovic",    "shopId": "80729SE00124301",    "companyID": 99019,    "rnd": "string",    "hash": "Bn+wuilTgrYjr1Rgz3pgE54lefSUjgAPQkk7JEvACbaK9A2ELgnNny7p2XuIcldpEswdKyZt2chIAizk0p3gIQ=="  **}** |

##### **5.9.1.2 Generate Link Response**

**GenerateLink** message output parameters

|  |  |  |
| --- | --- | --- |
| **Parametar** | **Type** | **Description** |
| **{** |  |  |
| result | String | Result of transaction. Possible values of this field are listed in the section 5.9.1.4 |
| timestamp | String | Timestamp of the response in format: yyyy-MM-ddThh:mm:ss |
| mac | String | Hash from the payment gateway that is used for checking the integrity of the transaction. |
| completeLink | String | Generated link that can be sent to the client for payment. |
| token | String | This represents the unique token for this specific link, it is also included in the completeLink field itself. |
| creationDate | String | Timestamp of the link's creation date in format: yyyy-MM-ddThh:mm:ss |
| status | String | Current status of the generated link, it can have the following values: **00** - Created The link has been created correctly (technical status, not visible from the outside) **01** - Returned The link has been communicated to the operator **02** -Sent The link has been sent to the cardholder **03** - Used The link has been used at least once **04** - Paid The payment tied to the link has been completed **05** - Revoked The link has been revoked |
| expirationDate | String | This represents the date until which the created link is active and payable. After this date has passed the link can no longer be paid. Format: yyyy-MM-ddThh:mm:ss |
| orderId | String | The unique reference of the purchase order for the specified link, generated in the eCommerce or merchant system. Once the link has been paid OrderID is no longer repeatable and can't be used again. AN(40). |
| linkOptions | String | It contains the indicators of the additional options that are to be activated for the payment under way. The order in which the options appear is irrelevant. |
| **}** |  |  |

**GenerateLink** message response example

|  |
| --- |
| **{**      "result": "00",      "timestamp": "2025-04-03T14:06:35",      "mac": "5a84e41c998e2c6e50f69887b57b51d821d9f6985b2c99db519fb6b7b1473571",      "completeLink": "https://virtualpostest.sia.eu/vpos/payments/main?PAGE=PBL&TOKEN=ovj5w4flcofu5c132gpjk6g92",      "token": "ovj5w4flcofu5c132gpjk6g92",      "creationDate": "2025-04-03T13:58:12.274",      "status": "02",      "expirationDate": "2024-11-30T22:00:00.000",      "orderId": "PBL17",      "linkOptions": null  **}** |

##### **5.9.1.3 Hash calculation**

This section explains how eCommerce needs to generate the hash for submit request from eCommerce web site.

**Hash – SHA 512 Algorithm**

The hash is created by doing the SHA512 algorithm from the given string, then doing base64 encoding. Parameters in the hash string must be separated by “|”.

**plaintext = rnd + | + linkAmount + | + linkOrderId + | + secretKey**

\*Parameters within the hash string must not contain backslash ("\") and pipeline ("|") characters.

\*secretKey will be delivered by the PaySpot system.

**Example:**

plaintext= **BrHklmcUE67kV0oK2oQT|9900.00|PBL17| ZFdYCZ5d8k5**

hash = Base64(SHA512(plaintext))

##### **5.9.1.4 Result codes**

The field **result** can have the following values:

|  |  |
| --- | --- |
| Value | Description |
| 00 | Success |
| 02 | ReqRefNum duplicated or incorrect |
| 03 | Incorrect message format, missing or incorrect field |
| 04 | Incorrect API authentication, incorrect MAC |
| 06 | Unforeseen error during processing of request |
| 13 | Duplicated order number |
| 40 | Empty xml or missing ‘date’ parameter |
| 41 | Xml not parsable |
| 98 | Application error |
| 99 | Transaction failed, see specific outcome attached to the element of the reply. |

#### **5.9.2 List All Links**

**[POST]: api/ECommerce/PayByLink/ListLinks**

**REQUEST PRODUCTION URL:** [**https://www.nsgway.rs:50010/**](https://www.nsgway.rs:50010/api/ECommerce/AuthorizeRequest)**api/ECommerce/PayByLink/ListLinks**

**REQUEST TEST URL:**

**<https://test.nsgway.rs:50009/api/ECommerce/PayByLink/ListLinks>**

**ListLinks** message input parameters

|  |  |  |  |
| --- | --- | --- | --- |
| **Parametar** | **Type** |  | **Description** |
| **{** |  |  |  |
| startDate | String | M | Search start date, yyyy-MM-dd format. AN(10) |
| endDate | String | M | Search end date, yyyy-MM-dd format. AN(10) |
| shopId | String | M | Identifier of the merchant’s shop assigned by Payment Gateway |
| companyID | Int | M | Unambiguous identification of the external system that executes the PaySpot web service call.  \*Will be delivered by the PaySpot system. |
| orderId | String | O | The unique reference of the purchase order for the specified link, generated in the eCommerce or merchant system. You can search by this parameter if you want to AN(40). |
| hash | String | M | Hash calculated as explained in the section 5.9.2.3. **It is mandatory to calculate the hash on the eCommerce server.** |
| rnd | String | M | Random string used for hash calculation |
| } |  |  |  |

##### **5.9.2.1 List Links Request**

**ListLinks** message request example

|  |
| --- |
| **{**    "startDate": "2024-05-27",    "endDate": "2024-11-27",    "shopId": "80729SE00124301",    "companyID": 99019,    "orderID": "PBL3",    "rnd": "string",    "hash": "dTK4AuEc5iMszh8iN64ydr08noKj8+4kVdKw9EHd3FybuyWb7TvbbfMzKuTpApxC5gb3eJ/u4nOBnqArfE9njQ=="  **}** |

##### **5.9.2.2 List Links Response**

**ListLinks** message output parameters

|  |  |  |
| --- | --- | --- |
| **Parametar** | **Type** | **Description** |
| **{** |  |  |
| result | String | Result of transaction. Possible values of this field are listed in the section 5.9.2.4 |
| timestamp | String | Timestamp of the response in format: yyyy-MM-ddThh:mm:ss |
| mac | String | Hash from the payment gateway that is used for checking the integrity of the transaction. |
| linksCreated | Array | Array of created links. |
| **[** |  |  |
| completeLink | String | Generated link that can be sent to the client for payment. |
| token | String | This represents the unique token for this specific link, it is also included in the completeLink field itself. |
| creationDate | String | Timestamp of the link's creation date in format: yyyy-MM-ddThh:mm:ss |
| status | String | Current status of the generated link, it can have the following values: **00** - Created The link has been created correctly (technical status, not visible from the outside) **01** - Returned The link has been communicated to the operator **02** -Sent The link has been sent to the cardholder **03** - Used The link has been used at least once **04** - Paid The payment tied to the link has been completed **05** - Revoked The link has been revoked |
| lastUseDate | String | Timestamp of the last time that the link has been used in format: yyyy-MM-ddThh:mm:ss |
| expirationDate | String | This represents the date until which the created link is active and payable. After this date has passed the link can no longer be paid. Format: yyyy-MM-ddThh:mm:ss |
| revokeDate | String | Timestamp of date when the link was revoked in format: yyyy-MM-ddThh:mm:ss |
| orderId | String | The unique reference of the purchase order for the specified link, generated in the eCommerce or merchant system. Once the link has been paid OrderID is no longer repeatable and can't be used again. AN(40). |
| mac | String | Hash from the payment gateway that is used for checking the integrity of the transaction. |
| linksCreated | Array | Array of created links. |
| **]** |  |  |
| **}** |  |  |

**ListLinks** message response example

|  |
| --- |
| **{**      "result": "00",      "timestamp": "2025-04-03T14:56:07",      "mac": "9124d26fc44a84692fb4dd7ec16a9a4ee67c49a3a39147f3ebbe60532cb96738",      "linksCreated": **[**  **{**              "completeLink": "https://virtualpostest.sia.eu/vpos/payments/main?PAGE=PBL&TOKEN=1gsfg1np3t5qvs14q8tfyv881",              "token": "1gsfg1np3t5qvs14q8tfyv881",              "creationDate": "2024-05-27T13:17:42",              "status": "05",              "lastUseDate": "2024-05-28T22:00:00",              "expirationDate": "2024-05-30T22:00:00",              "revokeDate": "2024-05-27T13:25:43",              "orderId": "PBL3",              "mac": "7f90cbaeb73d13b48f0dc47d1452da3993b567774d22769bc6b61dde9e3c0808"  **}**  **]**  **}** |

##### **5.9.2.3 Hash calculation**

This section explains how eCommerce needs to generate the hash for submit request from eCommerce web site.

**Hash – SHA 512 Algorithm**

The hash is created by doing the SHA512 algorithm from the given string, then doing base64 encoding. Parameters in the hash string must be separated by “|”.

**plaintext = rnd + | + shopId + | + endDate+ | + secretKey**

\*Parameters within the hash string must not contain backslash ("\") and pipeline ("|") characters.

\*secretKey will be delivered by the PaySpot system.

**Example:**

plaintext= **BrHklmcUE67kV0oK2oQT|80729SE00124301|2024-11-27| ZFdYCZ5d8k5**

hash = Base64(SHA512(plaintext))

##### **5.9.2.4 Result codes**

The field **result** can have the following values:

|  |  |
| --- | --- |
| Value | Description |
| 00 | Success |
| 02 | ReqRefNum duplicated or incorrect |
| 03 | Incorrect message format, missing or incorrect field |
| 04 | Incorrect API authentication, incorrect MAC |
| 06 | Unforeseen error during processing of request |
| 40 | Empty xml or missing ‘date’ parameter |
| 41 | Xml not parsable |
| 52 | No link found with the preset search criteria |
| 98 | Application error |
| 99 | Transaction failed, see specific outcome attached to the element of the reply. |

#### **5.9.3 Revoke Link**

**[POST]: api/ECommerce/PayByLink/RevokeLink**

**REQUEST PRODUCTION URL:**[**https://www.nsgway.rs:50010/**](https://www.nsgway.rs:50010/api/ECommerce/AuthorizeRequest)**api/ECommerce/PayByLink/RevokeLink**

**REQUEST TEST URL:**

**<https://test.nsgway.rs:50009/api/ECommerce/PayByLink/RevokeLink>**

**RevokeLink** message input parameters

|  |  |  |  |
| --- | --- | --- | --- |
| **Parametar** | **Type** |  | **Description** |
| **{** |  |  |  |
| token | String | M | This represents the unique token for this specific link that is being revoked. AN(25) |
| shopId | String | M | Identifier of the merchant’s shop assigned by Payment Gateway |
| companyID | Int | M | Unambiguous identification of the external system that executes the PaySpot web service call.  \*Will be delivered by the PaySpot system. |
| hash | String | M | Hash calculated as explained in the section 5.9.3.3. **It is mandatory to calculate the hash on the eCommerce server.** |
| rnd | String | M | Random string used for hash calculation |
| } |  |  |  |

##### **5.9.3.1 Revoke Link Request**

**RevokeLink** message request example

|  |
| --- |
| **{**    "token": "l1s6jfco8mm543ja6xcv1mp81",    "shopId": "80729SE00124301",    "companyID": 99019,    "rnd": "string",    "hash": "nKj9sABDxmgtLHGMSm3E2AM3w/DQ4mV5WnzR7MiNw3bysHijr8o5I5/UKnVzLWdANCSpVRgYfvM5/qKtWdtbPw=="  **}** |

##### **5.9.3.2 Revoke Link Response**

**RevokeLink** message output parameters

|  |  |  |
| --- | --- | --- |
| **Parametar** | **Type** | **Description** |
| **{** |  |  |
| result | String | Result of transaction. Possible values of this field are listed in the section 5.9.3.4 |
| timestamp | String | Timestamp of the response in format: yyyy-MM-ddThh:mm:ss |
| mac | String | Hash from the payment gateway that is used for checking the integrity of the transaction. |
| token | String | This represents the unique token for this specific link that is being revoked. |
| **}** |  |  |

**RevokeLink** message response example

|  |
| --- |
| **{**  "result": "51",      "timestamp": "2025-04-03T15:57:49",      "mac": "259ee5faa596817b57305980fc483724e91c104f3cd88f4008482ccbcfcf7cad",      "token": null  **}** |

##### **5.9.3.3 Hash calculation**

This section explains how eCommerce needs to generate the hash for submit request from eCommerce web site.

**Hash – SHA 512 Algorithm**

The hash is created by doing the SHA512 algorithm from the given string, then doing base64 encoding. Parameters in the hash string must be separated by “|”.

**plaintext = rnd + | + companyId + | + token + | + secretKey**

\*Parameters within the hash string must not contain backslash ("\") and pipeline ("|") characters.

\*secretKey will be delivered by the PaySpot system.

**Example:**

plaintext=**BrHklmcUE67kV0oK2oQT|99019|l1s6jfco8mm543ja6xcv1mp81| ZFdYCZ5d8k5**

hash = Base64(SHA512(plaintext))

##### **5.9.2.4 Result codes**

The field **result** can have the following values:

|  |  |
| --- | --- |
| Value | Description |
| 00 | Success |
| 02 | ReqRefNum duplicated or incorrect |
| 03 | Incorrect message format, missing or incorrect field |
| 04 | Incorrect API authentication, incorrect MAC |
| 06 | Unforeseen error during processing of request |
| 40 | Empty xml or missing ‘date’ parameter |
| 41 | Xml not parsable |
| 50 | Token not found |
| 51 | Status of link not valid (link cannot be revoked) |
| 52 | No link found with the preset search criteria |
| 98 | Application error |
| 99 | Transaction failed, see specific outcome attached to the element of the reply. |

# **6. Integration with PaySpot Payment methods web page from eCommerce web site (UPC)**

### **Form submit**

**[POST]: /api/Ecommerce/submit**

**REQUEST PRODUCTION URL:** [**https://www.nsgway.rs:50010/api/ecommerce/submit**](https://www.nsgway.rs:50010/api/ecommerce/submit)

**REQUEST TEST URL:** [**https://test.nsgway.rs:50009/api/ecommerce/submit**](https://test.nsgway.rs:50009/api/ecommerce/submit)

#### **Form submit input parameters**

**Form submit** input parameters

|  |  |  |  |
| --- | --- | --- | --- |
| **Parametar** | **Type** |  | **Description** |
| companyID | Int | M | Unambiguous identification of the external system that executes the PaySpot web service call.  \*Will be delivered by the PaySpot system. |
| merchantOrderID | String | M | The unique reference of the purchase order, generated in the eCommerce or merchant system. AN(40). |
| merchantOrderAmount | Decimal | M | Total value of the purchase order/purchase for which the payment was initiated. D (15.2)  Contains the total value of goods and/or services + the amount of additional fee that the external partner system may charge for a transaction |
| merchantCurrencyCode | Int | M | Currency code for purchase order. ISO numeric code. The default is the national currency (941 for RSD) |
| language | Int | O | Language ID (1=default (sr), 2=eng ... ) |
| email | String | M | Customer’s e-mail address |
| customerId | String | O | ID which is an alternative to email for CIT transactions, can be used to store user's saved cards in case email is not provided. |
| successURL | String | O | eCommerce has to provide an URL endpoint where the customer browser will be redirected after successful payment |
| cancelURL | String | M | eCommerce has to provide an URL endpoint where the customer browser will be redirected if customer quits purchase before payment |
| errorURL | String | O | eCommerce has to provide an URL endpoint where the customer browser will be redirected in the case of unsuccessful payment |
| hash | String | M | Hash calculated as explained in the section 6.1.2. **It is mandatory to calculate the hash on the eCommerce server.** |
| rnd | String | M | Random string used for hash calculation |
| currentDate | String | M | Date and time of the submit request in the format: YYYY-MM-DD hh:mm:ss (ISO). **Must be sent in UTC timezone,** |
| requestType | Int | M | Request type in PaySpot System. ( Marketplace platform = 10, Web shop platform =11) |
| paymentPurpose | String | O | Payment purpose for IPS payment. AN(35) |
| timeout | Int | O | Redirect User on Cancel URL aftter timeout ( value in seconds ). Default value = 300 |

Html example for integration with PaySpot Payment Methods web page:

      <form

        method="post"

        action='https://test.nsgway.rs:50009/api/ecommerce/submit'

        target="output\_frame">

        <input type="hidden" name="companyId" value=123456 />

        <input type="hidden" name="merchantOrderID" value='Order123' />

        <input type="hidden" name="merchantOrderAmount" value=100.32 />

        <input type="hidden" name="merchantCurrencyCode" value=941 />

        <input type="hidden" name="language" value=1 />

        <input type="hidden" name="successURL" value='https://successUrl.com' />

        <input type="hidden" name="cancelURL" value='https://cancelUrl.com' />

        <input type="hidden" name="errorURL" value='https://failUrl.com' />

        <input type="hidden" name="email" value='test@gmail.com' /> <input type="hidden" name="customerId" value='123456789' />

        <input type="hidden" name="requestType" value=11/>

        <input type="hidden" name="hash" value= '4Oav8Pg9qfHWitVAk/BJOx/cujvdQfNV5tH07VvWPYxuT9nX3NuJAFrZCcdo7D28WeB92TzmB/R3xnFW43KkjQ==' />

        <input type="hidden" name="rnd" value='mIlyKRws6GHHi8mlJtg6' />

<input type="hidden" name="timeout" value='600' />

        <input type="hidden" name="currentDate" value=new Date().toISOString().replace('T', ' ').substring(0, 16) />

      </form>

Optional iframe (if you want to display PaySpot Payment Methods page within iframe)

<iframe name="output\_frame" src="" id="output\_frame" width="100%" height="100%"></iframe>

#### **6.1.2 Hash calculation**

This section explains how eCommerce needs to generate the hash for submit request from eCommerce web site.

**Hash – SHA 512 Algorithm**

The hash is created by doing the SHA512 algorithm from the given string, then doing base64 encoding. Parameters in the hash string must be separated by “|”.

**plaintext = rnd + | + currentDate + | + merchantOrderID + | + merchantOrderAmount + | + secretKey**

\*Parameters within the hash string must not contain backslash ("\") and pipeline ("|") characters.

\*secretKey will be delivered by the PaySpot system.

\*Difference between CurrentDate parameter (**in UTC**) and time when the request is called can't be more than **20min**

**Example:**

plaintext= BrHklmcUE67kV0oK2oQT|2022-10-17 08:38:19|TestGenerate1|100.32|Test123456!

hash = Base64(SHA512(plaintext))

### **Callback URL Data**

ECommerce needs to predefined an URL endpoint where transaction status will be sent using HTTP POST method.

**CallbackURL** message input parameters

|  |  |  |  |
| --- | --- | --- | --- |
| **Parametar** | **Type** |  | **Description** |
| **{** |  |  |  |
| orderID | String | M | Merchant order from initial request |
| shopID | String | O(only for payment cards) | Identifier of the merchant’s shop assigned by Payment Gateway |
| authNumber | String | O(only for payment cards) | Authorization number.  The value is returned only in case of successful authorization. If authorization is not granted,  the field will be filled in with “NULL” |
| amount | String | M | Total value of the purchase order/purchase for which the payment was initiated |
| currency | String | M | Currency |
| transactionID | String | O(only for payment cards) | Identifier of transaction assigned by the PGW |
| result | String | M | Result of the transaction. Described below in section 6.4 |
| transactionDate | String | O(only for payment cards) | Date and time of the transaction. Format:YYMMDDhhmmss |
| paySpotOrderID | String | O(only for IPS) | Unique IPS reference |
| rnd | String | O(only for IPS) | Random string used for hash calculation |
| hash | String | M | Hash calculated as explained in the sections below depending on payment method. **It is mandatory to calculate the hash on the eCommerce server.** |
| maskedPan | String | O(only for payment cards) | Masked Pan of card |
| expiryDate | String | O(only for payment cards) | Expiry date of card |
| cardBrand | String | O(only for payment cards) | Brand of card ( VISA,MASTERCARD,DINA, MAESTRO) |
| panAlias | String | O(only for payment cards) | Token of card |
| responseCode | String | O(only for eMoney) | Represents authorization code for eMoney transaction |
| responseMsg | String | O(only for eMoney) | Represents description of the authorization code for eMoney transaction |
| **}** |  |  |  |

#### **6.2.1 Hash Calculation for Payment by card**

This section explains how eCommerce needs to generate the hash to know that request come from PaySpot Payment Service.

**Hash – SHA 512 Algorithm**

The hash is created by doing the SHA512 algorithm from the given string, then doing base64 encoding. Parameters in the hash string must be separated by “|”.

**plaintext = orderID + | + shopID + | + amount + | + result + | + secretKeyCallback**

\*Parameters within the hash string must not contain backslash ("\") and pipeline ("|") characters.

\*secretKeyCallback will be delivered by the PaySpot system.

**Example:**

plaintext = order1234567890|shopIDTest|100.32|00|Test123456!

hash = Base64(SHA512(plaintext))

#### **6.2.2 Hash Calculation for Payment by IPS**

This section explains how eCommerce needs to generate the hash to know that request come from PaySpot Payment Service.

**Hash – SHA 512 Algorithm**

The hash is created by doing the SHA512 algorithm from the given string, then doing base64 encoding. Parameters in the hash string must be separated by “|”.

**plaintext = rnd + | + paySpotOrderID + | + orderID + | + secretKeyCallback**

\*Parameters within the hash string must not contain backslash ("\") and pipeline ("|") characters.

\*secretKey will be delivered by the PaySpot system.

**Example:**

plaintext= BrHklmcUE67kV0oK2oQT|I000009522290000001|TestGenerate1|Test123456!

hash = Base64(SHA512(plaintext))

### **API Completion Request**

**[POST]: /api/UPC/Completion**

**REQUEST PRODUCTION URL:** [**https://www.nsgway.rs:50010/api/UPC/Completion**](https://www.nsgway.rs:50010/api/UPC/Completion)

**REQUEST TEST URL:**

**<https://test.nsgway.rs:50009/api/UPC/Completion>**

Authorized (Pre-authorized) transactions can be completed using Web API service.  
If the authorization was successful, the values of parameters used for sending Completion request are  
obtained from the previous authorization step.  
If the authorization is not successful, it is impossible to make the  
completion.  
The transaction can be completed to the maximum extent of 30 days from the moment of  
purchase/pre-authorization. Beware that Voiding purchase/pre-authorization transaction is not available after 5 days have passed from the moment of purchase/pre-authorization, which means that every transaction that is not completed in that time period will only have Refund option and Void won't be available.

**AuthCompletionRequest** message input parameters

|  |  |  |  |
| --- | --- | --- | --- |
| **Parametar** | **Type** |  | **Description** |
| **{** |  |  |  |
| transactionId | String | M | Identifier of transaction assigned by the PGW |
| orderId | String | M | Order unique identifier |
| shopId | String | M | Identifier of the merchant’s shop assigned by Payment Gateway |
| terminalId |  | M | POS terminal ID where the payment transaction was made with a payment card |
| amount | String | M | Total value of the purchase order/purchase for which the payment was initiated |
| initialAmount | String | M | Initial amount of the transaction |
| authorizationNumber | String | M | Authorization number received in callback request from PaySpot service |
| transactionDate | String | M | Transaction date in format YYMMDDhhmmss |
| currency | String | M | Currency |
| companyID | String | M | Unambiguous identification of the external system that executes the PaySpot web service call.  \*Will be delivered by the PaySpot system. |
| hash | String | M | Hash calculated as explained in the section 6.3.1. **It is mandatory to calculate the hash on the eCommerce server.** |
| rnd | String | M | Random string used for hash calculation |
| **}** |  |  |  |

**AuthCompletionRequest** message request example

|  |
| --- |
| {    "transactionDate": "250131114651",    "transactionId": "503112252554",    "amount": "99.77",    "orderId": "wdx7gc5c",    "initialAmount": "99.77",    "terminalId": "E7058775",    "authorizationNumber": 461646,    "shopId": "1732150",  "currency": "941",    "companyID": 123456,    "rnd": "string",    "hash": "FCCb6o8R8vZWgmCaC2KSGxhS5got1GF5s40KHyeSJJ4oyGHhaLi3wfCodLZ4+aI3+krRBy6d3UFs3nF/lnH2ZQ=="  } |

#### **6.3.1** **Hash calculation**

This section explains how eCommerce marketplace needs to generate the hash for submit request from eCommerce web site.

**Hash – SHA 512 Algorithm**

The hash is created by doing the SHA512 algorithm from the given string, then doing base64 encoding. Parameters in the hash string must be separated by “|”.

**plaintext = rnd + | + orderId + | + amount + | + secretKey**

\*Parameters within the hash string must not contain backslash ("\") and pipeline ("|") characters.

\*secretKey will be delivered by the PaySpot system.

**Example:**

plaintext= BrHklmcUE67kV0oK2oQT|1zd9k0o9|100.32|Test123456!

hash = Base64(SHA512(plaintext))

#### **6.3.2** **API Completion Response**

**AuthCompletionResponse** message response parameters

|  |  |  |  |
| --- | --- | --- | --- |
| **Parametar** | **Type** |  | **Description** |
| **{** |  |  |  |
| terminalId | String | M | POS terminal ID where the payment transaction was made with a payment card |
| orderId | String | M | Order unique identifier |
| shopId | String | M | Identifier of the merchant’s shop assigned by Payment Gateway |
| amount | Decimal | M | Completion value of the order/purchase for which the payment was initiated |
| currency | String | M | Currency |
| result | String | M | Result of the transaction. Described below in section 6.4 |
| **}** |  |  |  |

**AuthCompletionResponse** message request example

|  |
| --- |
| {      "amount": 99.77,      "orderId": "0huqpxi2",      "terminalId": "E7058775",      "shopId": "1732150",      "companyID": 1,      "result": "000"  } |

### **Response Result Field**

#### **6.4.1 Card Payment**

The field **RESULT** can have the following values:

|  |  |
| --- | --- |
| Value | Description |
| 000 | Success |
| 101 | Invalid card expiration date |
| 105 | The transaction is not allowed by the issuing bank |
| 108 | Lost or stolen card |
| 111 | Non-existent card |
| 116 | Insufficient funds |
| 130 | Exceeded the allowable spending limit |
| 131 | Additional customer authentication required |
| 290 | Issuer is not accessible |
| 291 | Technical or communication problem |
| 401 | Wrong format |
| 402 | Invalid acquirer / seller details |
| 403 | Component communication failure |
| 404 | Authentication error |
| 405 | Signature is invalid |
| 406 | Transaction quota exceeded |
| 407 | Seller is not active |
| 408 | Transaction not found |
| 409 | Too many transactions detected |
| 410 | Order paid (repeat possible) |
| 411 | Order request time outdated |
| 412 | Replay order condition |
| 413 | Unknown card type |
| 414 | CVC required |
| 420 | The total number of successful transactions per day is limited |
| 421 | Limit transaction amount (without full 3-D Secure authentication) |
| 430 | Transaction denied by gateway |
| 431 | Attempt 3D-Secure is not accepted |
| 432 | Card in the stop list |
| 433 | The number of transactions has exceeded the limit |
| 434 | The store does not accept cards from the country |
| 435 | Client IP address in the stop list |
| 436 | The amount of transactions has exceeded the limit |
| 437 | The limit for entering the number of cards has been exceeded |
| 438 | Invalid currency code |
| 439 | Time limit from request to authorization exceeded |
| 440 | Authorization time limit exceeded |
| 441 | MPI interoperability problem |
| 442 | ACS communication problem |
| 450 | Recurring payments are prohibited |
| 451 | MPI service not enabled |
| 452 | Payment from card to card is not included |
| 460 | Token service not included |
| 501 | Canceled by user |
| 502 | The web session has expired |
| 503 | The transaction was canceled by the seller |
| 504 | Transaction was canceled by the cancellation gateway |
| 505 | Incorrect sequence of operations |
| 506 | Pre-authorization has expired |
| 507 | Pre-authorized transaction already processed with payment |
| 508 | Invalid amount to pay for a pre-authorized transaction |
| 509 | Not able to trace back to original transaction |
| 510 | Refund is expired |
| 511 | Transaction canceled by settlement action |
| 512 | Repeated cancellation or refund |
| 601 | Not completed |
| 602 | Waiting for confirmation of payment |
| 701 | Restricted by payment host/schema |
| 902 | Unable to process transaction |
| 904 | Formar error |
| 909 | Can not process transaction |

#### **6.4.2 IPS Payment**

The field **RESULT** can have the following values:

|  |  |
| --- | --- |
| Value | Description |
| 00 | Success |
| -1 | Denied |
| 82 | Timeout |

### **6.5 Get Order Status**

**[POST]: /api/UPC/Status**

**REQUEST PRODUCTION URL:** [**https://www.nsgway.rs:50010/api/UPC/Status**](https://www.nsgway.rs:50010/api/UPC/Status)

**REQUEST TEST URL:**

**<https://test.nsgway.rs:50009/api/UPC/Status>**

The service is provided for merchants who want to check the status of the transaction.

**GetOrderStatus** message input parameters

|  |  |  |  |
| --- | --- | --- | --- |
| **Parametar** | **Type** |  | **Description** |
| **{** |  |  |  |
| orderId | String | M | Order unique identifier |
| shopId | String | M | Identifier of the merchant’s shop assigned by Payment Gateway |
| companyID | String | M | Unambiguous identification of the external system that executes the PaySpot web service call.  \*Will be delivered by the PaySpot system. |
| amount | String | M | Amount of transaction |
| transactionDate | String | M | Transaction date in format YYMMDDhhmmss |
| terminalId | String | M | POS terminal ID where the payment transaction was made with a payment card |
| hash | String | M | Hash calculated as explained in the section 6.5.1. **It is mandatory to calculate the hash on the eCommerce server.** |
| currency | String | M | Currency |
| rnd | String | M | Random string used for hash calculation |
| **}** |  |  |  |

**GetOrderStatus** message request example

|  |
| --- |
| {    "transactionDate": "250205120008",    "orderId": "0huqpxi2",    "amount": "99.77",  "currency": "941",    "terminalId": "E7058775",    "shopId": "1732150",    "companyID": 12345,    "rnd": "string",    "hash": "WfEf4p7S2jjp5kcEX5AZdNyBAZQ5NxTKVtUyzKRDEImK0LYt66y/ZilszlbTTkUZ3Ef+0SQZ7IMGRw6v2qob9g=="  } |

#### **6.5.1** **Hash calculation**

This section explains how eCommerce marketplace needs to generate the hash for submit request from eCommerce web site.

**Hash – SHA 512 Algorithm**

The hash is created by doing the SHA512 algorithm from the given string, then doing base64 encoding. Parameters in the hash string must be separated by “|”.

**plaintext = rnd + | + orderId + | + amount + | + secretKey**

\*Parameters within the hash string must not contain backslash ("\") and pipeline ("|") characters.

\*secretKey will be delivered by the PaySpot system.

**Example:**

plaintext= BrHklmcUE67kV0oK2oQT|1zd9k0o9|100.32|Test123456!

hash = Base64(SHA512(plaintext))

#### **6.5.2** **Get Order Status response**

**GetOrderStatus** message output parameters

|  |  |  |
| --- | --- | --- |
| **Parametar** | **Type** | **Description** |
| **{** |  |  |
| transactionId | String | Identifier of transaction assigned by the PGW |
| orderId | String | Merchant order unique identifier |
| shopId | String | Identifier of the merchant’s shop assigned by Payment Gateway |
| transactionAmount | String | The transaction amount in currency unit |
| maskedPan | String | Represents a masked version of full card PAN, it includes first 6 and last 4 digits of the card PAN |
| transactionResult | String | Outcome of transaction. Described below in section 6.5.3.2 |
| transactionStatus | String | Current status of the transaction . Described below in section 6.5.3.1 |
| authorizationNumber | String | Authorization number.  The value is returned only in case of successful authorization. If authorization is not granted,  the field will be filled in with “NULL” |
| transactionDate | String | Date and time of transaction in yyyy-mm-gg HH:mm:ss.mss format |
| currency | String | Currency |
| **}** |  |  |

**GetOrderStatus**message response example

|  |
| --- |
| {      "shopId": "1732150",      "terminalId": "E7058775",      "authorizationNumber": "555500",      "transactionId": "503613260328",      "maskedPan": "499999\*\*\*\*\*\*0011",      "transactionDate": "2025-02-05 12:10:09.631",      "transactionAmount": "99.77",      "currency": "941",      "transactionStatus": "04",      "transactionResult": "000"  } |

#### **6.5.3.1 Transaction Status codes**

The field **transactionStatus** can have the following values:

|  |  |
| --- | --- |
| Value | Description |
| 00 | PREAUTHORIZATION |
| 02 | POSTAUTHORIZATION |
| 03 | PURCHASE |
| 04 | REVERSAL |

#### **6.5.3.2 Transaction Result codes**

The field **transactionResult** can have the following values:

|  |  |
| --- | --- |
| Value | Description |
| 000 | Success |
| 101 | Invalid card expiration date |
| 105 | The transaction is not allowed by the issuing bank |
| 108 | Lost or stolen card |
| 111 | Non-existent card |
| 116 | Insufficient funds |
| 130 | Exceeded the allowable spending limit |
| 131 | Additional customer authentication required |
| 290 | Issuer is not accessible |
| 291 | Technical or communication problem |
| 401 | Wrong format |
| 402 | Invalid acquirer / seller details |
| 403 | Component communication failure |
| 404 | Authentication error |
| 405 | Signature is invalid |
| 406 | Transaction quota exceeded |
| 407 | Seller is not active |
| 408 | Transaction not found |
| 409 | Too many transactions detected |
| 410 | Order paid (repeat possible) |
| 411 | Order request time outdated |
| 412 | Replay order condition |
| 413 | Unknown card type |
| 414 | CVC required |
| 420 | The total number of successful transactions per day is limited |
| 421 | Limit transaction amount (without full 3-D Secure authentication) |
| 430 | Transaction denied by gateway |
| 431 | Attempt 3D-Secure is not accepted |
| 432 | Card in the stop list |
| 433 | The number of transactions has exceeded the limit |
| 434 | The store does not accept cards from the country |
| 435 | Client IP address in the stop list |
| 436 | The amount of transactions has exceeded the limit |
| 437 | The limit for entering the number of cards has been exceeded |
| 438 | Invalid currency code |
| 439 | Time limit from request to authorization exceeded |
| 440 | Authorization time limit exceeded |
| 441 | MPI interoperability problem |
| 442 | ACS communication problem |
| 450 | Recurring payments are prohibited |
| 451 | MPI service not enabled |
| 452 | Payment from card to card is not included |
| 460 | Token service not included |
| 501 | Canceled by user |
| 502 | The web session has expired |
| 503 | The transaction was canceled by the seller |
| 504 | Transaction was canceled by the cancellation gateway |
| 505 | Incorrect sequence of operations |
| 506 | Pre-authorization has expired |
| 507 | Pre-authorized transaction already processed with payment |
| 508 | Invalid amount to pay for a pre-authorized transaction |
| 509 | Not able to trace back to original transaction |
| 510 | Refund is expired |
| 511 | Transaction canceled by settlement action |
| 512 | Repeated cancellation or refund |
| 601 | Not completed |
| 602 | Waiting for confirmation of payment |
| 701 | Restricted by payment host/schema |
| 902 | Unable to process transaction |
| 904 | Formar error |
| 909 | Can not process transaction |

### **6.6 Refund / Void Transaction**

**[POST]: /api/UPC/Refund**

**REQUEST PRODUCTION URL:** [**https://www.nsgway.rs:50010/api/UPC/Refund**](https://www.nsgway.rs:50010/api/UPC/Refund)

**REQUEST TEST URL:**

**<https://test.nsgway.rs:50009/api/UPC/Refund>**

The service is provided for merchants who want to void the transaction before the transaction was accounted (before financial transaction happened in bank). This window for Auth transactions is day of the transaction until midnight, while for PreAuthorized transaction the window for voiding the transaction is 5 days from the day since transaction happened. Transactions can be voided only for the full amount, there is no partial void therefore Amount field plays no role in Void transactions only in Refund. All transactions sent to this endpoint that can't be processed as Void will be saved in our system and processed as Refund transactions which are done through our Call Center. These Refund transactions can be partial and time window is declared by the issuing bank.

**RefundTransaction** message input parameters

|  |  |  |  |
| --- | --- | --- | --- |
| **Parametar** | **Type** |  | **Description** |
| **{** |  |  |  |
| transactionId | String | M | Identifier of transaction assigned by the PGW |
| orderId | String | M | Order unique identifier |
| shopId | String | M | Identifier of the merchant’s shop assigned by Payment Gateway |
| terminalId | String | M | POS terminal ID where the payment transaction was made with a payment card |
| amount | String | M | Amount that is going to be refunded (in case of Void it will always be full amount of the transaction as there is no partial Void transaction). |
| initialAmount | String | M | Initial amount of the transaction |
| authorizationNumber | String | M | Authorization number. |
| transactionDate | String | M | Date and time of transaction in YYMMDDhhmmss format |
| currency | String | M | Currency |
| companyID | String | M | Unambiguous identification of the external system that executes the PaySpot web service call.  \*Will be delivered by the PaySpot system. |
| hash | String | M | Hash calculated as explained in the section 6.3.1. **It is mandatory to calculate the hash on the eCommerce server.** |
| rnd | String | M | Random string used for hash calculation |
| **}** |  |  |  |

**RefundTransaction** message request example

|  |
| --- |
| {    "transactionDate": "250205120008",    "transactionId": "503613260328",    "amount": "99.77",    "orderId": "0huqpxi2",    "initialAmount": "99.77",  "currency": "941",    "terminalId": "E7058775",    "authorizationNumber": 906535,  "shopId": "1732150",    "companyID": 12345,    "rnd": "string",    "hash": "WfEf4p7S2jjp5kcEX5AZdNyBAZQ5NxTKVtUyzKRDEImK0LYt66y/ZilszlbTTkUZ3Ef+0SQZ7IMGRw6v2qob9g=="  } |

#### **6.6.1** **Hash calculation**

This section explains how eCommerce marketplace needs to generate the hash for submit request from eCommerce web site.

**Hash – SHA 512 Algorithm**

The hash is created by doing the SHA512 algorithm from the given string, then doing base64 encoding. Parameters in the hash string must be separated by “|”.

**plaintext = rnd + | + orderId + | + amount + | + secretKey**

\*Parameters within the hash string must not contain backslash ("\") and pipeline ("|") characters.

\*secretKey will be delivered by the PaySpot system.

**Example:**

plaintext= BrHklmcUE67kV0oK2oQT|1zd9k0o9|100.32|Test123456!

hash = Base64(SHA512(plaintext))

#### **6.6.2** **Void / Refund Transaction Response**

**RefundTransaction** message output parameters

|  |  |  |
| --- | --- | --- |
| **Parametar** | **Type** | **Description** |
| **{** |  |  |
| companyID | String | Unambiguous identification of the external system that creates the PaySpot web service call |
| orderId | String | Merchant order unique identifier |
| shopId | String | Identifier of the merchant’s shop assigned by Payment Gateway |
| terminalId | String | POS terminal ID where the payment transaction was made with a payment card |
| amount | Decimal | The transaction amount in currency unit |
| result | String | Outcome of transaction. Described below in section 4.3.3.2 |
| currency | String | Currency |
| **}** |  |  |

**RefundTransaction**message response example

|  |
| --- |
| {      "amount": 99.77,      "orderId": "0huqpxi2",      "terminalId": "E7058775",      "shopId": "1732150",      "companyID": 1,      "result": "000",      "currency": "941"  } |

### **6.7 Client Delete Saved Card Data**

**[POST]: /api/ECommerce/ClientDeleteSavedCardData**

**REQUEST PRODUCTION URL:** [**https://www.nsgway.rs:50010/api/ECommerce/ClientDeleteSavedCardData**](https://www.nsgway.rs:50010/api/ECommerce/ClientDeleteSavedCardData)

**REQUEST TEST URL:**

**<https://test.nsgway.rs:50009/api/ECommerce/ClientDeleteSavedCardData>**

The service is provided for merchants who are using our CIT (Client Initiated Transaction) flow which includes saving the card token (tokenization) on the client side for further payments. This service deletes the tokenized data stored on our servers and removes the card from our frontend meaning that user can't use it for payments anymore.

**ClientDeleteSavedCardData** message input parameters

|  |  |  |  |
| --- | --- | --- | --- |
| **Parametar** | **Type** |  | **Description** |
| **{** |  |  |  |
| panAlias | String | M | Represents a card token |
| maskedPan | String | M | Represents a masked version of full card PAN, it includes first 6 and last 4 digits of the card PAN |
| shopId | String | M | Identifier of the merchant’s shop assigned by Payment Gateway |
| companyID | Int | M | Unambiguous identification of the external system that executes the PaySpot web service call.  \*Will be delivered by the PaySpot system. |
| customerId | String | O | ID which is an alternative to email for CIT transactions, can be used to store user's saved cards in case email is not provided. |
| customerEmail | String | O | User email which is used to store tokens, either customerId or customerEmail must be sent |
| amount | String | M | Amount that is going to be refunded (in case of Void it will always be full amount of the transaction as there is no partial Void transaction). |
| hash | String | M | Hash calculated as explained in the section 5.5.1. **It is mandatory to calculate the hash on the eCommerce server.** |
| rnd | String | M | Random string used for hash calculation |
| **}** |  |  |  |

**ClientDeleteSavedCardData** message request example

|  |
| --- |
| {    "PanAlias": "1234567890",  "maskedPan": "534223xxxxxx1234",    "shopId": "testShopID",    "companyID": 123456,  "clientId": "123456",    "rnd": "mIlyKRws6GHHi8mlJtg6",    "hash": "4Oav8Pg9qfHWitVAk/BJOx/cujvdQfNV5tH07VvWPYxuT9nX3NuJAFrZCcdo7D28WeB92TzmB/R3xnFW43KkjQ=="  } |

#### **6.7.1** **Hash calculation**

This section explains how eCommerce marketplace needs to generate the hash for submit request from eCommerce web site.

**Hash – SHA 512 Algorithm**

The hash is created by doing the SHA512 algorithm from the given string, then doing base64 encoding. Parameters in the hash string must be separated by “|”.

**plaintext = rnd + | + panAlias + | + secretKey**

\*Parameters within the hash string must not contain backslash ("\") and pipeline ("|") characters.

\*secretKey will be delivered by the PaySpot system.

**Example:**

plaintext= mIlyKRws6GHHi8mlJtg6|01231231231|Test123456!

hash = Base64(SHA512(plaintext))

#### **6.7.2** **Client Delete Saved Card Data Response**

**ClientDeleteSavedCardData** message output parameters

|  |  |  |
| --- | --- | --- |
| **Parametar** | **Type** | **Description** |
| **{** |  |  |
| panAlias | String | Represents a card token that was deleted |
| maskedPan | String | Represents a masked version of full card PAN, it includes first 6 and last 4 digits of the card PAN that was deleted |
| shopId | String | Identifier of the merchant’s shop assigned by Payment Gateway |
| companyId | String | Unambiguous identification of the external system that executes the PaySpot web service call.  \*Will be delivered by the PaySpot system. |
| customerId | String | ID which is an alternative to email for CIT transactions, can be used to store user's saved cards in case email is not provided. |
| customerEmail | String | User email which is used to store tokens, either customerId or customerEmail must be sent |
| errorCode | Int | Error code |
| errorMessage | String | Description of the Error code |
| statusAlias | String | The current status of the saved token |
| **savedCards [ {** |  | Array of saved cards |
| panAlias | String | Represents a card token |
| maskedPan | String | Represents a masked version of full card PAN, it includes first 6 and last 4 digits of the card PAN |
| panExpiryDate | String | Date of expiration of the tokenized card in format YYMM |
| shopId | String | Identifier of the merchant’s shop assigned by Payment Gateway |
| treCurr | String | Type of recurring transaction, **C**-Client Initiated Transaction, **U**-Unscheduled Merchant initiated recurring transaction and **R**-Recurring Merchant initiated transaction |
| creCurr | String | Identifier that connects initial recurring tansaction with following recurring transactions |
| **] }** |  |  |
| hash | String | Hash that is used to verify that the response is actually comming from our system, it's calculated according to specification in section 5.8.1 |

**ClientDeleteSavedCardData** message response example

|  |
| --- |
| {    "panAlias": "1234567890",    "maskedPan": "534223xxxxxx1234",  "companyId": 123456,  "customerId": "7890123",  "customerEmail": "test@gmail.com",    "shopId": "testshopID",    "errorCode": 0,    "errorMessage": "OK",    "statusAlias": 0,    "savedCards": [          {              "panAlias": "0000487552113990718",              "maskedPan": "534223xxxxxx1234",              "panExpiryDate": "2801",              "shopID": "80729SE00124302",              "treCurr": "C",              "creCurr": "COF0123456789"          },          {              "panAlias": "0000700670416298520",              "maskedPan": "534223xxxxxx1234",              "panExpiryDate": "3512",              "shopID": "80729SE00124302",              "treCurr": "C",              "creCurr": "COF0123456789"          }      ],  "hash": "aVdkL3zechCDz5FJJZKaP9cYTk7jVsLUWEdVBmw71zKbIBGM9pOsAGk1htg=="  } |

#### **6.7.3** **Client Delete Saved Card Callback URL**

ECommerce needs to predefined an URL endpoint where deleted credit card info will be sent using HTTP POST method.

**Client Delete Saved Card Callback** message input parameters

|  |  |  |  |
| --- | --- | --- | --- |
| **Parametar** | **Type** |  | **Description** |
| **{** |  |  |  |
| panAlias | String | M | Token of card |
| maskedPan | String | M | Masked PAN number of the card which represents first 6 and the last 4 digits of the card's PAN |
| companyID | String | M | Customer identifier that is used to link the PAN and Card to specific user |
| shopID | String | M | Identifier of the merchant’s shop assigned by Payment Gateway |
| customerId | String | O | ID which is an alternative to email for CIT transactions, can be used to store user's saved cards in case email is not provided. |
| customerEmail | String | O | User email which is used to store tokens, either customerId or customerEmail must be sent |
| errorCode | Int | M | Error code |
| errorMessage | String | M | Description of the Error code |
| statusAlias | String | M | The current status of the saved token |
| **savedCards [ {** |  | M | Array of saved cards |
| panAlias | String | M | Token of card |
| maskedPan | String | M | Represents a masked version of full card PAN, it includes first 6 and last 4 digits of the card PAN |
| panExpiryDate | String | M | Date of expiration of the tokenized card in format YYMM |
| shopId | String | M | Identifier of the merchant’s shop assigned by Payment Gateway for this specific token |
| treCurr | String | M | Type of recurring transaction, **C**-Client Initiated Transaction, **U**-Unscheduled Merchant initiated recurring transaction and **R**-Recurring Merchant initiated transaction |
| creCurr | String | M | Identifier that connects initial recurring transaction with following recurring transactions |
| **] }** |  |  |  |
| hash | String | M | Hash calculated as explained in the section 5.8.1. **It is mandatory to calculate the hash on the eCommerce server.** |

##### **6.7.3.1 Hash Calculation for Client Delete saved Card Callback**

This section explains how eCommerce needs to generate the hash to know that request come from PaySpot Payment Service.

**Hash – SHA 512 Algorithm**

The hash is created by doing the SHA512 algorithm from the given string, then doing base64 encoding. Parameters in the hash string must be separated by “|”.

**plaintext = shopID + | + panAlias + | + maskedPan + | + statusAlias + | + secretKeyCallback**

\*Parameters within the hash string must not contain backslash ("\") and pipeline ("|") characters.

\*secretKeyCallback will be delivered by the PaySpot system.

**Example:**

plaintext = shopIDTest|0000624648507414609|534223xxxxxx1234|0|Test123456!

hash = Base64(SHA512(plaintext))

### **6.8 Merchant Initiate Transactions (MIT)**

#### **6.8.1 Registration card Request**

**[POST]: /api/ECommerce/GetPaymentParams**

**REQUEST PRODUCTION URL:** [**https://www.nsgway.rs:50010/api/UPC/GetPaymentParams**](https://www.nsgway.rs:50010/api/UPC/GetPaymentParams)

**REQUEST TEST URL:**

**<https://test.nsgway.rs:50009/api/UPC/GetPaymentParams>**

**RegistrationCard** message input parameters

|  |  |  |  |
| --- | --- | --- | --- |
| **Parametar** | **Type** |  | **Description** |
| **{** |  |  |  |
| merchantOrderID | String | M | The unique reference of the purchase order, generated in the eCommerce or merchant system. AN(40). |
| merchantOrderAmount | Decimal | M | Total value of the purchase order/purchase for which the payment was initiated. D (15.2)  Contains the total value of goods and/or services + the amount of additional fee that the external partner system may charge for a transaction |
| merchantCurrencyCode | Int | M | Currency code for purchase order. ISO numeric code. The default is the national currency (941 for RSD) |
| language | Int | O | Language ID (1=default (sr), 2=eng ... ) |
| customerEmail | String | M | Customer’s e-mail address |
| customerId | String | O | ID which is an alternative to email for CIT/MIT transactions, can be used to store user's saved cards in case email is not provided. |
| panAlias | String | O | Merchant token for payment card |
| successURL | String | M | eCommerce has to provide an URL endpoint where the customer browser will be redirected after successful payment |
| cancelURL | String | M | eCommerce has to provide an URL endpoint where the customer browser will be redirected if customer quits purchase before payment |
| errorURL | String | M | eCommerce has to provide an URL endpoint where the customer browser will be redirected in the case of unsuccessful payment |
| requestType | Int | M | Request type in PaySpot System. ( Marketplace platform = 10, Web shop platform =11) |
| companyID | Int | M | Unambiguous identification of the external system that executes the PaySpot web service call.  \*Will be delivered by the PaySpot system. |
| hash | String | M | Hash calculated as explained in the section 5.8.1.3. **It is mandatory to calculate the hash on the eCommerce server.** |
| rnd | String | M | Random string used for hash calculation |
| currentDate | String | M | Date and time of the submit request in the format: YYYY-MM-DD hh:mm:ss (ISO). **Must be sent in UTC timezone,** |
| transactionType | String | M | Type of transaction .  -REDIRECT\_MIT |
| } |  |  |  |

##### **6.8.1.1 Registration Card Request**

**RegistrationCardRequest** message request example

|  |
| --- |
| {  "merchantOrderID":"testoid",           "merchantOrderAmount":1.00,           "merchantCurrencyCode":941,           "language":1,           "customerId":"123456",           "customerEmail":"test@gmail.com",           "successURL":"https://test.com",           "cancelURL":"https://test.com",           "errorUrl":"https://test.com",           "panAlias":"1234567890",           "requestType":10,           "companyID":123456,           "hash":"hashstring",           "transactionType":"REDIRECT\_MIT",           "rnd":"randomstring",           "currentDate":"2024-02-02 08:38:19"        } |

##### **6.8.1.2 RegistrationCard Response**

**RegistrationCardResponse** message response parameters

|  |  |  |  |
| --- | --- | --- | --- |
| **Parametar** | **Type** |  | **Description** |
| **{** |  |  |  |
| action | String | M | Url to call PGW |
| orderID | String | M | Order unique identifier |
| version | String | M | Version, always 1 |
| merchantID | String | M | Identifier of the merchant’s shop assigned by Payment Gateway |
| terminalID | String | M | POS terminal ID where the payment transaction will be proceed with a payment card |
| totalAmount | String | M | Total value of the purchase order/purchase for which the payment was initiated |
| currency | String | M | Currency |
| locale | String | M | Language of payment page |
| purchaseTime | String | M | Transaction date in format YYMMDDhhmmss |
| signature | String | M | Hash string |
| **}** |  |  |  |

**RegistrationCardResponse** message request example

|  |
| --- |
| {      "action": "https://ecg.test.upc.ua/rbrs/enter",      "Version": "1",      "MerchantID": "1732150",      "TerminalID": "E7058775",      "PurchaseTime": "250401120824",      "OrderID": "testoid",      "Currency": "941",      "TotalAmount": "100",      "Locale": "SR",      "Signature": "bOWjSaXRuOZ8rUXv6F133c+orqAKT76jGPAjFEPhukwPKa4w9gNuUbDL7jtbr7wqeTS1UjFENTBnBksIPJKMsLvCw4mhH0PAj/1adQlVEii56pEEZLeADBOsRLqGsPjl2ezmlfLpjEdHxzCvlZ0aw1MKzLKFzdzQa5AKSCqXSXA=",      "Recurrent": "true",      "errorCode": 0,      "errorMessage": "OK"  } |

##### **6.8.1.3 Hash calculation**

This section explains how eCommerce needs to generate the hash for submit request from eCommerce web site.

**Hash – SHA 512 Algorithm**

The hash is created by doing the SHA512 algorithm from the given string, then doing base64 encoding. Parameters in the hash string must be separated by “|”.

**plaintext = rnd + | + currentDate + | + merchantOrderID + | + merchantOrderAmount + | + secretKey**

\*Parameters within the hash string must not contain backslash ("\") and pipeline ("|") characters.

\*secretKey will be delivered by the PaySpot system.

**Example:**

plaintext= BrHklmcUE67kV0oK2oQT|2022-10-17 08:38:19|TestGenerate1|100.32|Test123456!

hash = Base64(SHA512(plaintext))

##### **6.8.1.4 Form Example**

  <form action="https://ecg.test.upc.ua/rbrs/enter" method="post">

    <input name="Version" type="hidden" value="1">

    <input name="MerchantID" type="hidden" value="1732150">

    <input name="TerminalID" type="hidden" value="E7058775">

    <input name="TotalAmount" type="hidden" value="132">

    <input name="Currency" type="hidden" value="941">

    <input name="Locale" type="hidden" value="SR">

    <input name="PurchaseTime" type="hidden" value="241129150000">

    <input name="OrderID" type="hidden" value="123456">

    <input name="Signature" type="hidden" value="zahCwW0epD0UJJz26zEc82v5MwWP4xXAhRbSKv7R//c2wjm6wPfq/DHUPLlwbw8bwI2S6Jf9jM82bDs1R2mHTvYEDfRv7DnB1jZPCxdHbFaWqJwbG93729ytUxOCyMKLxD/iqHIV3A1amR/mPpIgUStfqUD9x2syrsx9LWBEU5Q=">

    <button class="btn btn\_\_primary green" type="submit">

      <p>Register card </p>

    </button>

  </form>

#### **6.8.2 Authorize Request**

**[POST]: /api/ECommerce/AuthorizeRequest**

**REQUEST PRODUCTION URL:** [**https://www.nsgway.rs:50010/api/UPC/AuthorizeRequest**](https://www.nsgway.rs:50010/api/UPC/AuthorizeRequest)

**REQUEST TEST URL:**

**<https://test.nsgway.rs:50009/api/UPC/AuthorizeRequest>**

**AuthorizeRequest** message input parameters

|  |  |  |  |
| --- | --- | --- | --- |
| **Parametar** | **Type** |  | **Description** |
| **{** |  |  |  |
| orderId | String | M | The unique reference of the purchase order, generated in the eCommerce or merchant system. AN(40). |
| amount | Decimal | M | Total value of the purchase order/purchase for which the payment was initiated. D (15.2)  Contains the total value of goods and/or services + the amount of additional fee that the external partner system may charge for a transaction |
| currency | String | M | Currency code for purchase order. ISO numeric code. The default is the national currency (941 for RSD) |
| customerEmail | String | M | Customer’s e-mail address |
| customerId | String | O | ID which is an alternative to email for CIT/MIT transactions, can be used to store user's saved cards in case email is not provided. |
| panAlias | String | O | Merchant token for payment card |
| shopId | String | M | Identifier of the merchant’s shop assigned by Payment Gateway |
| transactionType | String | M | Type of transaction ( REDIRECT/REDIRECT\_MIT/CIT/MIT\_UNSC ) |
| companyID | String | M | Unambiguous identification of the external system that executes the PaySpot web service call.  \*Will be delivered by the PaySpot system. |
| requestType | Int | M | Request type in PaySpot System. ( Marketplace platform = 10, Web shop platform =11) |
| hash | String | M | Hash calculated as explained in the section 5.8.2.6. **It is mandatory to calculate the hash on the eCommerce server.** |
| rnd | String | M | Random string used for hash calculation |
| currentDate | String | M | Date and time of the submit request in the format: YYYY-MM-DD hh:mm:ss (ISO). **Must be sent in UTC timezone,** |
| } |  |  |  |

##### **6.8.2.1 Authorize Request**

**AuthorizeRequest** message request example

|  |
| --- |
| {    "orderId": "Test123",    "amount": 123.25,    "currency": "941",    "customerId": "1234567",    "customerEmail": "test@gmail.com",    "panAlias": "D0AF93D6C03EF3E364620F41CC7109C4",    "shopId": "ShopIdTest",    "companyID": 1,    "rnd": "string",    "hash": "BtvO8kFlu2cVaYWyqbeRtLf8Tzt5VfsqKAZHC6p4EMJRhvYAxpdK+HC0ZZR6E//xJ982el480wzsCCNa8jyYpQ==",    "transactionType": "MIT\_UNSC",    "requestType": 11,    "currentDate": "2025-03-01 13:17:19"  } |

##### **6.8.2.2 Authorize response**

**AuthorizeResponse** message output parameters

|  |  |  |
| --- | --- | --- |
| **Parametar** | **Type** | **Description** |
| **{** |  |  |
| transactionId | String | Identifier of transaction assigned by the PGW |
| orderId | String | Merchant order unique identifier |
| shopId | String | Identifier of the merchant’s shop assigned by Payment Gateway |
| transactionAmount | String | The transaction amount in currency unit |
| transactionResult | String | Outcome of transaction. |
| transactionResultDescription | String | Description of transaction result |
| authorizationNumber | String | Authorization number.  The value is returned only in case of successful authorization. If authorization is not granted,  the field will be filled in with “NULL” |
| transactionDate | String | Date and time of transaction in yyyy-mm-ggTHH:mm:ss format |
| currency | String | Currency |
| **}** |  |  |

**AuthorizeResponse** message response example

|  |
| --- |
| {              "transactionId": "509112413280",              "shopId": "1732150",              "orderId": "Test011211",              "transactionAmount": "123.25",              "currency": "941",              "authorizationNumber": "075621",              "transactionDate": "2025-04-01T11:20:45",              "transactionResult": "000",              "transactionResultDescription": "Approved"  } |

##### **6.8.2.3 Transaction Status codes**

The field **transactionStatus** can have the following values:

|  |  |
| --- | --- |
| Value | Description |
| 00 | Preauthorization |
| 01 | Preauthorization denied |
| 02 | Authorization to be processed |
| 03 | Authorization processed by clearing |
| 04 | Refund |
| 21 | Authorization to be reversed due to transaction error |
| 99 | Authorization underway with MyBank or  BancomatPay |

##### **6.8.2.4 Transaction Result codes**

The field **transactionResult** can have the following values:

|  |  |
| --- | --- |
| Value | Description |
| 00 | Success |
| 01 | Denied due to problems in the request message |
| 02 | Denied due to problems in the store registry |
| 03 | Denied due to communication problems with the authorization circuits |
| 04 | Denied by card issuer |
| 05 | Denied due to incorrect card number |
| 06 | Unforeseen error during processing of request |
| 10 | Card not eligible for Installments |
| 45 | Denied authorization due to failed antifraud check. |
| 51 | Installment number out of bounds (acquirer side) |
| 99 | Authorization underway with MyBank or BancomatPay |

##### **6.8.2.5 Result codes**

The field **result** can have the following values:

|  |  |
| --- | --- |
| Value | Description |
| 00 | Success |
| 01 | Order or ReqRefNum not found |
| 02 | ReqRefNum duplicated or not valid |
| 03 | Incorrect message format, missing or incorrect field |
| 04 | Incorrect API authentication, incorrect MAC or timestamp exceeding the limit range |
| 05 | Incorrect date, or period indicated is empty |
| 06 | Unforeseen error during processing of request |
| 07 | TransactionID not found |
| 08 | Operator indicated not found |
| 09 | TRANSACTIONID indicated does not make reference to the entered ORDERID |
| 10 | Amount indicated exceeds maximum amount permitted |
| 11 | Incorrect status. Transaction not possible in the current status |
| 12 | Circuit disabled |
| 13 | Duplicated order |
| 16 | Currency not supported or not available for the merchant |
| 17 | Exponent not supported for the chosen currency |
| 20 | The card is VBV/SecureCode/SafeKey-enabled; the reply contains the data for redirection to ACS website |
| 21 | Maximum time-limit for forwarding VBV request step 2 expired |
| 25 | A call to 3DS method must be performed by the Requestor |
| 26 | A challenge flow must be initiated by the Requestor |
| 35 | No payment instrument is acceptable |
| 37 | Missing CVV2: this is compulsory for the circuit selected |
| 38 | Pan alias not found or revoked 40 Empty Xml or missing ‘data’ parameter |
| 41 | Xml not parsable |
| 50 | Installments not available |
| 51 | Installment number out of bounds (client side) |
| 98 | Application error |
| 99 | Transaction failed, see specific outcome attached to the element of the reply. |

##### **6.8.2.6 Hash calculation**

This section explains how eCommerce needs to generate the hash for submit request from eCommerce web site.

**Hash – SHA 512 Algorithm**

The hash is created by doing the SHA512 algorithm from the given string, then doing base64 encoding. Parameters in the hash string must be separated by “|”.

**plaintext = rnd + | + currentDate + | + orderId + | + amount + | + secretKey**

\*Parameters within the hash string must not contain backslash ("\") and pipeline ("|") characters.

\*secretKey will be delivered by the PaySpot system.

**Example:**

plaintext= BrHklmcUE67kV0oK2oQT|2022-10-17 08:38:19|TestGenerate1|100.32|Test123456!

hash = Base64(SHA512(plaintext))

# **Card payments**

### **Card payments messages**

### **Marketplace card transaction (msgType 62)**

**[POST]: /api/marketplacetransactions**

**REQUEST URL PRODUCTION:**[**https://www.nsgway.rs:50010/api/marketplacetransactions**](https://www.nsgway.rs:50010/api/marketplacetransactions%20)**REQUEST URL TEST:** [**https://test.nsgway.rs:50009/api/marketplacetransactions**](https://test.nsgway.rs:50009/api/marketplacetransactions)

Message type ***MsgType=62 MarketplaceTransactions*** is used to send data related to card transaction status.

#### **MarketplaceTransactions input parameters**

**MarketplaceTransactions**message input parameters

|  |  |  |  |
| --- | --- | --- | --- |
| **Parameter** | **Type** |  | **Description** |
| gatewayOrderId | String | M | Card transaction unique ID in the payment gateway system. Transaction ID returned by PGW should be sent in this field. Some gateways named that ID as OrderId or TransId. If you are not sure what to send in this field please consult PaySpot experts. |
| STAN | String | O | Alphanumeric field that indicates the sequence of transaction. |
| approvalCode | String | M | Transaction approval/authorization code. |
| shopID | String | M | Represents a unique identification of web shop assigned to marketplace either by PGW or PaySpot. |
| merchantOrderID | String | M | Unique identifier of the order received from merchant |
| amount | Decimal | M | Charging amount.  **Note**: for the refund the amount should be the amount which is due to be held by the cardholder. Example: if the original amount was 1000, and 100 should be refunded, the amount in the refund message should be 900 |
| currencyCode | Integer | M | Transaction currency represented in ISO 4217 numeric code. |
| actionSuccess | String | M | ActionSuccess parameter represents status of this action. In the event of successful action ActionSuccess value will be 1, and if the action was unsuccessful the parameter will be 0. |
| authorized | String | M | Authorized parameter can have two possible values: 1 or 0. 1 means that transaction was authorized, 0 means that transaction was not authorized. |
| completed | String | M | Completed parameter can have two possible values: 1 or 0. 1 means that transaction is completed, 0 means that transaction is incomplete. |
| voided | String | M | Voided parameter can have two possible values: 1 or 0. 1 means that transaction is voided, 0 means that transaction is not voided. |
| refunded | String | M | Refunded parameter can have two possible values: 1 or 0. 1 means that transaction is refunded, 0 means that transaction is not refunded. |
| creditCardName | String | M | Name of card brand that was be used for this transaction. |
| creditCardNumber | String | M | Masked number of credit card used for this transaction. |
| transactionDateTime | String | M | Date and time of transaction.  For Completion, Void and Refund the value has to be the date and time when the corresponding action is performed, not the date and time of the original transaction (Authorization).Format:  YYYY-MM-DD hh:mm:ss |
| Partner | String | O | Represents a partner who has processed the transaction. |
| ECI | String | O | Represents result of 3D Secure authentication (Electronic Commerce Indicator). |
| expirationDate | String | O | Expiration date of credit card used for this transaction. (YYMM format). |

**MarketplaceTransactions** message request example

|  |
| --- |
| {     "data": {        "header": {           "companyID": 123456,           "requestDateTime": "2022-02-04 12:33:12",           "msgType": 62,           "rnd": "X5pGjoNkeSCHa4v80KtF",           "hash":"kolW0q5MoPBOIYXgyh0ymKzKsfXiQ8j356eOKicoaNmsSW3fj+43UEpqYu95Bjh73Ugdg54yIIiB35sL4uzHoQ==",  "language": 2        },        "body": {           "gatewayOrderId": "64c8cec9-fbb9-4e53-9b7c-test",           "STAN": "673700",           "approvalCode": "817646",           "shopID": "Shop123",           "merchantOrderID": "Order123",           "amount": 1000,           "currencyCode": 941,           "actionSuccess": "1",           "authorized": "1",           "completed": "1",           "voided": "0",           "refunded": "0",           "creditCardName": "VISA",           "creditCardNumber": "487077\*\*\*\*\*\*6558",           "transactionDateTime": "2022-02-04 12:33:16",           "partner": "Bib",           "ECI": "5",           "expirationDate": "2410"        }     }  } |

#### **MarketplaceTransactions output parameters**

**MarketplaceTransactions**message output parameters

|  |  |  |  |
| --- | --- | --- | --- |
| **Parameter** | **Type** |  | **Description** |
| errorCode | Integer | M | Error code |
| errorMessage | String | M | Description of errorCode |

**MarketplaceTransactions** message response example

|  |
| --- |
| {     "data": {        "header": {           "companyID": 123456,           "responseDateTime": "2022-02-04T12:07:46.080927",           "msgType": 62,  "language": 2        },        "body": {           "errorCode": 0,           "errorMessage": "OK"        },        "status": {           "errorCode": 0,           "errorMessage": "OK",           "dateTime": "2022-02-04T12:07:46.080927"        }     }  } |

# **PaySpot e-money payments**

### **PaySpot e-money messages**

### **E-money payment authorization (msgType 72)**

**[POST]: /api/payspotemoney/payment**

**REQUEST URL PRODUCTION:**[**https://www.nsgway.rs:50010/api/payspotemoney/payment**](https://www.nsgway.rs:50010/api/payspotemoney/payment)**REQUEST URL TEST:** [**https://test.nsgway.rs:50009/api/payspotemoney/payment**](https://test.nsgway.rs:50009/api/payspotemoney/payment)

Message type ***MsgType=72 emoney payment authorization*** is used to send data related to payment authorization.

#### **Emoney payment authorization input parameters**

**Emoney payment authorization**message input parameters

|  |  |  |  |
| --- | --- | --- | --- |
| **Parameter** | **Type** |  | **Description** |
| pan | String | M | PaySpot unique emoney identificator/number. For vouchers a barcode number needs to be entered or scanned. |
| panAlias | String | M | PaySpot token for emoney instrument |
| secureCode | String | M | PaySpot emoney secure code. |
| amount | Decimal | M | Charging amount. |
| merchantOrderID | String | M | Unique identifier of the order received from merchant. |
| expiryDate | String | M | PaySpot emoney expiry date |
| currency | String | M | Currency code |
| customerEmail | String | M | The email of the customer who owns the e-money instrument, mandatory if customerId is empty |
| customerId | String | M | A unique reference of the e-money instrument user in the merchant's system, mandatory if customerEmail is empty |

**Emoney payment authorization** message request example

|  |
| --- |
| {     "data": {        "header": {           "companyID": 123456,           "requestDateTime": "2022-02-04 12:33:12",           "msgType": 72,           "rnd": "X5pGjoNkeSCHa4v80KtF",           "hash":"kolW0q5MoPBOIYXgyh0ymKzKsfXiQ8j356eOKicoaNmsSW3fj+43UEpqYu95Bjh73Ugdg54yIIiB35sL4uzHoQ==",  "language": 2        },        "body": {           "pan": "1234567890",           "panAlias": "123-test-123",           "secureCode": "6677",           "amount": 1000,           "merchantOrderID": "Order123",           "expiryDate": "03/25",           "currency": "941",  "customerEmail": "customerEmail@gmail.com",  "customerId": "test123"        }     }  } |

#### **Emoney payment authorization output parameters**

**Emoney payment auhorization**message output parameters

|  |  |  |  |
| --- | --- | --- | --- |
| **Parameter** | **Type** |  | **Description** |
| authorizationId | String | M | Unique authorization ID in the PaySpot system if authorization is successful, otherwise NULL |
| authorizationDateTime | Sring | M | Date and time of authorization in format YYYY-MM-DD hh:mm:ss |
| amount | Decimal | M | Authorization amount if authorization successful, otherwise 0. |
| result | Integer | M | 1 – Successful authorization  -1 – Failed authorization |
| authNumber | String | M | Authorization number |
| transactionID | String | M | Transaction number in PaySpot system |
| orderId | String | M | Merchant order number |
| shopId | String | M | Represents a unique identification of web shop assigned to marketplace either by PGW or PaySpot. |
| currency | String | M | Currency code |
| responseCode | Integer | M | Response code |
| responseMsg | String | M | Response message |
| panAlias | String | M | PaySpot token for emoney instrument |
| redemptionCode | String | M | Redemption code for emoney instrument |
| maskedPan | String | M | Represents a masked version of full card PAN, it includes first 6 and last 4 digits of the card PAN |
| expiryDate | String | M | Expiry date of emoney instrument |
| customerEmail | String | M | Customer Emai from request |
| customerId | String | M | Customer ID from request |
| errorCode | Integer | M | Error code |
| errorMessage | String | M | Description of errorCode |

**Emoney payment authorization** message response example

|  |
| --- |
| {     "data": {        "header": {           "companyID": 123456,           "responseDateTime": "2022-02-04T12:07:46.080927",           "msgType": 72,  "language": 2        },        "body": {           "authorizationId": "authorization122345",           "authorizationDateTime": "2024-03-12T15:47:05.367323",           "amount": 1000.00,           "result": "00",           "currency": ”941”,           "transactionID": "12345",           "shopId": "803123451",           "panAlias": "123-test-123",  "redemptionCode": "524715",  "maskedPan": "1725\*\*\*\*\*\*8196",  "expiryDate": "03/25",  "authNumber": "306589",  "customerEmail": "customerEmail@gmail.com",  "customerId": "test123",           "responseCode": 0,           "responseMessage": "OK"           "errorCode": 0,           "errorMessage": "OK"        },        "status": {           "errorCode": 0,           "errorMessage": "OK",           "dateTime": "2022-02-04T12:07:46.080927"        }     }  } |

#### **Response code**

The field **ResponseCode** can have the following values:

|  |  |
| --- | --- |
| Value | Description |
| 1 | Kreiran |
| 2 | Aktivan |
| 3 | Privremena blokada |
| 4 | Interna privremena bokada |
| 5 | Interna trajna blokada |
| 6 | Istekao |
| 7 | Storniran |
| 8 | Ugašen |
| 9 | Storno blokada |
| 10 | Obrisan |

### **E-money details (msgType 74)**

**[POST]: /api/payspotemoney/details**

**REQUEST URL PRODUCTION:**[**https://www.nsgway.rs:50010/api/payspotemoney/details**](https://www.nsgway.rs:50010/api/payspotemoney/details)**REQUEST URL TEST:** [**https://test.nsgway.rs:50009/api/payspotemoney/details**](https://test.nsgway.rs:50009/api/payspotemoney/details)

Message type ***MsgType=74 emoney details*** is used to get data related to PaySpot emoney.

#### **Emoney details input parameters**

**Emoney details**message input parameters

|  |  |  |  |
| --- | --- | --- | --- |
| **Parameter** | **Type** |  | **Description** |
| pan | String | M | PaySpot unique emoney identificator/number. For vouchers a barcode number needs to be entered or scanned. |
| panAlias | String | M | PaySpot token for emoney instrument |
| secureCode | String | M | PaySpot emoney secure code. |
| expiryDate | String | M | Expiry date of emoney vouher |

**Emoney payment authorization** message request example

|  |
| --- |
| {     "data": {        "header": {           "companyID": 123456,           "requestDateTime": "2022-02-04 12:33:12",           "msgType": 74,           "rnd": "X5pGjoNkeSCHa4v80KtF",           "hash":"kolW0q5MoPBOIYXgyh0ymKzKsfXiQ8j356eOKicoaNmsSW3fj+43UEpqYu95Bjh73Ugdg54yIIiB35sL4uzHoQ==",  "language": 2        },        "body": {           "pan": "1234567890",           "panAlias": "123-test-123",           "secureCode": "6677"           "expiryDate": "03/25"        }     }  } |

#### **Emoney details output parameters**

**Emoney details**message output parameters

|  |  |  |  |
| --- | --- | --- | --- |
| **Parameter** | **Type** |  | **Description** |
| availableAmount | Decimal | M | Available amount |
| initialAmount | Decimal | M | Initial amount (amount when emoney was purchased) |
| validUntil | String | M | ISO datetime string representing until emoney is valid |
| cardStatus | Integer | M | Card status code, described below |
| cardStatusDescr | String | M | Description of cardStatus |
| errorCode | Integer | M | Error code |
| errorMessage | String | M | Description of errorCode |

**Emoney details** message response example

|  |
| --- |
| {     "data": {        "header": {           "companyID": 123456,           "responseDateTime": "2022-02-04T12:07:46.080927",           "msgType": 74,  "language": 2        },        "body": {           "availableAmount": 432.00,           "initialAmount": 1000.00,           "validUntil": "2024-02-04T12:07:46.080927",           "cardStatus": 1,           "cardStatusDescr": “Active”,           "errorCode": 0,           "errorMessage": "OK"        },        "status": {           "errorCode": 0,           "errorMessage": "OK",           "dateTime": "2022-02-04T12:07:46.080927"        }     }  } |

#### **Card status codes**

The field **cardCode** can have the following values:

|  |  |
| --- | --- |
| Value | Description |
| 1 | Kreiran |
| 2 | Aktivan |
| 3 | Privremena blokada |
| 4 | Interna privremena bokada |
| 5 | Interna trajna blokada |
| 6 | Istekao |
| 7 | Storniran |
| 8 | Ugašen |
| 9 | Storno blokada |
| 10 | Obrisan |

### **Emoney voucher list (msgType 73)**

**[POST]: /api/payspotemoney/voucherList**

**REQUEST URL PRODUCTION:**[**https://www.nsgway.rs:50010/api/payspotemoney/voucherList**](https://www.nsgway.rs:50010/api/payspotemoney/voucherList)**REQUEST URL TEST:** [**https://test.nsgway.rs:50009/api/payspotemoney/voucherList**](https://test.nsgway.rs:50009/api/payspotemoney/voucherList)

Message type ***MsgType=73 emoney voucher list*** is used to get data related to PaySpot emoney vouchers for specific customer.

#### **8.1.3.1 Emoney voucher list input parameters**

**Emoney voucher list** message input parameters

|  |  |  |  |
| --- | --- | --- | --- |
| **Parameter** | **Type** |  | **Description** |
| customerEmail | String | M | The email of the customer who owns the e-money instrument, mandatory if customerId is empty |
| customerId | String | M | A unique reference of the e-money instrument user in the merchant's system, mandatory if customerEmail is empty |

**Emoney voucher list** message request example

|  |
| --- |
| {     "data": {        "header": {           "companyID": 123456,           "requestDateTime": "2022-02-04 12:33:12",           "msgType": 73,           "rnd": "X5pGjoNkeSCHa4v80KtF",           "hash":"kolW0q5MoPBOIYXgyh0ymKzKsfXiQ8j356eOKicoaNmsSW3fj+43UEpqYu95Bjh73Ugdg54yIIiB35sL4uzHoQ==",           "language": 2        },        "body": {           "customerEmail": "customerEmail@gmail.com",           "customerId": "test123"        }     }  } |

#### **Emoney voucher list output parameters**

**Emoney voucher list**message output parameters

|  |  |  |  |
| --- | --- | --- | --- |
| **Parameter** | **Type** |  | **Description** |
| availableAmount | Decimal | M | Available amount |
| initialAmount | Decimal | M | Initial amount (amount when emoney was purchased) |
| validUntil | String | M | ISO datetime string representing until emoney is valid |
| cardStatus | Integer | M | Card status code, described below |
| cardStatusDescr | String | M | Description of cardStatus |
| panAlias | String | M | PaySpot token for emoney instrument |
| redemptionCode | String | M | Redemption code for emoney instrument |
| maskedPan | String | M | Represents a masked version of full card PAN, it includes first 6 and last 4 digits of the card PAN |
| expiryDate | String | M | Expiry date of emoney instrument |
| customerEmail | String | M | Customer Emai from request |
| customerId | String | M | Customer ID from request |
| errorCode | Integer | M | Error code |
| errorMessage | String | M | Description of errorCode |

**Emoney voucher list** message response example

|  |
| --- |
| {    "data": {      "header": {        "companyID": 123456,        "externalRequestID": null,        "responseDateTime": "2025-02-25T13:12:27.686547",        "msgType": 73,        "language": 1      },      "body": {        "voucherDetails": [          {            "availableAmount": 0,            "initialAmount": 600,            "validUntil": "2026-02-28T00:00:00",            "cardStatus": 2,            "cardStatusDescr": "Aktivan",            "maskedPan": "1725\*\*\*\*\*\*2069",            "panAlias": "2f434568-123d-4f5f-84d3-ac55b71555fe",            "redemptionCode": "621579",            "expiryDate": "2026-02-2",            "customerEmail": "customerEmail@gmail.com",            "customerId": "test123"          },          {            "availableAmount": 0,            "initialAmount": 600,            "validUntil": "2026-02-28T00:00:00",            "cardStatus": 2,            "cardStatusDescr": "Aktivan",            "maskedPan": "1725\*\*\*\*\*\*8213",            "panAlias": "483ae115-ed67-4e9f-a963-79803555ffb4",            "redemptionCode": "580338",            "expiryDate": "2026-02-2",            "customerEmail": "customerEmail@gmail.com ",            "customerId": "test123"          }        ],        "errorCode": 0,        "errorMessage": "OK"      },      "status": {        "errorCode": 0,        "errorMessage": "OK",        "dateTime": "2025-02-25T13:12:27.686547"      }    }  } |

### **Emoney Purchase Callback URL Data**

ECommerce needs to have predefined an URL endpoint where transaction data and EMoney instrument data will be sent, after the purchase of the instrument, using HTTP POST method.

**CallbackURL** message input parameters

|  |  |  |  |
| --- | --- | --- | --- |
| **Parametar** | **Type** |  | **Description** |
| **{** |  |  |  |
| rnd | String | M | Random string used for hash calculation |
| hash | String | M | Hash calculated as explained in the section below. **It is mandatory to calculate the hash on the eCommerce server.** |
| maskedPan | String | M | Masked Pan of emoney |
| expiryDate | String | M | Expiry date of emoney |
| panAlias | String | M | Token of emoney |
| redemptionToken | String | M | Redemption token |
| orderID | String | M | Merchant unique order id |
| **}** |  |  |  |

#### **8.2.1 Hash Calculation**

This section explains how eCommerce needs to generate the hash to know that request come from PaySpot Payment Service.

**Hash – SHA 512 Algorithm**

The hash is created by doing the SHA512 algorithm from the given string, then doing base64 encoding. Parameters in the hash string must be separated by “|”.

**plaintext = panAlias + | + rnd + | + redemptionToken + | + secretKeyCallback**

\*Parameters within the hash string must not contain backslash ("\") and pipeline ("|") characters.

\*secretKeyCallback will be delivered by the PaySpot system.

**Example:**

plaintext = order1234567890|shopIDTest|100.32|00|Test123456!

hash = Base64(SHA512(plaintext))

### **8.3 Emoney Payment Callback URL Data**

ECommerce needs to predefined an URL endpoint where EMoney transaction status will be sent using HTTP POST method. This callback is sent to a separate URL from the Emoney purchase callback, therefore you need to provide one URL for purchase and another URL for payment callback.

**CallbackURL** message input parameters

|  |  |  |  |
| --- | --- | --- | --- |
| **Parametar** | **Type** |  | **Description** |
| **{** |  |  |  |
| orderID | String | M | Merchant order from initial request |
| amount | String | M | Total value of the purchase order/purchase for which the payment was initiated |
| currency | String | M | Currency |
| result | String | M | Result of the transaction. Described below in section 5.4 |
| hash | String | M | Hash calculated as explained in the sections below depending on payment method. **It is mandatory to calculate the hash on the eCommerce server.** |
| maskedPan | String | M | Masked Pan of emoney instrument |
| expiryDate | String | M | Expiry date of emoney instrument |
| panAlias | String | M | Token of emoney instrument |
| customerEmail | String | M | The email of the customer who owns the e-money instrument |
| customerId | String | M | A unique reference of the e-money instrument user in the merchant's system |
| redemptionCode | String | M | Redemption code |
| responseCode | String | M | Represents authorization code for emoney transaction |
| responseMsg | String | M | Represents description of the authorization code for emoney transaction |
| **}** |  |  |  |

#### **8.3.1 Hash Calculation for Emoney Payment**

This section explains how eCommerce needs to generate the hash to know that request come from PaySpot Payment Service.

**Hash – SHA 512 Algorithm**

The hash is created by doing the SHA512 algorithm from the given string, then doing base64 encoding. Parameters in the hash string must be separated by “|”.

**plaintext = orderID + | + amount + | + result + | + secretKeyCallback**

\*Parameters within the hash string must not contain backslash ("\") and pipeline ("|") characters.

\*secretKeyCallback will be delivered by the PaySpot system.

**Example:**

plaintext = order1234567890 |100.32|00|Test123456!

hash = Base64(SHA512(plaintext))

# **9. Appendix**

### **9.1 Workflows**

### **9.1.1 Payment order request**

