switchio

TECHNICAL DOCUMENT

GSA API

CREATOR	**MONET+,a.s.** Za Dvorem 505, Zlín - Štípa
UPDATED	**24. 9. 2025**
VERSION #	**3.07.0**

CONTENT

DOCUMENT VERSION HISTORY

VERSION	DATE	AUTHOR	DESCRIPTION
1.00	06.10.2022	M+ (AAD)	1st version of the document
1.01	04.11.2022	M+ (PZI)	Changed ending of request's body from **\r\n** to **\r\n\r\n** Additional info about Transaction status
1.02	16.11.2022	M+ (PZI)	The cancel request can cancel tokenization
1.03	22.11.2022	M+ (PZI)	Update Token result enum
1.04	24.11.2022	M+(PZI)	Change transport payment transaction parameter **transportData** encoding to Base64
1.05	2.1.2023	M+(PZI)	Scan request can be cancelled with cancel request
1.06	23.02.2023	M+(AAD)	Added status "Idle" to Operation status enum
1.07	13.03.2023	M+(AAD)	Clarification of StatusCode 400 & 404
1.08	22.05.2023	M+(AAD)	Fixed authCode param to string
1.09	24.07.2023	M+(AAD)	Added params to Transaction result • tipAmount • refundId
1.10	28.07.2023	M+(AAD)	Added Transaction result v5
1.11	19.9.2023	M+(AAD)	Added Get taps v5, Get denylist last update v5
2.00	04.10.2023	M+(AAD, PZI)	Fusion of GSA API doc and GSA API with Bluetooth. Complete revision.
2.01	7.3.2024	M+(PZI)	Change text message for Already processed transaction to Duplicate transaction SwitchioPay - 4.4.0
3.00	10. 10. 2024	M+ (KLA)	Added new endpoints: • PREAUTH DECREMENT • QR PAYMENT Added new endpoint versions for v6. Added new Transaction result for v6 - Response v6. Minor changes and corrections.
3.01	5. 11. 2024	M+ (KLA)	Updated:

			• Transport payment request with new optional fields • Added new Terminal Error Code Added: • Cashback
3.02	25. 2. 2025	M+(OMR)	Added new Transaction result for v7 - Response v7.

3.03.0	1. 5. 2025	M+(PZI)	API changes:
			• Added new Terminal Error Code
			  ○ `READING_CARD_FAILED(8008)`
			  ○ `SECOND_TAP_CARD_MISMATCH(8009)`
			• Adjusted token request with new optional parameters to show amount and currency
			• Adjusted transport payment request by adding additional fields TRANSPORT PAYMENT
			Added new endpoints for:
			• TRANSPORT REFUND
			• TRANSPORT REVERSAL

3.04.0	2. 6. 2025	M+(FMI)	Added new transactions endpoints • MOTO PREAUTHORIZATION • EXTERNAL PREAUTH INCREMENT • EXTERNAL PREAUTH DECREMENT
3.05.0	17. 7. 2025	M+(PZI)	Added new **transportData** field for Transport refund request for version 4.7.1 SwitchioPay
3.06.0	31. 7. 2025	M+(PMR)	Updated the TerminalErrorCodes table with new error mappings.
3.07.0	24. 9. 2025	M+ (FMI)	Added new endpoint versions for v8. Added new Transaction result for v8 - Result structure v8. Added new Transaction results fields: **additionalFeeAmount**, **transferId** and fields for future releases **offlineAuthentication** , **systemTraceAuditNumber** and **gatewayUniqueTransactionId**
3.07.1	13. 10. 2025	M+ (OMR)	Fixed request for Cashback parameter amount to saleAmount

APPLICATION VERSION HISTORY

ENDPOINT VERSION	APPLICATION VERSION	DESCRIPTION OF CHANGES
v1	1.8.0	First release
v2	3.0.0	New endpoints • Last transaction • Get successful transactions
v4	3.4.0	New endpoint • Card Verify
v5	4.1.0	New endpoints • Get taps v5 • Get denylist last update v5 • Preauth decrement - from Switchio Pay v4.4.0 • QR payment - from Switchio Pay v4.4.0 New fields in TRANSACTION RESULT response • monetToken • par • cardToken
v6	4.5.0	New fields in transactionResult response - see Response v6: • dccOffer • dccResult • surcharge • externalTransactionId New fields in Transport payment request: • transportAmount • transportCurrencyCode New endpoint from Switchio Pay v4.6.0: • Cashback
v7	4.7.0	Added new endpoints for: • TRANSPORT REFUND • TRANSPORT REVERSAL • MOTO PREAUTHORIZATION • EXTERNAL PREAUTH INCREMENT • EXTERNAL PREAUTH DECREMENT
v8	4.8.0	New version of TransactionResult with new fields: • additionalFeeAmount • transferId • systemTraceAuditNumber - RFU • offlineAuthentication - RFU • gatewayUniqueTransactionId - RFU

1. INTRODUCTION

The present document is intended for companies which implement the Gapa Simple Api (GSA) protocol and GSA with Bluetooth protocol to communicate with the MONET+ system.

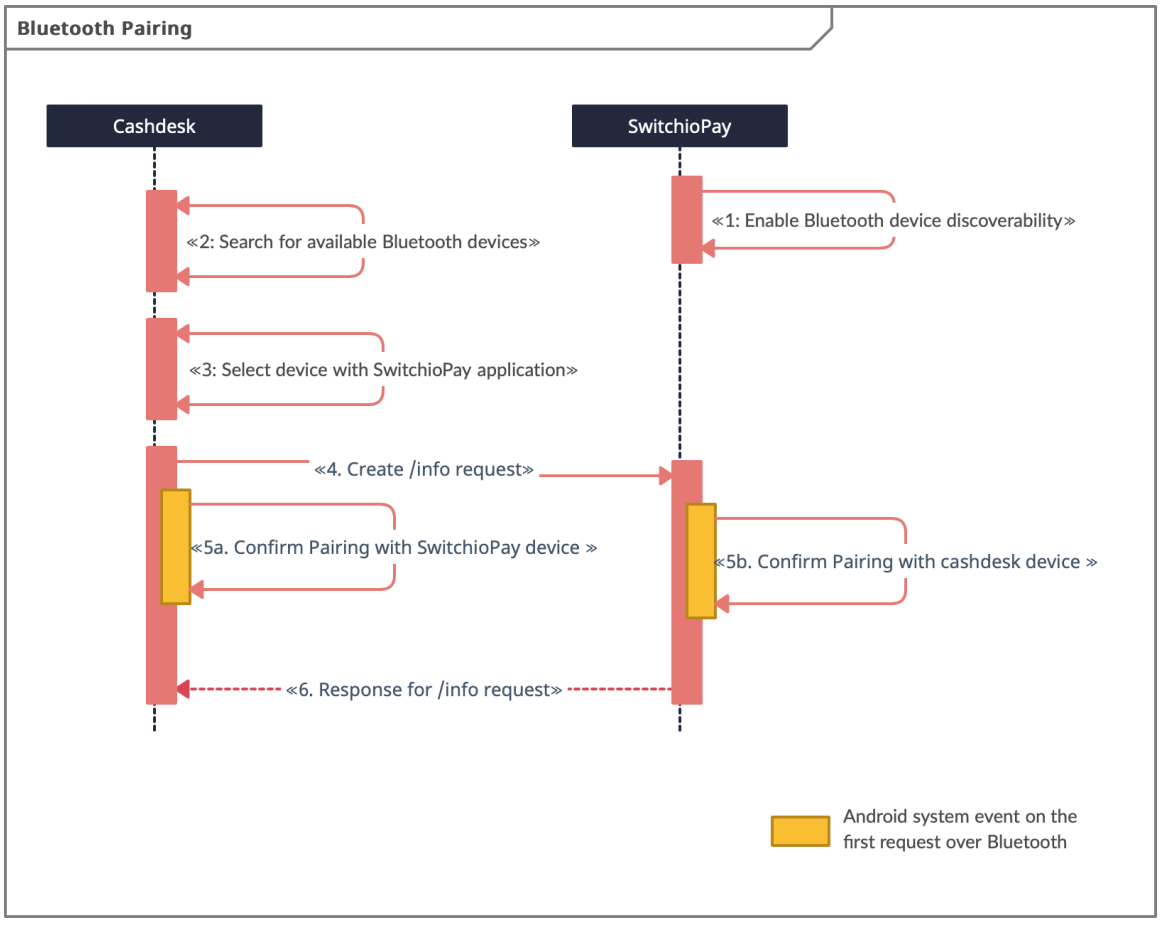
Protocols include supported requests and responses with JSON examples and the description on how to proceed with establishing connection between SwitchioPay and third party app.

1.1. BLUETOOTH CONNECTION

Bluetooth connection is supported in Switchio Pay since version 3.3.0.

Following diagram describes how to pair devices and establish bluetooth connection.

The first step to establishing a connection and sending requests between cash register and SwitchioPay is to pair these two devices according to the diagram:



1. Turn on Bluetooth on the device where is SwitchioPay installed and enable Bluetooth discoverability of device following these steps in the SwitchioPay app:

a) Go to Settings > Bluetooth
b) Turn on discoverability - Device will be discoverable for 120 seconds

2. On consumer device (cash register) search for available bluetooth devices
3. From the list obtained in the second step, select device with a name corresponding to the name on SwitchioPay application in Bluetooth settings
4. To be able to create a request the connection has to be established by using the selected device and UUID **38400000-8cf0-11bd-b23e-10b96e4ef00d.** Here is the code you can use to establish the connection on an Android device with Kotlin:

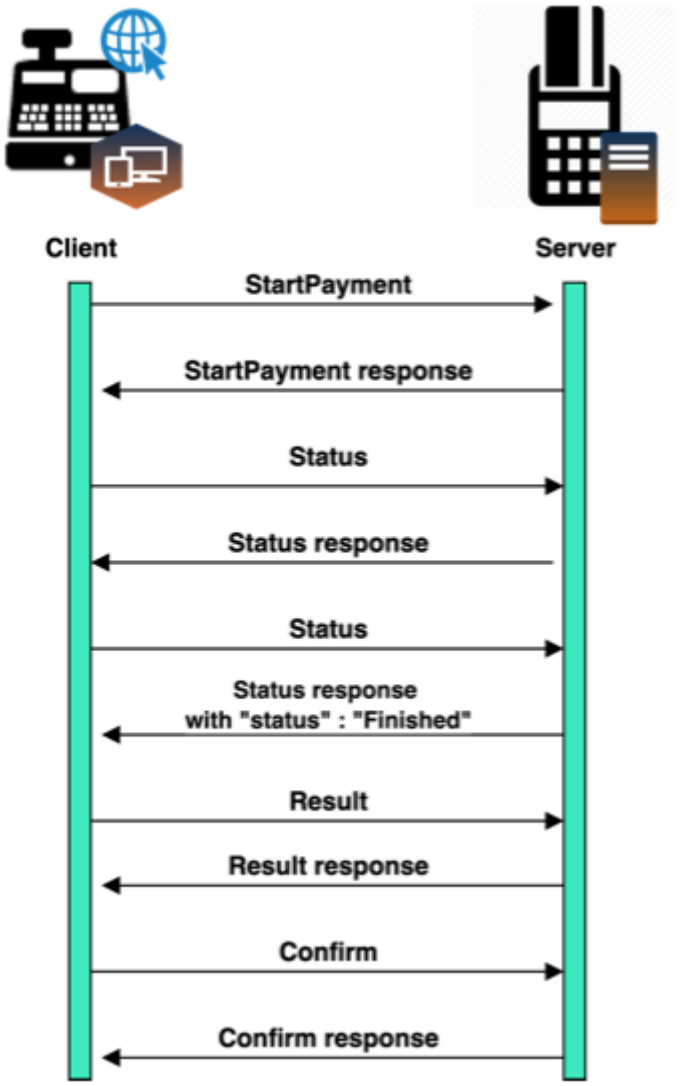
```
1     // 1. Get Bluetooth device
2     val selectedBluetoothDevice: BluetoothDevice = getSelectedDeviceFromList()
//
      get selected device, for example from RecyclerView
3     // 2. create socket with corresponding UUID, which identifies SwitchioPay
      Bluetooth interface
4     val serverSocket =
      selectedBluetoothDevice.createRfcommSocketToServiceRecord("38400000-8cf0-11b
d-b
      23e-10b96e4ef00d")
5     // 3. Establish connection
6     serverSocket.connect()
7     // 4. Use serverSocket.outputStream to deliver request to SwitchioPay
8     PrintWriter(serverSocket.outputStream, true).apply {
9         print("request")
10        flush()
11    }
12
13    .
14    .
15    .
16
17    // 5. Use serverSocket.inputStream to read response from SwitchioPay
18    serverSocket.inputStream
```

5. When the first request is sent, the Android system will recognize the action and will display a system dialog, which will request the user to pair the device or cancel the incoming pairing request.

1.2. CLIENT / SERVER CONNECTION

1.2.1. Static roles

Server is the terminal (SwitchioPay application)



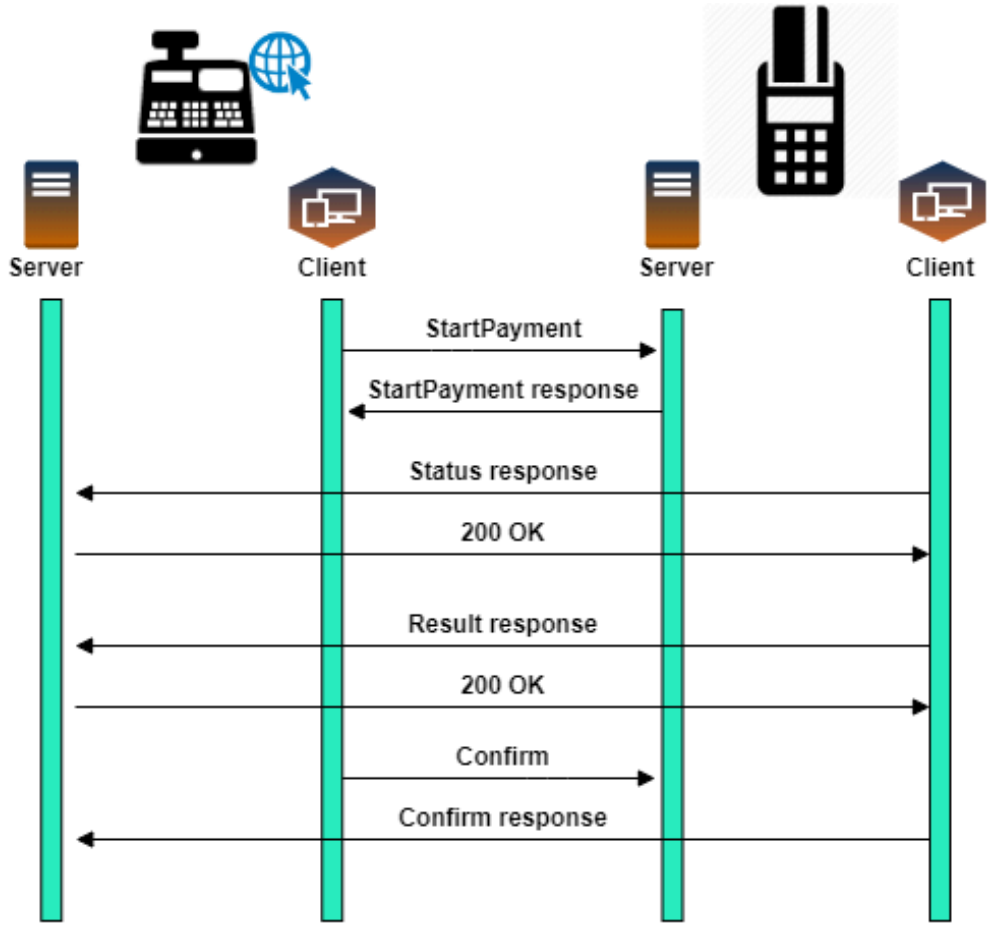
1.2.2. Dynamic roles

Both the cash register and the terminal work as client and server.

The difference is that with callback the cash register does not need to ask for status and result because the terminal pushes the status and the result automatically to the cash register every time when the status changes.

This payment flow is used for specific endpoints with **callback**.

- payment - /cbpayment
- MOTO - /cbpayment/moto
- refund - /cbrefund
- reversal - /cbreversal
- preauthorization - /cbpreauth
    - preauth complete - /cbpreauth/complete
    - preauth increment - /cbpreauth/increment
    - preauth cancel - /cbpreauth/cancel
- closing - /cbclosing



1.3. CASHDESK

Following section describes (http://demo.cashdesk.switchio.com/#/) the Monet+ demo-implementation of the cash register's application. This implementation communicates with a Switchio Pay application through the GSA protocol. The Switchio Pay is a payment application present in the terminal. Terminal works as the server, the cash register works as a client.

The communication between the terminal and the cash register is not via the public internet. Cash register doesn't have to have connectivity to the internet. But must be in the local network with the terminal.

1.3.1. Login

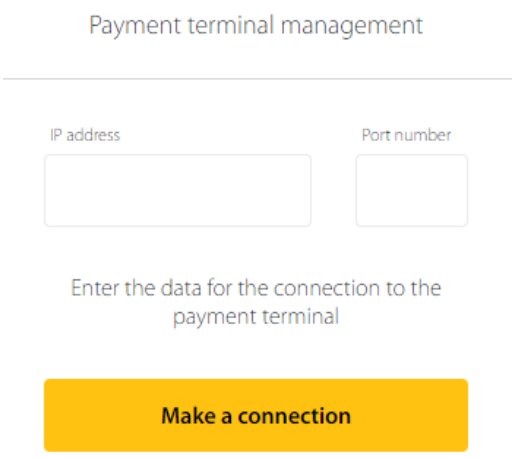
The web address will be provided by the Monet+.

To login into the application you will need the following:

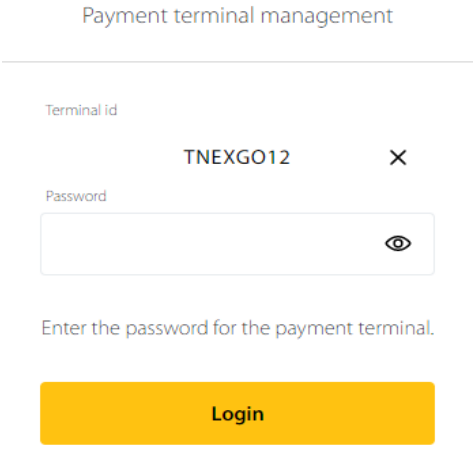
- IP address
- port number
- password

The IP address is subject to the local network. The port number can be set according to the client's request. And the password is generated and provided by the Monet+.

Enter the IP address and the port number to the colons and tap the *Make connection* button. (see Picture 1 - Login I) When the connection is made, a new tab occurs. Enter the password given from the Monet+ and tap the *Login* button.

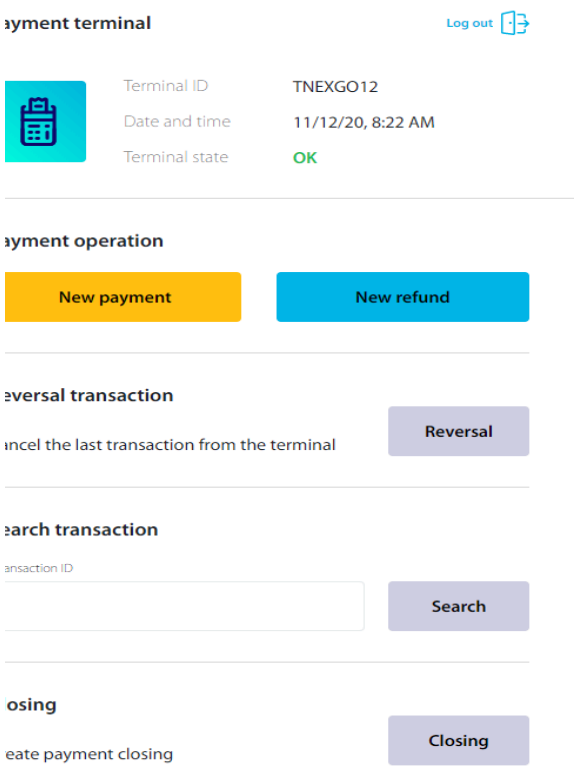


Pic. 1 *Login I*

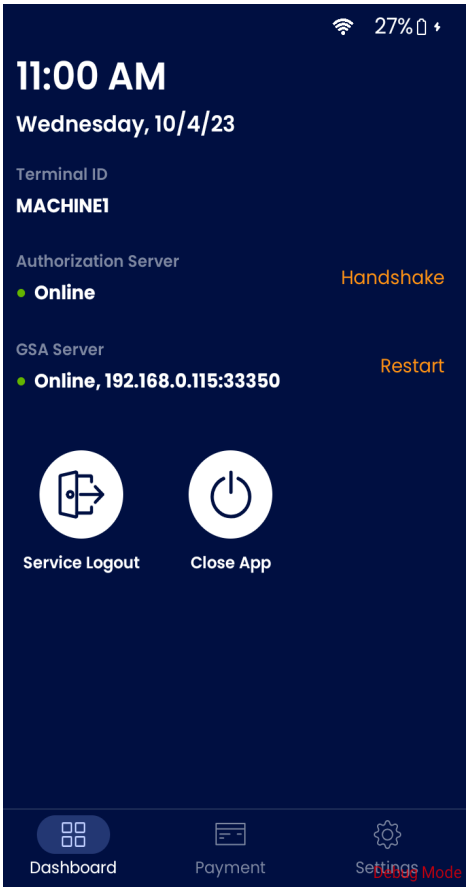


Pic. 2 *Login II*

When you're successfully logged, you will get to the homepage(see pic. 3 - The homepage on the web).



Pic. 3 *The homepage on the web*



Pic. 4 *The terminal's dashboard*

Start the Switchio Pay application on the terminal, the *Dashboard* page(see pic. 4 - The terminal's dashboard) will display.

Methods called during the Login process are:

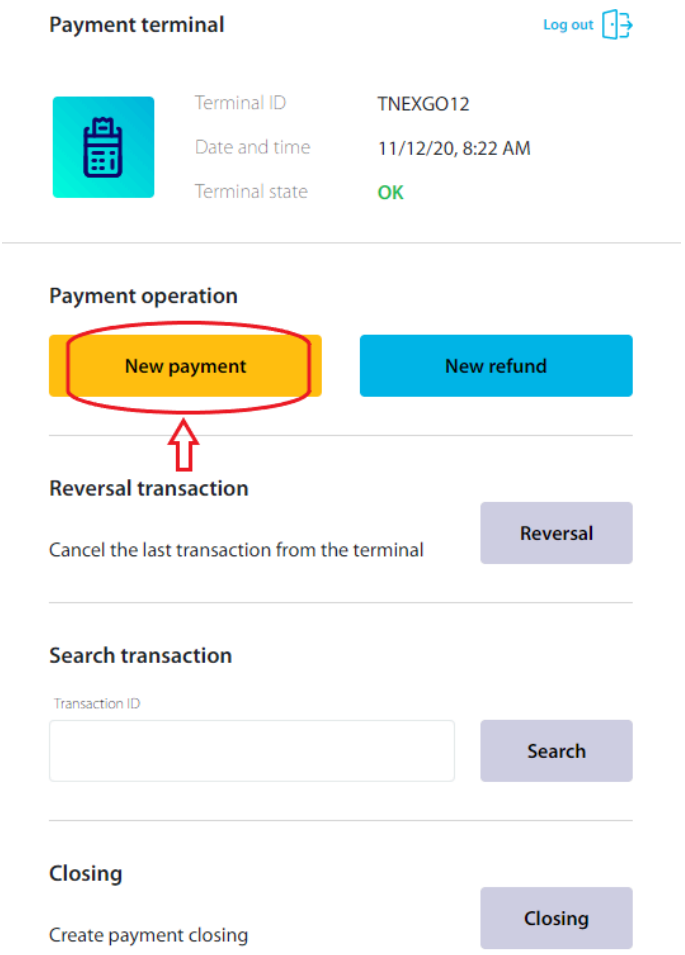
- Info method
- Terminal Status

1.3.2. Terminal Operation Examples

1.3.2.1. New payment

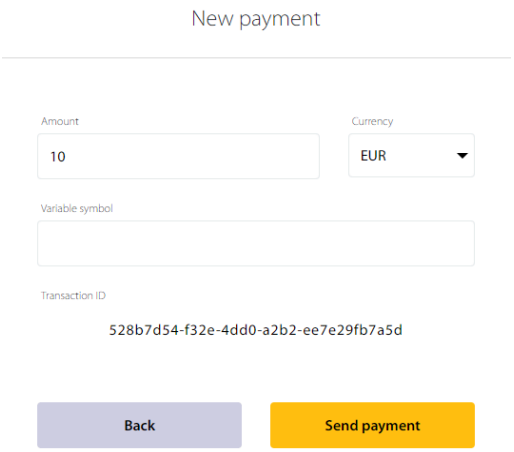
The new payment setup:

1. **Tap the *New payment* button.** (see pic. 5 - Tap the new payment button)



Pic. 5 *Tap the new payment button*

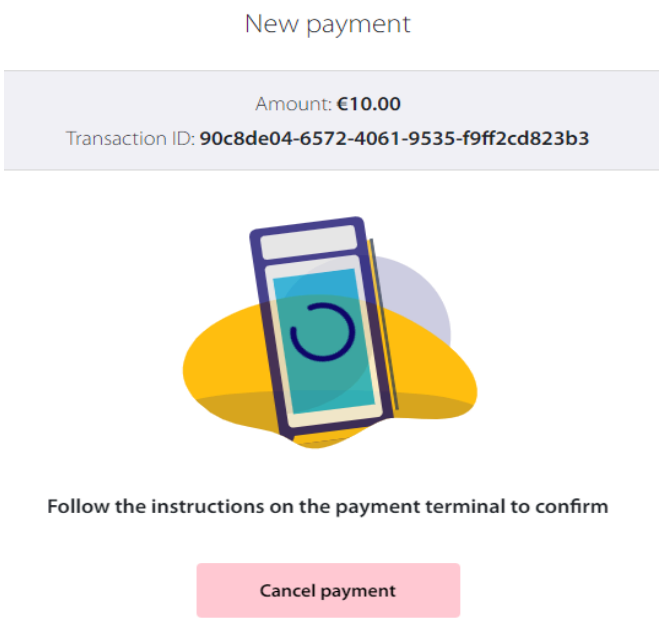
2. **Enter the *amount* payment**. Pick the *currency* from the scroll down button option. Entering the *Variable symbol* is optional. Tap the *Send payment* button (see pic. 6 - The new payment setting).



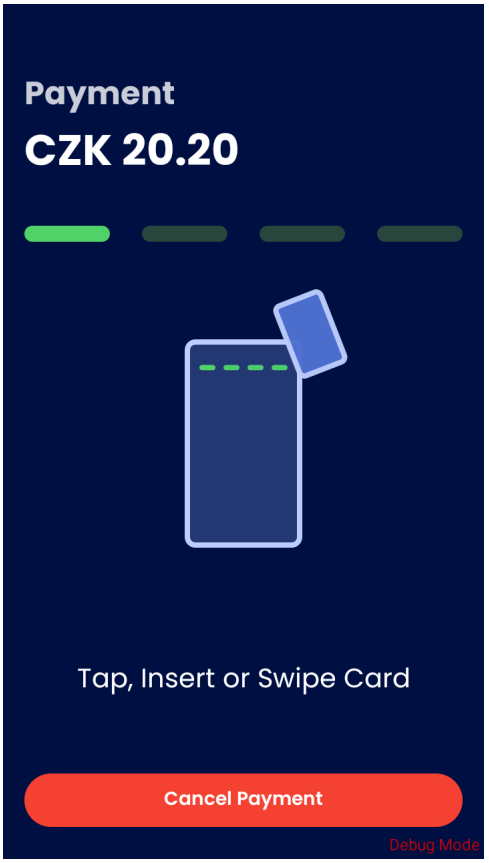
Pic. 6 *The new payment setting*

3. **Confirmation that the payment is being processed shows up.** (see pic.7 - Confirmation that payment being processed)

   Tap, Insert or Swipe the card to the terminal.(see pic.8 - Terminal payment process). At this point you are able to cancel the payment on both the terminal or the web cash register.

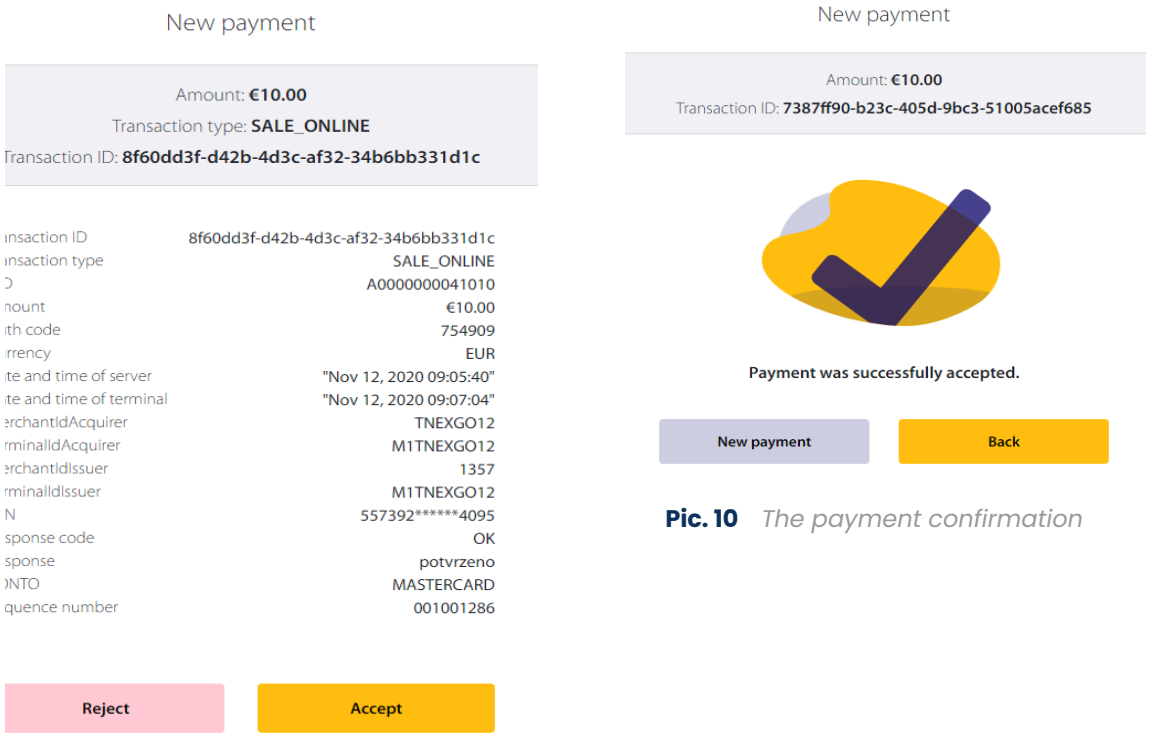


Pic. 7 *Confirmation that the payment being processed*



Pic. 8 *Terminal payment process*

4. **At the web cash register accept the payment** (see pic.9 - Accept payment)



Pic. 9 *Accept the payment*

Pic. 10 *The payment confirmation*

5. **The payment is confirmed** (see pic. 10 - The payment confirmation).

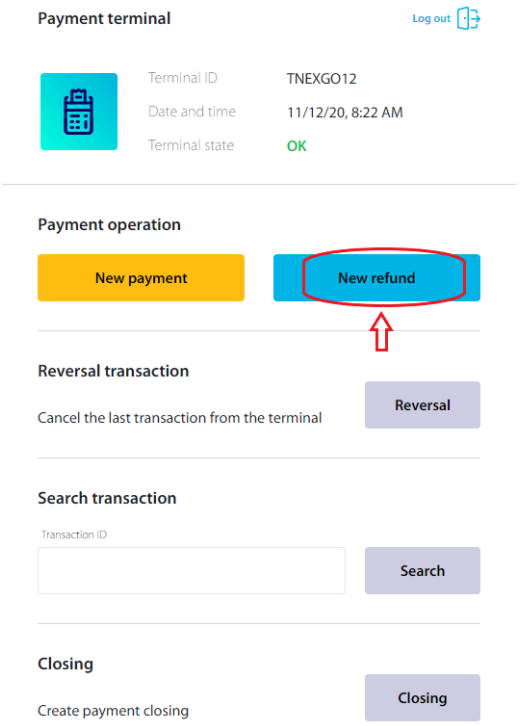
Methods called during the payment process are:

- Start payment
- Status
- Result
- Confirm

1.3.2.2. Refund

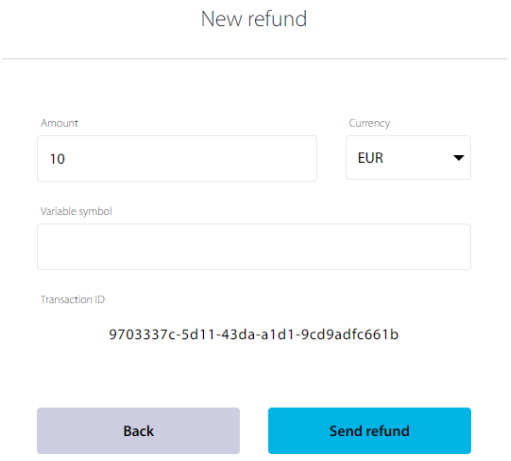
The refund setup:

1. **Tap the *refund* button.** (see pic. 11 - Tap the refund button)



Pic. 11 *Tap the refund button*

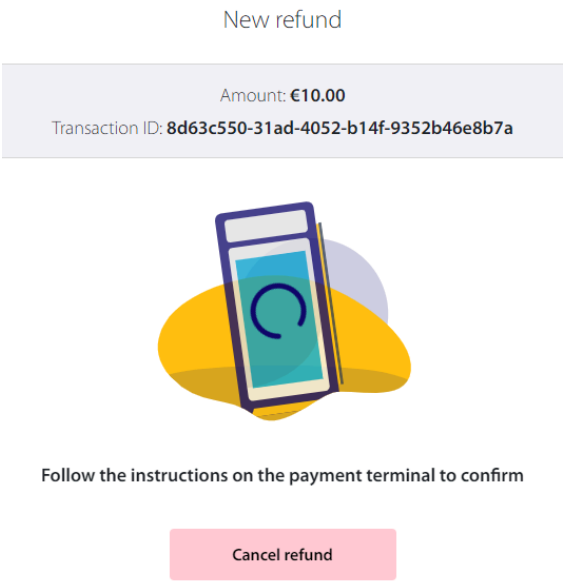
2. **Enter the *amount* of the refund.** Pick the *currency* from the scroll down button option. Entering the *Variable symbol* is optional. Tap the *Send refund* button (see pic. 6 - The refund setting)



Pic. 12 *The Refund setting*

3. **Confirmation that the refund is being processed shows up.** (see pic.13 - Confirmation that refund is being processed)

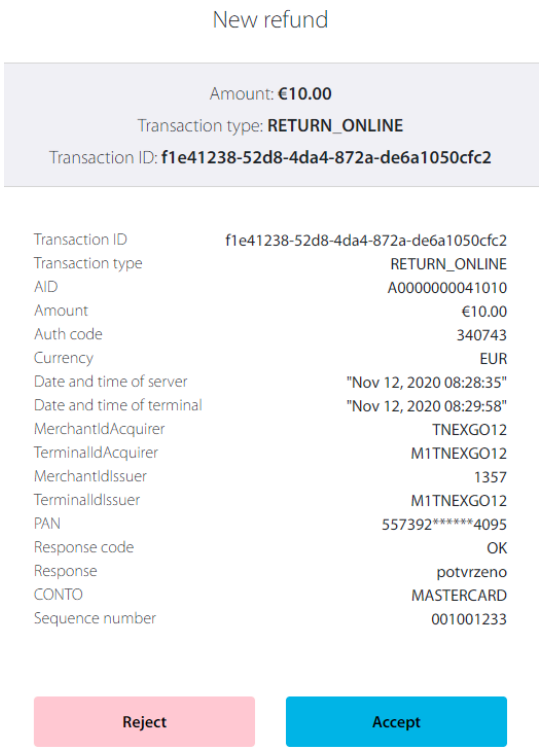
   Tap, Insert or Swipe the card to the terminal.(see pic.14 - Terminal refund process). At this point you are able to cancel the payment on both the terminal or the web cash register.



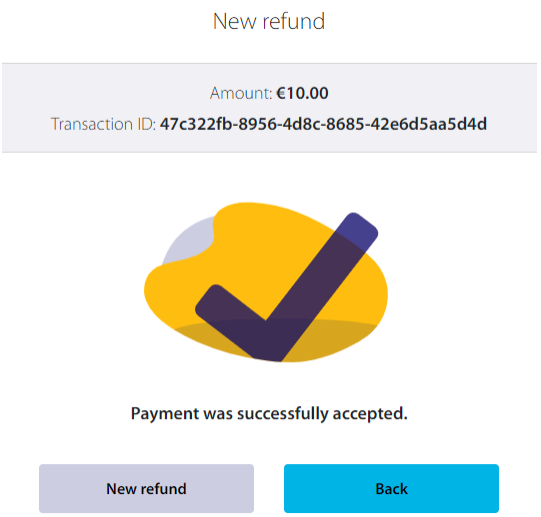
Pic. 13 *Confirmation that refund is being processed*

Pic. 14 *Terminal refund process*

4. **At the web cash register accept the payment.** (see pic.15 - Accept the refund)



Pic. 15 *Accept the refund*



Pic. 16 *The refund confirmation*

5. **The payment is confirmed.** (see pic. 16 - The refund confirmation).

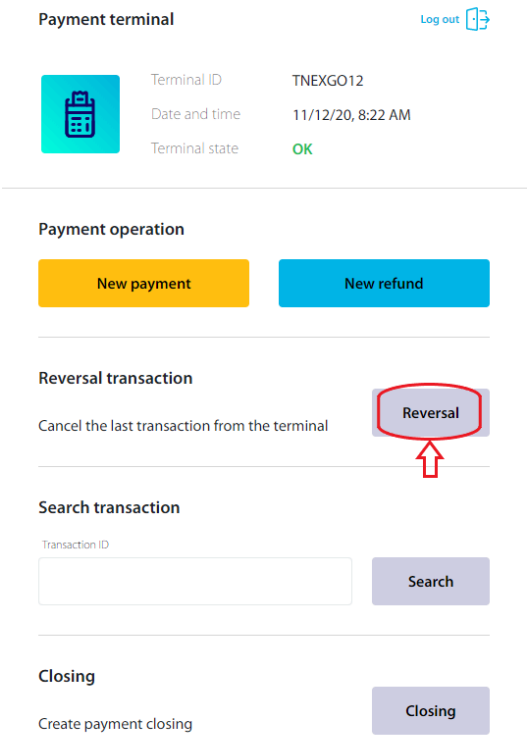
Methods called during the refund process are:

- Start refund
- Status
- Result
- Confirm

1.3.2.3. Reversal

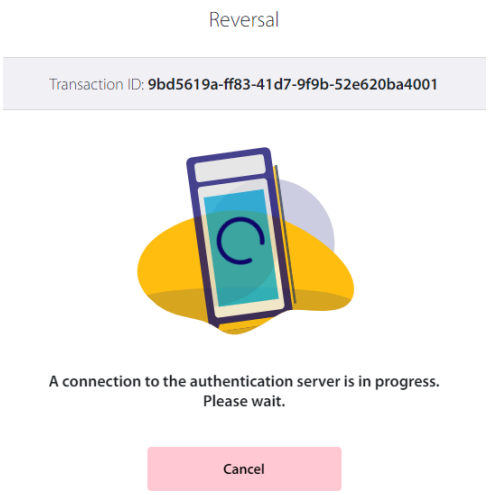
The reversal setup:

1. **Tap the *Reversal* button.** (see pic.17 - Tap the reversal button).



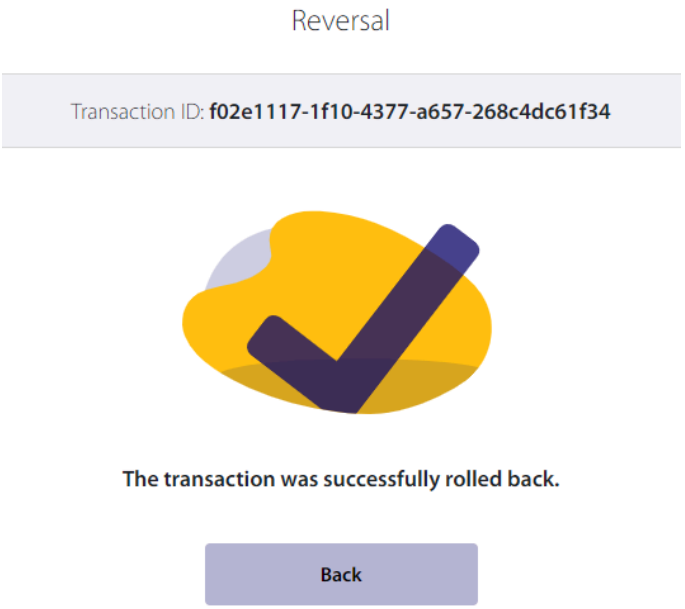
Pic. 17 *Tap the reversal button*

2. **Confirmation that the reverse of the last payment is being processed shows up.**
(see pic.18 - Confirmation that the reverse of the last payment is being processed)



Pic. 18 *Confirmation that the reverse of the last payment is being processed*

3. **The payment is confirmed.** (see pic. 20 - The reversal confirmation).



Pic. 19 *The reversal confirmation*

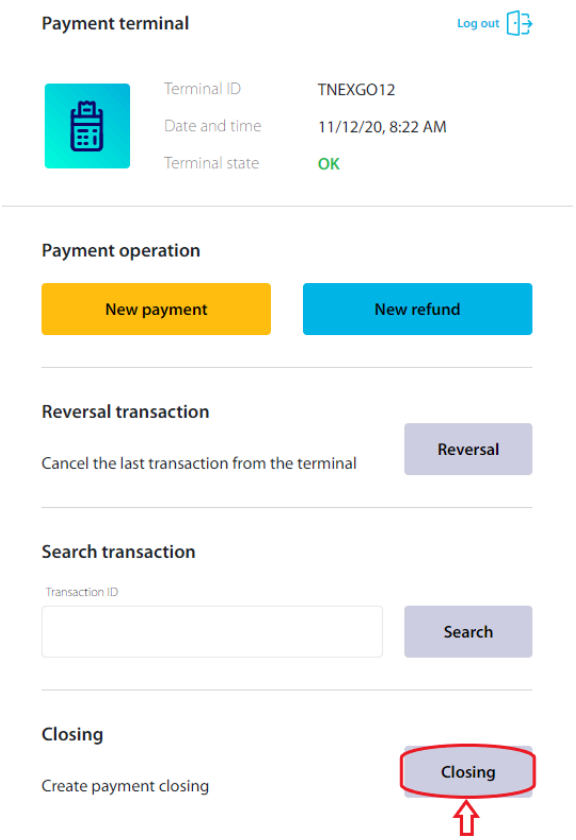
Methods called during the reversal process are:

- Start reverse
- Status
- Confirm

1.3.2.4. Closing

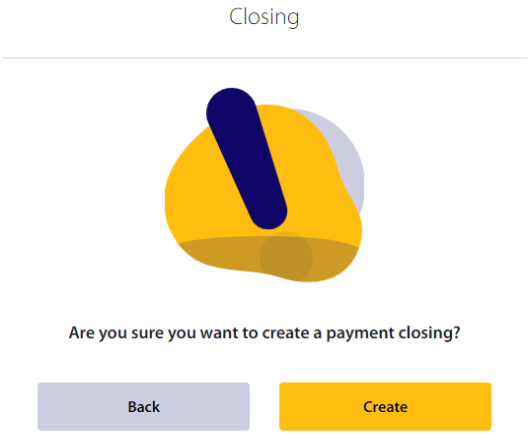
The closing setup:

1. **Tap the *closing* button.** (see pic.21 - Tap the closing button).



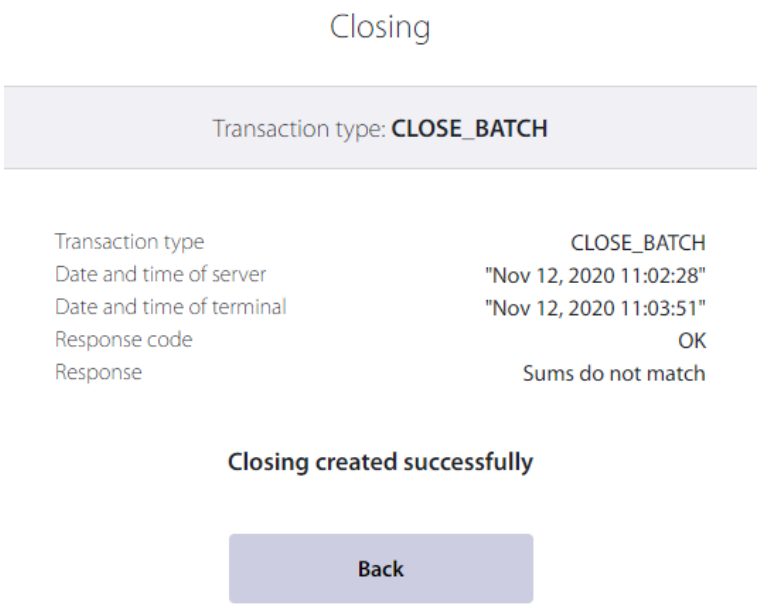
Pic. 20 *Tap the closing button*

2. **Tap the *Create* button.** (see pic.22 - Tap the Create button).



Pic. 21 *Tap the Create button*

3. **The confirmation of closing is displayed.** (see pic.23 - The confirmation of the closing).



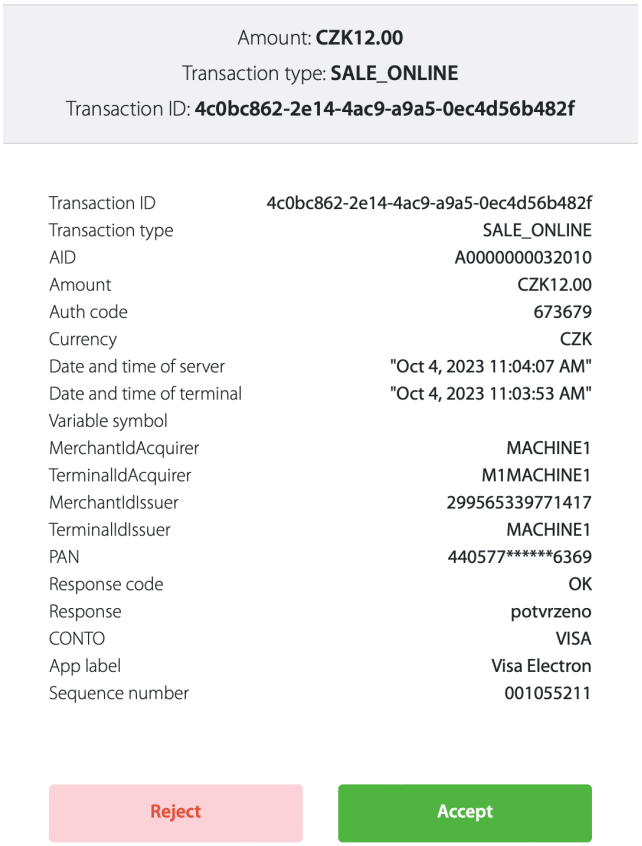
Pic. 22 *The confirmation of the closing*

Methods called during the closing process are:

- Start closing
- Status

1.3.2.5. Result

Pressing **Accept** will send **/confirm** request.



2. CONTROL ENDPOINTS

2.1. INFO

With the request, the cash register can query about the terminalID and GSA protocol version used in the application.

2.1.1. Request

GSA v1 - GET ip:port/paya/info
GSA v2 - GET ip:port/api/switchio/pay/v2/info
GSA v4 - GET ip:port/api/switchio/pay/v4/info
GSA v5 - GET ip:port/api/switchio/pay/v5/info
GSA v6 - GET ip:port/api/switchio/pay/v6/info
GSA v7 - GET ip:port/api/switchio/pay/v7/info
GSA v8 - GET ip:port/api/switchio/pay/v8/info

2.1.2. Response

Response example:

```
{
"protocol":"GSA",
"version":0,
"terminalId":"TESTTERM01"
}
```

PARAMETR	TYPE	DESCRIPTION
protocol	string	Name of the used protocol
version	int	Version of the protocol
terminalId	string	ID of the terminal

2.2. TERMINAL STATUS

It is used to check the status of the terminal, find out the time of the terminal and optionally check the connection to the authorization server.

2.2.1. Request

GSA v1 - POST ip:port/paya/terminal
GSA v2 - POST ip:port/api/switchio/pay/v2/terminal
GSA v4 - POST ip:port/api/switchio/pay/v4/terminal
GSA v5 - POST ip:port/api/switchio/pay/v5/terminal
GSA v6 - POST ip:port/api/switchio/pay/v6/terminal
GSA v7 - POST ip:port/api/switchio/pay/v7/terminal
GSA v8 - POST ip:port/api/switchio/pay/v8/terminal

Request example:

```
{
   "secureString":"password",
   "checkOnline":true
}
```

PARAMETER	TYPE	DESCRIPTION
secureString	string	Password of the terminal
checkOnline	bool	Information whether the test of connection to the authorization server is required

2.2.2. Response

Response example:

```
{
    "terminalId":"TESTTERM01",
    "dateTime":"1512469800000",
    "isOnline":0,
    "onlineStatus":"Not requested"
}
```

PARAMETER	TYPE	DESCRIPTION
terminalId	string	ID of the terminal
dateTime	long	Date and time on terminal Format unix timestamp.
isOnline	int	Status of the connection to the authorization server.

		0 - wasn't required 1 - OK 2 - Error
onlineStatus	string	Message of the connection to the authorization server status.

2.3. OPERATION STATUS

The Status method is called when we receive response of method:

- Start payment
- Start refund
- Start reverse
- Start closing

The purpose of the Status method is to inquire about the phase of the transaction (purchase, preauth, preauth completion, preauth cancel) in regular intervals (3 seconds). When the parameter *status* from this method response contains *Finished*, then the Result method is called to obtain the result of the transaction.

2.3.1. Request

GSA v1 - POST ip:port/paya/status
GSA v2 - POST ip:port/api/switchio/pay/v2/status
GSA v4 - POST ip:port/api/switchio/pay/v4/status
GSA v5 - POST ip:port/api/switchio/pay/v5/status
GSA v6 - POST ip:port/api/switchio/pay/v6/status
GSA v7 - POST ip:port/api/switchio/pay/v7/status
GSA v8 - POST ip:port/api/switchio/pay/v8/status

Request example:

```
{
    "transactionId":"1e6a1251-762f-45fc-85fb-227a1c3290b6",
    "secureString":"password"
}
```

PARAMETER	TYPE	DESCRIPTION
transactionId	string	ID of the transaction. Example: *1e6a1251-762f-45fc-85fb-227a1c3290b6*
secureString	string	Password of the terminal

2.3.2. Response

StatusCode: 200 → transaction has started, cash register is polling **/api/switchio/pay/v2/status** request until it receives the **"Finished"** state

Response example:

```
{
    "transactionId":"1e6a1251-762f-45fc-85fb-227a1c3290b6",
    "status":"WaitForCard"
}
```

PARAMETER	TYPE	DESCRIPTION
transactionId	string	ID of the transaction. Example: *1e6a1251-762f-45fc-85fb-227a1c3290b6*
status	string	Message of the Operation status. Enums of the status parameter are in the appendix section of GSA protocol documentation

StatusCode: 400 → Invalid data

StatusCode: 401 → Unauthorized

StatusCode: 404 → Transaction has not been found with the given originalTransactionId

Response example:

```
{
    "message":"Duplicate Transaction"
}
```

2.3.3. Operation status enum

Status

```
Idle
WaitForCard
WaitForPin
Connecting
Connected
Disconnected
Sending
Receiving
Finished
FinishedPlain      //only for token, transport init, transport payment, transport upload, scan
```

2.4. CONFIRM TRANSACTION

The cash register has to explicitly confirm some transactions, the confirmation means that the signature is correct, the receipt has been printed out and the payment is treated as authorized.
We can consider the payment is confirmed only when the terminal responds with isConfiparameter *isConfirm : true.* Upon expiration of the time period (1 minute), an automatic reversal of the unacknowledged transaction that required confirmation is performed.

2.4.1. Request

GSA v1 - POST ip:port/paya/confirm
GSA v2 - POST ip:port/api/switchio/pay/v2/confirm
GSA v4 - POST ip:port/api/switchio/pay/v4/confirm
GSA v5 - POST ip:port/api/switchio/pay/v5/confirm
GSA v6 - POST ip:port/api/switchio/pay/v6/confirm
GSA v7 - POST ip:port/api/switchio/pay/v7/confirm
GSA v8 - POST ip:port/api/switchio/pay/v8/confirm

Request example:

```
{
    "secureString":"password",
    "transactionId":"1e6a1251-762f-45fc-85fb-227a1c3290b6",
    "confirm":true
}
```

PARAMETER	TYPE	DESCRIPTION
secureString	string	Password of the terminal
transactionId	string	ID of the transaction. Example: *1e6a1251-762f-45fc-85fb-227a1c3290b6*
confirm	bool	Information whether is transaction confirmed or not

2.4.2. Response

StatusCode: 200 → Request is OK

Response example:

```
{
    "transactionId":"1e6a1251-762f-45fc-85fb-227a1c3290b6",
    "isConfirmed":true
}
```

PARAMETER	TYPE	DESCRIPTION
transactionId	string	ID of the transaction. Example: *1e6a1251-762f-45fc-85fb-227a1c3290b6*
isConfirmed	bool	information whether the transaction is confirmed or not.

StatusCode: 404 → Transaction has not been found with the given originalTransactionId

Response example:

```
{
    "message":"Not found"
}
```

2.5. CLOSING POLLING

The closing request starts the closing of the transaction. It should be used daily after the shift is finished. To get the closing request status, use the OPERATION STATUS endpoint. If the operation status is FINISHED, use the TRANSACTION RESULT endpoint to get the closing result.

2.5.1. Request

GSA v1 - POST ip:port/paya/closing
GSA v2 - POST ip:port/api/switchio/pay/v2/closing
GSA v4 - POST ip:port/api/switchio/pay/v4/closing
GSA v5 - POST ip:port/api/switchio/pay/v5/closing
GSA v6 - POST ip:port/api/switchio/pay/v6/closing

Request example:

```
{
    "secureString":"password",
    "request":{
        "transactionId":"1e6a1251-762f-45fc-85fb-227a1c3290b6",
        "daughterCompanyId":""
    }
}
```

PARAMETER		TYPE	DESCRIPTION
secureString		string	Password of the terminal
request		obj	object of the request
	transactionId	string	ID of the transaction. Example: *1e6a1251-762f-45fc-85fb-227a1c3290b6*
	daughterCompanyId	string	ID of the daughter company. Required only when virtualization is enabled.

2.5.2. Request with callback

This endpoint is not available for GSA over Bluetooth.
There is no need to call **/status** and **/result**.

GSA v1 - POST ip:port/paya/cbclosing
GSA v2 - POST ip:port/api/switchio/pay/v2/cbclosing
GSA v4 - POST ip:port/api/switchio/pay/v4/cbclosing
GSA v5 - POST ip:port/api/switchio/pay/v5/cbclosing
GSA v6 - POST ip:port/api/switchio/pay/v6/cbclosing

Request example:

```
{
    "secureString":"password",
    "request":{
        "transactionId":"1e6a1251-762f-45fc-85fb-227a1c3290b6"
        "daughterCompanyId":""
    },
    "progressCallback":{
        "ip":"172.25.11.14",
        "port":7777,
        "endpoint":"/gsa/progress"
    },
    "resultCallback":{
        "ip":"172.25.11.14",
        "port":7777,
        "endpoint":"/gsa/result"
    }
}
```

PARAMETER		TYPE	DESCRIPTION
progressCallback		obj	Object of progress to callback
	ip	string	IP to callback
	port	int	Port to callback
	endpoint	string	Endpoint of method to callback
resultCallback		obj	Object of result to callback
	ip	string	IP to callback
	port	int	Port to callback
	endpoint	string	Endpoint of the method to callback

2.5.3. Response

Response example:

```
{
    "transactionId":"1e6a1251-762f-45fc-85fb-227a1c3290b6",
    "isStarted":false,
    "status":"Other payment in progress"
}
```

PARAMETER	TYPE	DESCRIPTION
transactionId	string	ID of the transaction. Example: *1e6a1251-762f-45fc-85fb-227a1c3290b6*
isStarted	bool	Information whether the transaction started or not.
status	string	Message of the transaction status

2.6. CANCEL EXTERNAL TRANSACTION

It allows external canceling of a transaction from a different terminal. Starting from version 3.3.0, it is possible to cancel a request even if the "**WaitForUserInput**" state occurs during the progress. So it is possible in states "**WaitForCard**" or "**WaitForUserInput**," otherwise, the request is rejected.

This method is available for:

- payment - /payment , /cbpayment
- preauth - /preauth, /cbpreauth
- refund - /refund, /cbrefund
- moto - /payment/moto, /cbpayment/moto
- scan - /scan
- transport payment - /transport/payment

2.6.1. Request

GSA v1 - POST ip:port/paya/cancel
GSA v2 - POST ip:port/api/switchio/pay/v2/cancel
GSA v4 - POST ip:port/api/switchio/pay/v4/cancel
GSA v5 - POST ip:port/api/switchio/pay/v5/cancel
GSA v6 - POST ip:port/api/switchio/pay/v6/cancel

Request example:

```
{
    "secureString":"password",
    "transactionId":"1e6a1251-762f-45fc-85fb-227a1c3290b6"
}
```

PARAMETER	TYPE	DESCRIPTION
secureString	string	Password of the terminal
transactionId	string	ID of the transaction. Example: *1e6a1251-762f-45fc-85fb-227a1c3290b6*

2.6.2. Response

StatusCode: 200 → Transaction has been canceled, cash register is waiting for **"Finished"** state in **/api/switchio/pay/v2/status**

StatusCode: 404 → Transaction has not been found with given transactionId

StatusCode: 405 → Method is not allowed for transaction

Response example:

```
{
    "message":"OK"
}
```

2.7. DAUGHTER COMPANIES

Obtaining a list of merchants to support the virtualization of the terminal.

2.7.1. Request

GSA v1 - POST ip:port/paya/daughter_companies
GSA v2 - POST ip:port/api/switchio/pay/v2/daughter_companies
GSA v4 - POST ip:port/api/switchio/pay/v4/daughter_companies
GSA v5 - POST ip:port/api/switchio/pay/v5/daughter_companies
GSA v6 - POST ip:port/api/switchio/pay/v6/daughter_companies

Request example:

```
{
    "secureString":"password"
}
```

PARAMETER	TYPE	DESCRIPTION
secureString	string	Password of the terminal

2.7.2. Response

Response example:

```
[
    { "daughterCompanyId":"Kaufland" },
    { "daughterCompanyId":"Tesco" }
]
```

PARAMETER	TYPE	DESCRIPTION
daughterCompanyId	string	ID of the daughter company

2.8. LAST TRANSACTION

The cash register will receive the last authorized transaction by using this request, if there are no transactions on the terminal, response will finish with 404 - Not found.

This endpoint is supported since version 3.0.0

2.8.1. Request

GSA v1 - POST ip:port/paya/last_transaction
GSA v2 - POST ip:port/api/switchio/pay/v2/last_transaction
GSA v4 - POST ip:port/api/switchio/pay/v4/last_transaction
GSA v5 - POST ip:port/api/switchio/pay/v5/last_transaction
GSA v6 - POST ip:port/api/switchio/pay/v6/last_transaction

Request example:

```
{
    "secureString":"password"
}
```

PARAMETER	TYPE	DESCRIPTION
secureString	string	Password of the terminal

2.8.2. Response

For response structure and examples for each version see TRANSACTION RESULT
The response will come as a transaction result.

Response example:

```
{
    "aid": "A0000000283101",
    "amount": 500,
    "aosa": "null",
    "appLabel": "Air Bank",
    "authCode": "114546",
    "availableBalance": "1000",
    "reversal": false,
    "cardInputType": "CLESS",
    "conto": "MASTERCARD",
    "currency": 203,
    "cvmTypeList": ["PIN_ONLINE"],
    "dateTimeServer": "Jul 16, 2020 14:40:50",
    "dateTimeTerminal": "Jul 16, 2020 14:39:37",
    "emvResponseData": {
        "smartCardScheme": "1",
        "authorisationResponseCode": "00"
```

```
    },
    "expiration": "****",
    "invoiceNumber": "VS",
    "isReversed":false,
    "merchantIdAcquirer": "TNEXGO01",
    "merchantIdIssuer": "1357",
    "monetToken": "MONETTOKEN",
    "cardToken": "CARDTOKEN",
    "pan": "970348******3910",
    "par": "PAR",
    "panSequenceNumber": "00",
    "responseCode": "OK",
    "responseMessage": "potvrzeno",
    "reversal": false,
    "sequenceNumber": "001001614",
    "terminalIdAcquirer": "M1TNEXGO01",
    "terminalIdIssuer": "M1TNEXGO01",
    "transactionId": "48a3ea7d-b82f-4b42-abbb-5043365e4faa",
    "transactionType": "SALE_ONLINE",
    "tokens": ["fkdfkjbekjrhljb" ]  //default - empty list
}
```

2.9. GET SUCCESSFUL TRANSACTION

The request is used for receiving all successful transactions on the terminal or empty list when no transactions are available.
This endpoint is supported since version 3.0.0.

2.9.1. Request

GSA v1 - POST ip:port/paya/transactions
GSA v2 - POST ip:port/api/switchio/pay/v2/transactions
GSA v4 - POST ip:port/api/switchio/pay/v4/transactions
GSA v5 - POST ip:port/api/switchio/pay/v5/transactions
GSA v6 - POST ip:port/api/switchio/pay/v6/transactions

Request example:

```
{
    "secureString":"password"
}
```

PARAMETER	TYPE	DESCRIPTION
secureString	string	Password of the terminal

2.9.2. Response

For response structure and examples for each version see TRANSACTION RESULT
The response will come as a list of transaction results (link above).

Response example:

```
[
 {transaction result 1}, {transaction result 2}, {transaction result 3}
]
```

2.10. TRANSACTION RESULT

2.10.1. Request

GSA v1 - POST ip:port/paya/result
GSA v2 - POST ip:port/api/switchio/pay/v2/result
GSA v4 - POST ip:port/api/switchio/pay/v4/result
GSA v5 - POST ip:port/api/switchio/pay/v5/result
GSA v6 - POST ip:port/api/switchio/pay/v6/result

The request has to be called for obtaining information about a transaction. By calling **/result** endpoint transaction flow is not considered fully finished for some types of transactions. For transactions that need to be confirmed you have to call **/confirm** to finish the process otherwise the transaction will be automatically reversed by SwitchioPay.

New parameters added in v5: par, monetToken, cardToken

New parameters added in v6: dccOffer, dccResult, surcharge, externalTransactionId

Request example:

```
{
    "transactionId":"1e6a1251-762f-45fc-85fb-227a1c3290b6",
    "secureString":"password"
}
```

PARAMETER	TYPE	DESCRIPTION
transactionId	string	ID of the transaction. Example: *1e6a1251-762f-45fc-85fb-227a1c3290b6*
secureString	string	Password of the terminal

2.11. Response v1 - transaction with 200 OK

Response example:

```
@Serializable
data class PayaTransactionResultV1(
    var transactionId: String? = UUID.randomUUID().toString(),
    var transactionType: TransactionType?,
    var aid: String? = null,
    var amount: Int? = null,
    var appLabel: String? = null,
    var authCode: String? = null,
    var reversal: Boolean?,
    var conto: String? = null,
    var currency: Int? = null,
    @Serializable(DateSerializer::class) var dateTimeTerminal: Date?,
    @Serializable(DateSerializer::class) var dateTimeServer: Date? = null,
    var emvResponseData: EmvResponseData? = null,
    var expiration: String? = null,
    var invoiceNumber: String? = null,
```

```
    var responseCode: TerminalErrorCodes?,
    var responseMessage: String?,
    var terminalIdAcquirer: String? = null,
    var terminalIdIssuer: String? = null,
    var merchantIdAcquirer: String? = null,
    var merchantIdIssuer: String? = null,
    var pan: String? = null,
    var token: String? = null,
    var panSequenceNumber: String? = null,
    var aosa: Int? = null,
    var cardHolderMessage: String? = null,
    var merchantMessage: String? = null,
    var availableBalance: String? = null,
    var sequenceNumber: String? = null,
    var cvmTypeList: List<CvmType>? = null,
    var cardInputType: CardInputType? = null,
    var isReversed: Boolean? = false,
    var tipAmount: Int? = null,
)

@Serializer(forClass = Date::class)
class DateSerializer : KSerializer<Date> {
                    override    val    descriptor:    SerialDescriptor    =
PrimitiveSerialDescriptor("Date", STRING)

    private val gson = Gson()

    override fun serialize(encoder: Encoder, obj: Date) =
        encoder.encodeString(gson.toJson(obj))

    override fun deserialize(decoder: Decoder): Date =
        gson.fromJson(decoder.decodeString(), Date::class.java)
}
```

2.11.1. Response v2 - transaction with 200 OK

Response example:

```
{
    "aid": "A0000000283101",
    "amount": 500,
    "aosa": "null",
    "appLabel": "Air Bank",
    "authCode": "114546",
    "availableBalance": "1000",
    "callReversal": false,
    "cardInputType": "CLESS",
    "conto": "MASTERCARD",
    "currency": 203,
    "cvmTypeList": ["PIN_ONLINE"],
    "dateTimeServer": "Jul 16, 2020 14:40:50",
    "dateTimeTerminal": "Jul 16, 2020 14:39:37",
    "emvResponseData": {
        "smartCardScheme": "1",
        "authorisationResponseCode": "00"
    },
    "expiration": "****",
    "invoiceNumber": "VS",
    "isReversed":false,
    "merchantIdAcquirer": "TNEXGO01",
    "merchantIdIssuer": "1357",
    "pan": "970348******3910",
    "panSequenceNumber": "00",
    "responseCode": "OK",
    "responseMessage": "potvrzeno",
    "reversal": false,
    "sequenceNumber": "001001614",
    "terminalIdAcquirer": "M1TNEXGO01",
    "terminalIdIssuer": "M1TNEXGO01",
    "transactionId": "48a3ea7d-b82f-4b42-abbb-5043365e4faa",
    "transactionType": "SALE_ONLINE",
    "tokens": ["fkdfkjbekjrhljb" ]  //default - empty list
}
```

PARAMETER	TYPE	DESCRIPTION
aid	string	Application identifier
amount	int	Amount of the transaction in cents. 500 is 5.00 euro
aosa	string	Available offline sending amount
appLabel	string	Label of application

authCode		string	Code of the authorization
availableBalance		int	Balance of the available funds
callReversal		bool	Information whether the reversal should be called or not.
cardInputType		string	Type of the input. Enum is in the appendix.
conto		string	Card issuer
currency		int	Code of the currency
cvmTypeList		list	Type of the TRX. Enum is in the appendix.
dateTimeServer		string	Date and time on server when, was the result made
dateTimeTerminal		string	Date and time on terminal, when was the result made
emvResponseData		obj	Object of the response.
	smartCardScheme	int	Scheme of the smart card.
	authorisationResponseCode	string	The response code of authorization.
expiration		string	Expiration date of the card.
invoiceNumber		string	Variable symbol, max length is 20 char
isReversed		bool	Information whether the transaction was reversed.
merchantIdAcquirer		string	ID of the merchant's acquirer.
merchantIdIssuer		string	ID of the merchant's issuer
pan		string	Primary account number
panSequenceNumber		string	Sequence number of the primary account number
responseCode		string	Code of the response
responseMessage		string	Message of the response
reversal		bool	Information if the transaction is a reversal or not
sequenceNumber		int	Number of the sequence
terminalIdAcquirer		string	ID of the acquirer's terminal
terminalIdIssuer		string	ID of the issuer's terminal
transactionId		string	ID of the transaction. Example: *1e6a1251-762f-45fc-85fb-227a1c3290b6*
transactionType		string	Type of the transaction. Enum is in the appendix.
tokens		string	Encrypted combination of data

2.11.2. Response v2 - closing with 200 OK

Response example:


```
{
    "batchTotals":  {
                        "adjsAmount":0,
                        "adjsCount":0,
                        "batchNumber":11,
                        "creditsAmount":10,
                        "creditsCount":0,
                        "debitsAmount":0,
                        "debitsCount":0,
                        "shiftNumber":1
                    },
    "dateTimeServer":"Jul 9, 2020 10:31:38 AM",
    "dateTimeTerminal":"Jul 9, 2020 10:31:50 AM",
    "responseCode":"OK",
    "responseMessage":"Soucty sum souhlasi",
    "transactionType":"CLOSE_BATCH"
}
```


PARAMETER		TYPE	DESCRIPTION
batchTotals		obj	Object of the batch totals
	adjsAmount	int	Amount of adjustments
	adjsCount	int	Count of adjustments
	batchNumber	int	Number of batch
	creditsAmount	int	Amount of refund
	creditsCount	int	Count of refund TRX
	debitsAmount	int	Amount of payment TRX
	debitsCount	int	Count of payment TRX
	shiftNumber	int	number of shift
dateTimeServer		string	Date and time on server when, was the result made
dateTimeTerminal		string	Date and time on terminal, when was the result made
responseCode		string	Code of the response
responseMessage		string	Message of the response
transactionType		string	Type of the TRX. Enum is in the appendix.

2.11.3. Response v2 - != 200

Response example:

```
{
    "message":"Unauthorized"
}
```

Transaction result v2

```
@Serializable
data class PayaTransactionResultV2(
    var transactionId: String? = UUID.randomUUID().toString(),
    var transactionType: TransactionType?,
    var aid: String? = null,
    var amount: Int? = null,
    var appLabel: String? = null,
    var authCode: String? = null,
    var reversal: Boolean?,
    var conto: String? = null,
    var currency: Int? = null,
    var dateTimeTerminal: String?,
    var dateTimeServer: String? = null,
    var emvResponseData: EmvResponseData? = null,
    var expiration: String? = null,
    var invoiceNumber: String? = null,
    var responseCode: TerminalErrorCodes?,
    var responseMessage: String?,
    var terminalIdAcquirer: String? = null,
    var terminalIdIssuer: String? = null,
    var merchantIdAcquirer: String? = null,
    var merchantIdIssuer: String? = null,
    var pan: String? = null,
    var tokens: List<String> = listOf(),
    var panSequenceNumber: String? = null,
    var aosa: Int? = null,
    var cardHolderMessage: String? = null,
    var merchantMessage: String? = null,
    var availableBalance: String? = null,
    var sequenceNumber: String? = null,
    var cvmTypeList: List<CvmType>? = null,
    var cardInputType: CardInputType? = null,
    var isReversed: Boolean? = false,
    var tipAmount: Int? = null,
    var refundId: String? = null,
)

@Serializable
class EmvResponseData {
    var smartCardScheme: Int? = null
    var authorisationResponseCode: String? = null
    var issuerAuthenticationData: String? = null
```

```
}
```

2.11.4. Response v4 - transaction with 200 OK - SwitchioPAY 3.4.0

```
@Serializable
data class PayaTransactionResultV4(
var transactionId: String? = UUID.randomUUID().toString(),
var transactionType: TransactionType?,
var aid: String? = null,
var amount: Int? = null,
var appLabel: String? = null,
var authCode: String? = null,
var reversal: Boolean?,
var brand: String? = null,
var currency: Int? = null,
var dateTimeTerminal: String?,
var dateTimeServer: String? = null,
var emvResponseData: EmvResponseData? = null,
var expiration: String? = null,
var invoiceNumber: String? = null,
var responseCode: TerminalErrorCodes?,
var responseMessage: String?,
var terminalIdAcquirer: String? = null,
var terminalIdIssuer: String? = null,
var merchantIdAcquirer: String? = null,
var merchantIdIssuer: String? = null,
var pan: String? = null,
var tokens: List<String> = listOf(),
var panSequenceNumber: String? = null,
var aosa: Int? = null,
var cardHolderMessage: String? = null,
var merchantMessage: String? = null,
var availableBalance: String? = null,
var sequenceNumber: String? = null,
var cvmTypeList: List<CvmType>? = null,
var cardInputType: CardInputType? = null,
var isReversed: Boolean? = false,
var tipAmount: Int? = null,
var refundId: String? = null,
)
```

2.11.5. Response v5 - transaction with 200 OK - SwitchioPAY 4.1.0

Response example:

```
{
    "aid": "A0000000283101",
    "amount": 500,
    "aosa": "null",
    "appLabel": "Air Bank",
    "authCode": "114546",
    "availableBalance": "1000",
    "reversal": false,
    "cardInputType": "CLESS",
    "conto": "MASTERCARD",
    "currency": 203,
    "cvmTypeList": ["PIN_ONLINE"],
    "dateTimeServer": "Jul 16, 2020 14:40:50",
    "dateTimeTerminal": "Jul 16, 2020 14:39:37",
    "emvResponseData": {
        "smartCardScheme": "1",
        "authorisationResponseCode": "00"
    },
    "expiration": "****",
    "invoiceNumber": "VS",
    "isReversed":false,
    "merchantIdAcquirer": "TNEXGO01",
    "merchantIdIssuer": "1357",
    "monetToken": "MONETTOKEN",
    "cardToken": "CARDTOKEN",
    "pan": "970348******3910",
    "par": "PAR",
    "panSequenceNumber": "00",
    "responseCode": "OK",
    "responseMessage": "potvrzeno",
    "reversal": false,
    "sequenceNumber": "001001614",
    "terminalIdAcquirer": "M1TNEXGO01",
    "terminalIdIssuer": "M1TNEXGO01",
    "transactionId": "48a3ea7d-b82f-4b42-abbb-5043365e4faa",
    "transactionType": "SALE_ONLINE",
    "tokens": ["fkdfkjbekjrhljb" ]  //default - empty list
}
```

PARAMETER	TYPE	DESCRIPTION
aid	string	Application identifier
amount	int	Amount of the transaction in cents. 500 is 5.00 euro
aosa	string	Available offline sending amount

appLabel		string	Label of application
authCode		string	Code of the authorization
availableBalance		int	Balance of the available funds
callReversal		bool	Information whether the reversal should be called or not.
cardInputType		string	Type of the input. Enum is in the appendix.
conto		string	Card issuer
currency		int	Code of the currency
cvmTypeList		list	Type of the TRX. Enum is in the appendix.
dateTimeServer		string	Date and time on server when, was the result made
dateTimeTerminal		string	Date and time on terminal, when was the result made
emvResponseData		obj	Object of the response.
	smartCardScheme	int	Scheme of the smart card.
	authorisationResponseCode	string	The response code of authorization.
expiration		string	Expiration date of the card.
invoiceNumber		string	Variable symbol, max length is 20 char
isReversed		bool	Information whether the transaction was reversed.
merchantIdAcquirer		string	ID of the merchant's acquirer.
merchantIdIssuer		string	ID of the merchant's issuer
monetToken		string	Monet tokento v CloudpOSu ni
cardToken		string	Card token
pan		string	Primary account number (14-19 characters, card ID)
par		string	Payment account reference (29 characters. It is used to link a payment account represented by the PAN to affiliated payment tokens.)
panSequenceNumber		string	Sequence number of the primary account number
responseCode		string	Code of the response
responseMessage		string	Message of the response
reversal		bool	Information if the transaction is a reversal or not
sequenceNumber		int	Number of the sequence
terminalIdAcquirer		string	ID of the acquirer's terminal

terminalIdIssuer	string	ID of the issuer's terminal
transactionId	string	ID of the transaction. Example: *1e6a1251-762f-45fc-85fb-227a1c3290b6*
transactionType	string	Type of the transaction. Enum is in the appendix.
tokens	string	Encrypted combination of data

2.11.6. Response v5 - != 200 - 4.1.0

Response example:

```
{
    "message":"Unauthorized"
}
```

Data structure

```
@Serializable
data class PayaTransactionResultV5(
    var transactionId: String? = UUID.randomUUID().toString(),
    var transactionType: TransactionType?,
    var aid: String? = null,
    var amount: Int? = null,
    var appLabel: String? = null,
    var authCode: String? = null,
    var reversal: Boolean?,
    var brand: String? = null,
    var currency: Int? = null,
    var dateTimeTerminal: String?,
    var dateTimeServer: String? = null,
    var emvResponseData: EmvResponseData? = null,
    var expiration: String? = null,
    var invoiceNumber: String? = null,
    var responseCode: TerminalErrorCodes?,
    var responseMessage: String?,
    var terminalIdAcquirer: String? = null,
    var terminalIdIssuer: String? = null,
    var merchantIdAcquirer: String? = null,
    var merchantIdIssuer: String? = null,
    var pan: String? = null,
    var tokens: List<String> = listOf(),
    var panSequenceNumber: String? = null,
    var aosa: Int? = null,
    var cardHolderMessage: String? = null,
    var merchantMessage: String? = null,
    var availableBalance: String? = null,
    var sequenceNumber: String? = null,
    var cvmTypeList: List<CvmType>? = null,
    var cardInputType: CardInputType? = null,
```

```
    var isReversed: Boolean? = false,
    var tipAmount: Int? = null,
    var refundId: String? = null,
    var par: String? = null,
    var monetToken: String? = null,
    var cardToken: String? = null,
)

@Serializable
class EmvResponseData {
    var smartCardScheme: Int? = null
    var authorisationResponseCode: String? = null
    var issuerAuthenticationData: String? = null
}
```

2.11.7. Response v6

Response example:

```
{
  "aid": "A0000000283101",
  "amount": 500,
  "aosa": "null",
  "appLabel": "Air Bank",
  "authCode": "114546",
  "availableBalance": "1000",
  "brand": "MASTERCARD",
  "cardHolderMessage": "Approved",
  "cardInputType": "CLESS",
  "currency": 203,
  "cvmTypeList": [
    "PIN_ONLINE"
  ],
  "dateTimeServer": "Jul 16, 2020 14:40:50",
  "dateTimeTerminal": "Jul 16, 2020 14:39:37",
  "dccOffer": {
    "type": "VISA",
    "currency": "USD",
    "amountExponent": 2,
    "convertedAmount": 11556691,
    "rateExponent": 6,
    "rate": 1155680,
    "language": "us",
    "ecbRateExponent": 0,
    "ecbRate": 0,
    "markup": 375,
    "currencyCode": 840
  },
  "dccResult": "PERFORMED",
```

```
  "emvResponseData": {
    "smartCardScheme": "1",
    "authorisationResponseCode": "00"
  },
  "expiration": "****",
  "externalTransactionId": "5043365e4faa-48a3ea7d-1357",
  "invoiceNumber": "VS",
  "isReversed": false,
  "merchantIdAcquirer": "TNEXGO01",
  "merchantIdIssuer": "1357",
  "merchantMessage": "Approved",
  "monetToken": "MONETTOKEN",
  "pan": "970348******3910",
  "par": "PAR",
  "panSequenceNumber": "00",
  "refundId": "1234567890",
  "responseCode": "OK",
  "responseMessage": "potvrzeno",
  "reversal": false,
  "sequenceNumber": "001001614",
  "surcharge": 120,
  "tipAmount": 100,
  "terminalIdAcquirer": "M1TNEXGO01",
  "terminalIdIssuer": "M1TNEXGO01",
  "tokens": ["fkdfkjbekjrhljb"],
  "transactionId": "48a3ea7d-b82f-4b42-abbb-5043365e4faa",
  "transactionType": "SALE_ONLINE"
}
```


Data structure

```
@Serializable
data class PayaTransactionResultV6(
    var transactionId: String? = UUID.randomUUID().toString(),
    var transactionType: TransactionType?,
    var aid: String? = null,
    var amount: Int? = null,
    var appLabel: String? = null,
    var authCode: String? = null,
    var reversal: Boolean?,
    var brand: String? = null,
    var currency: Int? = null,
    var dateTimeTerminal: String?,
    var dateTimeServer: String? = null,
    var emvResponseData: EmvResponseData? = null,
    var expiration: String? = null,
    var invoiceNumber: String? = null,
    var responseCode: TerminalErrorCodes?,
```

```
    var responseMessage: String?,
    var terminalIdAcquirer: String? = null,
    var terminalIdIssuer: String? = null,
    var merchantIdAcquirer: String? = null,
    var merchantIdIssuer: String? = null,
    var pan: String? = null,
    var tokens: List<String> = listOf(),
    var panSequenceNumber: String? = null,
    var aosa: Int? = null,
    var cardHolderMessage: String? = null,
    var merchantMessage: String? = null,
    var availableBalance: String? = null,
    var sequenceNumber: String? = null,
    var cvmTypeList: List<CvmType>? = null,
    var cardInputType: CardInputType? = null,
    var isReversed: Boolean? = false,
    var tipAmount: Int? = null,
    var refundId: String? = null,
    var par: String? = null,
    var monetToken: String? = null,
    var cardToken: String? = null,
    var dccOffer: DccOffer? = null,
    var dccResult: DccResult? = null,
    var surcharge: Int? = null,
    var externalTransactionId: String? = null,
)
```

Data structure - DccOffer, DccType and DccResult objects:

```
@Serializable
data class DccOffer(
    val type: DccType,
    val currency: String,
    val amountExponent: Int,
    val convertedAmount: Long,
    val rateExponent: Int,
    val rate: Long,
    val language: String,
    val ecbRateExponent: Int,
    val ecbRate: Long,
    val markup: Int,
    val currencyCode: Int,
)

enum class DccType {
    VISA,
    MASTERCARD,
}
```

```
enum class DccResult {
    PERFORMED,
    NOT_PERFORMED,
}
```

PARAMETER	TYPE	DESCRIPTION
aid	string	Application identifier
amount	int	Amount of the transaction in cents: 500 is 5.00 euro. It is the total amount that includes the surcharge amount as well as the tip amount (if applied on the transaction).
aosa	string	Available offline sending amount
appLabel	string	Label of application
authCode	string	Code of the authorization
availableBalance	int	Balance of the available funds
callReversal	bool	Information whether the reversal should be called or not.
cardInputType	string	Type of the input. Enum is in the appendix.
conto	string	Card issuer
currency	int	Code of the currency
cvmTypeList	list	Type of the TRX. Enum is in the appendix.
dateTimeServer	string	Date and time on server when, was the result made
dateTimeTerminal	string	Date and time on terminal, when was the result made
emvResponseData	obj	Object of the response.
smartCardScheme	int	Scheme of the smart card.
authorisationResponseCode	string	The response code of authorization.
expiration	string	Expiration date of the card.
invoiceNumber	string	Variable symbol, max length is 20 char
isReversed	bool	Information whether the transaction was reversed.
merchantIdAcquirer	string	ID of the merchant's acquirer.
merchantIdIssuer	string	ID of the merchant's issuer
monetToken	string	Monet token
cardToken	string	Card token

pan	string	Primary account number (14-19 characters, card ID)
par	string	Payment account reference (29 characters. It is used to link a payment account represented by the PAN to affiliated payment tokens.)
panSequenceNumber	string	Sequence number of the primary account number
responseCode	string	Code of the response
responseMessage	string	Message of the response
reversal	bool	Information if the transaction is a reversal or not
sequenceNumber	int	Number of the sequence
terminalIdAcquirer	string	ID of the acquirer's terminal
terminalIdIssuer	string	ID of the issuer's terminal
transactionId	string	ID of the transaction. Example: *1e6a1251-762f-45fc-85fb-227a1c3290b6*
transactionType	string	Type of the transaction. Enum is in the appendix.
tokens	string	Encrypted combination of data
dccOffer	obj	DCC Offer object
type	enum	Card type. Possible values: • VISA • MASTERCARD • UNKNOWN
currency	string	Name of the DCC currency
amountExponent	int	Amount exponent. Indicates how many decimal places the amount is calculated.
convertedAmount	long	Converted amount
rateExponent	int	DCC rate exponent. Indicates how many decimal places the currency is calculated.
rate	long	Conversion rate
language	string	Language code
ecbRateExponent	int	European Central Bank rate exponent. Indicates how many decimal places the currency is calculated.
ecbRate	long	European Central Bank rate.
markup	int	Markup amount
currencyCode	int	Code of the DCC currency
dccResult	enum	DccResult object. Contains information about the result

		of the DCC offer - possible values: • PERFORMED - DCC offer was accepted and DCC was performed • NOT_PERFORMED - DCC was not performed
surcharge	int	Surcharged amount
externalTransactionId	string	Transaction ID in an external system

2.11.8. Response v7

Response example:


```
{
    "aid": "A0000000283101",
    "amount": 500,
    "cashbackAmount": 1500,
    "aosa": "null",
    "appCryptogram": "5F52BA71C7B45441",
    "appLabel": "Air Bank",
    "authCode": "114546",
    "availableBalance": "1000",
    "brand": "MASTERCARD",
    "cardHolderMessage": "Approved",
    "cardInputType": "CLESS",
    "cid": "80",
    "currency": 203,
    "cvmTypeList": ["PIN_ONLINE"],
    "dateTimeServer": "Jul 16, 2020 14:40:50",
    "dateTimeTerminal": "Jul 16, 2020 14:39:37",
    "driverNumber": "123456",
    "emvResponseData": {
        "smartCardScheme": "1",
        "authorisationResponseCode": "00"
    },
    "expiration": "****",
    "invoiceNumber": "VS",
    "isReversed":false,
    "merchantIdAcquirer": "TNEXGO01",
    "merchantIdIssuer": "1357",
    "merchantMessage": "Approved",
    "monetToken": "MONETTOKEN",
    "odometer": 65403,
    "pan": "970348******3910",
    "par": "PAR",
    "panSequenceNumber": "00",
    "refundId": "1234567890",
    "responseCode": "OK",
    "responseMessage": "potvrzeno",
    "reversal": false,
```

```
    "sequenceNumber": "001001614",
    "tipAmount": 100,
    "terminalIdAcquirer": "M1TNEXGO01",
    "terminalIdIssuer": "M1TNEXGO01",
    "tokens": ["fkdfkjbekjrhljb"]  //default - empty list
    "transactionId": "48a3ea7d-b82f-4b42-abbb-5043365e4faa",
    "transactionType": "SALE_ONLINE",
    "vehicleCode": "1A21234",
}
```

Data structure:

```
@Serializable
data class PayaTransactionResultV7(
    val transactionId: String? = UUID.randomUUID().toString(),
    val transactionType: TransactionType?,
    val aid: String? = null,
    val amount: Int? = null,
    val cashbackAmount: Int? = null,
    val appLabel: String? = null,
    val authCode: String? = null,
    val reversal: Boolean?,
    val brand: String? = null,
    val currency: Int? = null,
    val dateTimeTerminal: String?,
    val dateTimeServer: String? = null,
    val emvResponseData: EmvResponseData? = null,
    val expiration: String? = null,
    val invoiceNumber: String? = null,
    val responseCode: TerminalErrorCodes?,
    val responseMessage: String?,
    val terminalIdAcquirer: String? = null,
    val terminalIdIssuer: String? = null,
    val merchantIdAcquirer: String? = null,
    val merchantIdIssuer: String? = null,
    val pan: String? = null,
    val tokens: List<String> = listOf(),
    val panSequenceNumber: String? = null,
    val aosa: Int? = null,
    val cardHolderMessage: String? = null,
    val merchantMessage: String? = null,
    val availableBalance: String? = null,
    val sequenceNumber: String? = null,
    val cvmTypeList: List<CvmType>? = null,
    val cardInputType: CardInputType? = null,
    val isReversed: Boolean? = false,
    val tipAmount: Int? = null,
    val refundId: String? = null,
    val par: String? = null,
```

```
    val monetToken: String? = null,
    val cardToken: String? = null,
    val dccOffer: DccOffer? = null,
    val dccResult: DccResult? = null,
    val surcharge: Int? = null,
    val externalTransactionId: String? = null,
    val odometer: Long? = null,
    val vehicleCode: String? = null,
    val driverNumber: String? = null,
    val cid: String? = null,
    val appCryptogram: String? = null,
    val authorizationResponseCode: String? = null,
)
```

PARAMETER	TYPE	DESCRIPTION
aid	string	Application identifier
amount	int	Amount of the transaction in cents: 500 is 5.00 euro. It is the total amount that includes the surcharge amount as well as the tip amount (if applied on the transaction).
aosa	string	Available offline sending amount
appCryptogram	string	Cryptogram returned by the ICC in response of the GENERATE AC command.
appLabel	string	Label of application
authCode	string	Code of the authorization
availableBalance	int	Balance of the available funds
callReversal	bool	Information whether the reversal should be called or not.
cardInputType	string	Type of the input. Enum is in the appendix.
cid	string	Values: 00 (AAC)-Application Authentication. Cryptogram-A cryptogram generated by the card at the end of offline and online declined transactions 40(TC)-Transaction Certificate-the result of card, terminal, and transaction data encrypted by a DES key 80 (ARQC)-Authorization Request. Cryptogram-A cryptogram used for a process called Online Card Authentication.
conto	string	Card issuer
currency	int	Code of the currency

cvmTypeList	list	Type of the TRX. Enum is in the appendix.
dateTimeServer	string	Date and time on server when, was the result made
dateTimeTerminal	string	Date and time on terminal, when was the result made
driverNumber	string	ID number of the driver.
emvResponseData	obj	Object of the response.
smartCardScheme	int	Scheme of the smart card.
authorisationResponseCode	string	The response code of authorization.
expiration	string	Expiration date of the card.
invoiceNumber	string	Variable symbol, max length is 20 char
isReversed	bool	Information whether the transaction was reversed.
merchantIdAcquirer	string	ID of the merchant's acquirer.
merchantIdIssuer	string	ID of the merchant's issuer
monetToken	string	Monet token
cardToken	string	Card token
pan	string	Primary account number (14-19 characters, card ID)
par	string	Payment account reference (29 characters. It is used to link a payment account represented by the PAN to affiliated payment tokens.)
panSequenceNumber	string	Sequence number of the primary account number
responseCode	string	Code of the response
responseMessage	string	Message of the response
reversal	bool	Information if the transaction is a reversal or not
sequenceNumber	int	Number of the sequence
terminalIdAcquirer	string	ID of the acquirer's terminal
terminalIdIssuer	string	ID of the issuer's terminal
transactionId	string	ID of the transaction. Example: *1e6a1251-762f-45fc-85fb-227a1c3290b6*
transactionType	string	Type of the transaction. Enum is in the appendix.
tokens	string	Encrypted combination of data
vehicleCode	string	Code for vehicle.
dccOffer	obj	DCC Offer object

	type	enum	Card type. Possible values: • VISA • MASTERCARD • UNKNOWN
	currency	string	Name of the DCC currency
	amountExponent	int	Amount exponent. Indicates how many decimal places the amount is calculated.
	convertedAmount	long	Converted amount
	rateExponent	int	DCC rate exponent. Indicates how many decimal places the currency is calculated.
	rate	long	Conversion rate
	language	string	Language code
	ecbRateExponent	int	European Central Bank rate exponent. Indicates how many decimal places the currency is calculated.
	ecbRate	long	European Central Bank rate.
	markup	int	Markup amount
	currencyCode	int	Code of the DCC currency
dccResult		enum	DccResult object. Contains information about the result of the DCC offer - possible values: • PERFORMED - DCC offer was accepted and DCC was performed • NOT_PERFORMED - DCC was not performed
surcharge		int	Surcharged amount
externalTransactionId		string	Transaction ID in an external system
odometer		Long	OPTIONAL Represents the vehicle's mileage at the time of the transaction, measured in kilometers
vehicleCode		String	OPTIONAL Identifies the specific vehicle within a fleet, typically represented as a unique number or alphanumeric code.
driverNumber		String	OPTIONAL A unique identifier assigned to the driver operating the vehicle during the transaction.
licensePlate		String	OPTIONAL The official registration number of the vehicle, typically displayed on the vehicle's plates.
authorizationResponseCode		String	A two-character code returned by the issuer that indicates the result of the authorization request.

2.11.9. Response v8

Response example:


```
{
    "aid": "A0000000283101",
    "amount": 500,
    "cashbackAmount": 1500,
    "aosa": "null",
    "appCryptogram": "5F52BA71C7B45441",
    "appLabel": "Air Bank",
    "authCode": "114546",
    "availableBalance": "1000",
    "brand": "MASTERCARD",
    "cardHolderMessage": "Approved",
    "cardInputType": "CLESS",
    "cid": "80",
    "currency": 203,
    "cvmTypeList": ["PIN_ONLINE"],
    "dateTimeServer": "Jul 16, 2020 14:40:50",
    "dateTimeTerminal": "Jul 16, 2020 14:39:37",
    "driverNumber": "123456",
    "emvResponseData": {
        "smartCardScheme": "1",
        "authorisationResponseCode": "00"
    },
    "expiration": "****",
    "invoiceNumber": "VS",
    "isReversed":false,
    "merchantIdAcquirer": "TNEXGO01",
    "merchantIdIssuer": "1357",
    "merchantMessage": "Approved",
    "monetToken": "MONETTOKEN",
    "odometer": 65403,
    "pan": "970348******3910",
    "par": "PAR",
    "panSequenceNumber": "00",
    "refundId": "1234567890",
    "responseCode": "OK",
    "responseMessage": "potvrzeno",
    "reversal": false,
    "sequenceNumber": "001001614",
    "tipAmount": 100,
    "terminalIdAcquirer": "M1TNEXGO01",
    "terminalIdIssuer": "M1TNEXGO01",
    "tokens": ["fkdfkjbekjrhljb"]  //default - empty list
    "transactionId": "48a3ea7d-b82f-4b42-abbb-5043365e4faa",
    "transactionType": "SALE_ONLINE",
    "vehicleCode": "1A21234",
```

```
    "driverNumber": "DRIVER1",
    "licensePlate": "EL100AC",
    "authorizationResponseCode": "00",
    "transferId": "split-e529f8b4790f11e7b5a5be2e44b06b52",
    "additionalFeeAmount": 200,
}
```

Data structure:

```
@Serializable
data class PayaTransactionResultV8(
    val transactionId: String? = UUID.randomUUID().toString(),
    val transactionType: TransactionType?,
    val aid: String? = null,
    val amount: Int? = null,
    val cashbackAmount: Int? = null,
    val appLabel: String? = null,
    val authCode: String? = null,
    val reversal: Boolean?,
    val brand: String? = null,
    val currency: Int? = null,
    val dateTimeTerminal: String?,
    val dateTimeServer: String? = null,
    val emvResponseData: EmvResponseData? = null,
    val expiration: String? = null,
    val invoiceNumber: String? = null,
    val responseCode: TerminalErrorCodes?,
    val responseMessage: String?,
    val terminalIdAcquirer: String? = null,
    val terminalIdIssuer: String? = null,
    val merchantIdAcquirer: String? = null,
    val merchantIdIssuer: String? = null,
    val pan: String? = null,
    val tokens: List<String> = listOf(),
    val panSequenceNumber: String? = null,
    val aosa: Int? = null,
    val cardHolderMessage: String? = null,
    val merchantMessage: String? = null,
    val availableBalance: String? = null,
    val sequenceNumber: String? = null,
    val cvmTypeList: List<CvmType>? = null,
    val cardInputType: CardInputType? = null,
    val isReversed: Boolean? = false,
    val tipAmount: Int? = null,
    val refundId: String? = null,
    val par: String? = null,
    val monetToken: String? = null,
    val cardToken: String? = null,
    val dccOffer: DccOffer? = null,
```

```
    val dccResult: DccResult? = null,
    val surcharge: Int? = null,
    val externalTransactionId: String? = null,
    val odometer: Long? = null,
    val vehicleCode: String? = null,
    val driverNumber: String? = null,
    val cid: String? = null,
    val appCryptogram: String? = null,
    val authorizationResponseCode: String? = null,
    val transferId: String? = null,
    val additionalFeeAmount: Int? = null,
)
```

PARAMETER	TYPE	DESCRIPTION
aid	string	Application identifier
amount	int	Amount of the transaction in cents: 500 is 5.00 euro. It is the total amount that includes the surcharge amount as well as the tip amount (if applied on the transaction).
aosa	string	Available offline sending amount
appCryptogram	string	Cryptogram returned by the ICC in response of the GENERATE AC command.
appLabel	string	Label of application
authCode	string	Code of the authorization
availableBalance	int	Balance of the available funds
callReversal	bool	Information whether the reversal should be called or not.
cardInputType	string	Type of the input. Enum is in the appendix.
cid	string	Values: 00 (AAC)-Application Authentication. Cryptogram-A cryptogram generated by the card at the end of offline and online declined transactions 40(TC)-Transaction Certificate-the result of card, terminal, and transaction data encrypted by a DES key 80 (ARQC)-Authorization Request. Cryptogram-A cryptogram used for a process called Online Card Authentication.
conto	string	Card issuer
currency	int	Code of the currency

cvmTypeList	list	Type of the TRX. Enum is in the appendix.
dateTimeServer	string	Date and time on server when, was the result made
dateTimeTerminal	string	Date and time on terminal, when was the result made
driverNumber	string	ID number of the driver.
emvResponseData	obj	Object of the response.
smartCardScheme	int	Scheme of the smart card.
authorisationResponseCode	string	The response code of authorization.
expiration	string	Expiration date of the card.
invoiceNumber	string	Variable symbol, max length is 20 char
isReversed	bool	Information whether the transaction was reversed.
merchantIdAcquirer	string	ID of the merchant's acquirer.
merchantIdIssuer	string	ID of the merchant's issuer
monetToken	string	Monet token
cardToken	string	Card token
pan	string	Primary account number (14-19 characters, card ID)
par	string	Payment account reference (29 characters. It is used to link a payment account represented by the PAN to affiliated payment tokens.)
panSequenceNumber	string	Sequence number of the primary account number
responseCode	string	Code of the response
responseMessage	string	Message of the response
reversal	bool	Information if the transaction is a reversal or not
sequenceNumber	int	Number of the sequence
terminalIdAcquirer	string	ID of the acquirer's terminal
terminalIdIssuer	string	ID of the issuer's terminal
transactionId	string	ID of the transaction. Example: *1e6a1251-762f-45fc-85fb-227a1c3290b6*
transactionType	string	Type of the transaction. Enum is in the appendix.
tokens	string	Encrypted combination of data
vehicleCode	string	Code for vehicle.
dccOffer	obj	DCC Offer object

	type	enum	Card type. Possible values: • VISA • MASTERCARD • UNKNOWN
	currency	string	Name of the DCC currency
	amountExponent	int	Amount exponent. Indicates how many decimal places the amount is calculated.
	convertedAmount	long	Converted amount
	rateExponent	int	DCC rate exponent. Indicates how many decimal places the currency is calculated.
	rate	long	Conversion rate
	language	string	Language code
	ecbRateExponent	int	European Central Bank rate exponent. Indicates how many decimal places the currency is calculated.
	ecbRate	long	European Central Bank rate.
	markup	int	Markup amount
	currencyCode	int	Code of the DCC currency
dccResult		enum	DccResult object. Contains information about the result of the DCC offer - possible values: • PERFORMED - DCC offer was accepted and DCC was performed • NOT_PERFORMED - DCC was not performed
surcharge		int	Surcharged amount
externalTransactionId		string	Transaction ID in an external system
odometer		Long	OPTIONAL Represents the vehicle's mileage at the time of the transaction, measured in kilometers
vehicleCode		String	OPTIONAL Identifies the specific vehicle within a fleet, typically represented as a unique number or alphanumeric code.
driverNumber		String	OPTIONAL A unique identifier assigned to the driver operating the vehicle during the transaction.
licensePlate		String	OPTIONAL The official registration number of the vehicle, typically displayed on the vehicle's plates.
authorizationResponseCode		String	A two-character code returned by the issuer that indicates the result of the authorization request.

transferId	String	OPTIONAL A unique identifier assigned to additional fee payment
additionalFeeAmount	integer	OPTIONAL Amount of the additional fee applied to payment
systemTraceAuditNumber	String	OPTIONAL - RFU This field is not used yet, but will be supported in future releases
offlineAuthentication	String	OPTIONAL - RFU This field is not used yet, but will be supported in future releases
gatewayUniqueTransactionId	String	OPTIONAL - RFU This field is not used yet, but will be supported in future releases

2.12. CARD VERIFY - 3.4.0

Verification of the card, which will authorize the card on the authorization server and the authorization server creates a cardToken, which will be returned to the cash register inside transaction Result.

2.12.1. Request

GSA v4 - POST ip:port/api/switchio/pay/v4/card_verify
GSA v5 - POST ip:port/api/switchio/pay/v5/card_verify
GSA v6 - POST ip:port/api/switchio/pay/v6/card_verify

Request example:

```
{
"secureString":"password",
"transactionId":"1e6a1251-762f-45fc-85fb-227a1c3290b6",
}
```

PARAMETER	TYPE	DESCRIPTION
secureString	string	Password of the terminal
transactionId	string	ID of the transaction. Example: *1e6a1251-762f-45fc-85fb-227a1c3290b6*

2.12.2. Response

StatusCode: 200 → Transaction has been processed, cash register is waiting for **"Finished"** state in **/api/switchio/pay/v4/status**

StatusCode: 202 → Request was processed, but there is some transaction on the terminal that needs to be handled first - server busy.

Response example:

```
{
    "transactionId":"1e6a1251-762f-45fc-85fb-227a1c3290b6",
    "isStarted":false,
    "status":"Other payment in progress"
}
```

PARAMETER	TYPE	DESCRIPTION
transactionId	string	ID of the transaction. Example: *1e6a1251-762f-45fc-85fb-227a1c3290b6*
isStarted	bool	Information whether the transaction started or not.
status	string	Message of the transaction status

StatusCode: 405 → Method is not allowed for transaction

StatusCode: 400 → Transaction with the same transactionId has been already processed - It was already processed, use different transactionId

Response example:

```
{
    "message":"Duplicate Transaction"
}
```

Response example when calling v4/result:

```
@Serializable
data class PayaCardVerifyResultV4(
    var transactionId: String? = UUID.randomUUID().toString(),
    var transactionType: TransactionType?,
    var aid: String? = null,
    var appLabel: String? = null,
    var authCode: String? = null,
    var brand: String? = null,
    var dateTimeTerminal: String?,
    var dateTimeServer: String? = null,
    var expiration: String? = null,
    var responseCode: TerminalErrorCodes?,
    var responseMessage: String?,
    var terminalIdAcquirer: String? = null,
    var terminalIdIssuer: String? = null,
    var merchantIdAcquirer: String? = null,
    var merchantIdIssuer: String? = null,
    var pan: String? = null,
    var panSequenceNumber: String? = null,
    var cardHolderMessage: String? = null,
    var merchantMessage: String? = null,
    var sequenceNumber: String? = null,
    var cvmTypeList: List<CvmType>? = null,
```

```
    var cardInputType: CardInputType? = null,
    var cardToken: String? = null,
    var par: String? = null,
    var transactionReference: String? = null
)
```

Response example when calling v5/result:

```
@Serializable
data class PayaCardVerifyResultV5(
var transactionId: String? = UUID.randomUUID().toString(),
var transactionType: TransactionType?,
var aid: String? = null,
var appLabel: String? = null,
var authCode: String? = null,
var brand: String? = null,
var dateTimeTerminal: String?,
var dateTimeServer: String? = null,
var expiration: String? = null,
var responseCode: TerminalErrorCodes?,
var responseMessage: String?,
var terminalIdAcquirer: String? = null,
var terminalIdIssuer: String? = null,
var merchantIdAcquirer: String? = null,
var merchantIdIssuer: String? = null,
var pan: String? = null,
var panSequenceNumber: String? = null,
var cardHolderMessage: String? = null,
var merchantMessage: String? = null,
var sequenceNumber: String? = null,
var cvmTypeList: List<CvmType>? = null,
var cardInputType: CardInputType? = null,
var cardToken: String? = null,
var par: String? = null,
var transactionReference: String? = null,
)
```

3. RETAIL ENDPOINTS

The communication runs on HTTP standard with REST API interface.
Example of GET request, every line except Body has to have CRLF ending (\r\n).

GET request

```
GET /api/switchio/pay/v2/heartbeat HTTP/1.1\r\n
Host: /\r\n\r\n
```

Example of POST request, every line except Body has to have CRLF ending (`\r\n\r\n`)

POST request

```
POST /api/switchio/pay/v2/terminal HTTP/1.1\r\n
Host: /\r\n
Content-Type: application/json\r\n
Content-Length: 52\r\n\r\n

{"secureString":"123456","checkOnline":true}\r\n\r\n
```

3.1. PAYMENT

The state of the transaction should be queried periodically by calling **/status**. Once the cash register receives a **"Finished"** state from the status request, the app must request the result of the transaction through **/result** request.

The cash register must confirm that the transaction result from **/result** was received correctly by calling **/confirm.** Otherwise, the transaction will be **automatically reversed**. There is a **60 seconds countdown** for a transaction to be confirmed.

3.1.1. Request

GSA v1 - POST ip:port/paya/payment
GSA v2 - POST ip:port/api/switchio/pay/v2/payment
GSA v4 - POST ip:port/api/switchio/pay/v4/payment
GSA v5 - POST ip:port/api/switchio/pay/v5/payment
GSA v6 - POST ip:port/api/switchio/pay/v6/payment
GSA v7 - POST ip:port/api/switchio/pay/v7/payment
GSA v8 - POST ip:port/api/switchio/pay/v8/payment

Request example:

```
{     "secureString":"password",
      "request":{
         "transactionId":"1e6a1251-762f-45fc-85fb-227a1c3290b6",
         "amount":10053,
         "currencyCode":203,
         "invoiceNumber":"",
         "daughterCompanyId":"",
         "additionalFeeAmount":200,
         "tipAmount":2000,
     }
}
```

PARAMETER	TYPE	DESCRIPTION
secureString	string	Password of the terminal
request	obj	Request object
transactionId	string	ID of the transaction. Example: *1e6a1251-762f-45fc-85fb-227a1c3290b6*
amount	int	Amount of the payment, in cents 10053 is 100.53 euro
currencyCode	int	Code of the currency
invoiceNumber	string	OPTIONAL Variable symbol length is 20 char
tipAmount	Int	OPTIONAL Amount of the tip for a waitress
additionalFeeAmount	Int	OPTIONAL Additional fee amount

	daughterCompanyId	string	OPTIONAL ID of the daughter company. Required only when virtualization is enabled.

Request with callback

This endpoint is not available for GSA over Bluetooth.
Cash register calls only **/confirm**. There is no need to call **/status** and **/result**.

GSA v1 - POST ip:port/paya/cbpayment
GSA v2 - POST ip:port/api/switchio/pay/v2/cbpayment
GSA v4 - POST ip:port/api/switchio/pay/v4/cbpayment
GSA v5 - POST ip:port/api/switchio/pay/v5/cbpayment

Request example:

```
{     "secureString":"password",
      "request":{
         "transactionId":"1e6a1251-762f-45fc-85fb-227a1c3290b6",
         "amount":10053,
         "currencyCode":203,
         "invoiceNumber":"",
         "daughterCompanyId":""
     },
     "progressCallback":{
         "ip":"172.25.11.14",
         "port":7777,
         "endpoint":"/gsa/progress"
     },
     "resultCallback":{
         "ip":"172.25.11.14",
         "port":7777,
         "endpoint":"/gsa/result"
     }
}
```

PARAMETER		TYPE	DESCRIPTION
progressCallback		obj	Object of progress to callback
	ip	string	IP to callback
	port	int	Port to callback
	endpoint	string	Endpoint of method to callback
resultCallback		obj	Object of result to callback
	ip	string	IP to callback
	port	int	Port to callback
	endpoint	string	Endpoint of the method to callback

3.1.2. Response

StatusCode: 200 → Transaction has started, the cash register is polling request until it receives the **"Finished"** state.

StatusCode: 202 → Request was processed, but there is some transaction on the terminal that needs to be handled first - server busy.

Response example:

```
{
    "transactionId":"1e6a1251-762f-45fc-85fb-227a1c3290b6",
    "isStarted":false,
    "status":"Other payment in progress"
}
```

PARAMETER	TYPE	DESCRIPTION
transactionId	string	ID of the transaction. Example: *1e6a1251-762f-45fc-85fb-227a1c3290b6*
isStarted	bool	Information whether the payment has started.
status	string	Message of the payment status.

If you send a request with a transaction ID that was already processed, you will get a different response.

StatusCode: 400 → Transaction with the same transactionId has been already processed - It was already processed, use a different transactionId.

StatusCode: 404 → Transaction with the indicated transactionID was not found.

StatusCode: 405 → Method not allowed for the current state of the transaction.

Response example:

```
{
    "message":"Duplicate Transaction"
}
```

3.2. MOTO PAYMENT POLLING

Requesting MOTO payment transaction. Moto payment is a transaction where a user has to insert card information manually on the terminal.

The state of the transaction should be queried periodically by calling **/status**. Once the cash register receives a **"Finished"** state from the status request, the app must request the result of the transaction through **/result** request.

The cash register must confirm that the transaction result from **/result** was received correctly by calling **/confirm.** Otherwise, the transaction will be **automatically reversed**. There is a **60 seconds countdown** for a transaction to be confirmed.

3.2.1. Request v1 (2.5.0 and lower)

GSA v1 - POST ip:port/paya/payment/moto

Request example:

```
{
    "secureString":"password",
    "request":{
        "transactionId":"",
        "cardNumber":"",
        "expiration":"0121",
        "cvc":"455",
        "amount":10053,
        "currencyCode":203,
        "invoiceNumber":"",
        "daughterCompanyId":""
    }
}
```

PARAMETER	TYPE	DESCRIPTION
secureString	string	Password of the terminal
request	obj	Request object
transactionId	string	ID of the transaction. Example: *1e6a1251-762f-45fc-85fb-227a1c3290b6*
cardnumber	string	Card number
expiration	string	Card expiration (MMYY)
cvc	string	Card Verification Code
amount	int	Amount of the payment, in cents 10053 is 100.53 euro
currencyCode	int	Code of the currency

	invoiceNumber	string	Variable symbol length is 20 char
	daughterCompanyId	string	ID of the daughter company. Required only when virtualization is enabled.

3.2.2. Request v2, v4, v5, v6

For request from Switchio Pay 3.0.0, it is not necessary to enter the card number, expiration and CVC - these values are entered by the user on the terminals. JSON is the same as for a normal payment.

GSA v2 - POST ip:port/api/switchio/pay/v2/payment/moto
GSA v4 - POST ip:port/api/switchio/pay/v4/payment/moto
GSA v5 - POST ip:port/api/switchio/pay/v5/payment/moto
GSA v6 - POST ip:port/api/switchio/pay/v6/payment/moto
GSA v7 - POST ip:port/api/switchio/pay/v7/payment/moto
GSA v8 - POST ip:port/api/switchio/pay/v8/payment/moto

Request example:

```
{
    "secureString":"password",
    "request":{
        "transactionId":"1e6a1251-762f-45fc-85fb-227a1c3290b6",
        "amount":10053,
        "currencyCode":203,
        "invoiceNumber":"",
        "daughterCompanyId":""
    }
}
```

PARAMETER	TYPE	DESCRIPTION
secureString	string	Password of the terminal
request	obj	Request object
transactionId	string	ID of the transaction. Example: *1e6a1251-762f-45fc-85fb-227a1c3290b6*
amount	int	Amount of the payment, in cents 10053 is 100.53 euro
currencyCode	int	Code of the currency
invoiceNumber	string	Variable symbol length is 20 char
daughterCompanyId	string	ID of the daughter company. Required only when virtualization is enabled.

3.2.3. Request with callback

This endpoint is not available for GSA over Bluetooth.
Cash register calls only **/confirm**. There is no need to call **/status** and **/result**.

GSA v1 - POST ip:port/paya/cbpayment/moto
GSA v2 - POST ip:port/api/switchio/pay/v2/cbpayment/moto
GSA v4 - POST ip:port/api/switchio/pay/v4/cbpayment/moto
GSA v5 - POST ip:port/api/switchio/pay/v5/cbpayment/moto
GSA v6 - POST ip:port/api/switchio/pay/v6/cbpayment/moto
GSA v7 - POST ip:port/api/switchio/pay/v7/cbpayment/moto
GSA v8 - POST ip:port/api/switchio/pay/v8/cbpayment/moto

Request example:

```
{     "secureString":"password",
      "request":{
         "transactionId":"1e6a1251-762f-45fc-85fb-227a1c3290b6",
         "amount":10053,
         "currencyCode":203,
         "invoiceNumber":"",
         "daughterCompanyId":""
     },
     "progressCallback":{
         "ip":"172.25.11.14",
         "port":7777,
         "endpoint":"/gsa/progress"
     },
     "resultCallback":{
         "ip":"172.25.11.14",
         "port":7777,
         "endpoint":"/gsa/result"
     }
}
```

PARAMETER	TYPE	DESCRIPTION
progressCallback	obj	Object of progress to callback
ip	string	IP to callback
port	int	Port to callback
endpoint	string	Endpoint of method to callback
resultCallback	obj	Object of result to callback
ip	string	IP to callback
port	int	Port to callback
endpoint	string	Endpoint of the method to callback

3.2.4. Response

StatusCode: 200 → Transaction has started, the cash register is polling **/status** request until it receives the **"Finished"** state

StatusCode: 202 → Request was processed, but there is some transaction on the terminal that needs to be handled first - server busy.

Response example:

```
{
    "transactionId":"1e6a1251-762f-45fc-85fb-227a1c3290b6",
    "isStarted":true,
    "status":"Other payment in progress"
}
```

PARAMETER	TYPE	DESCRIPTION
transactionId	string	ID of the transaction. Example: *1e6a1251-762f-45fc-85fb-227a1c3290b6*
isStarted	bool	Information whether the payment has started.
status	string	Message of the payment status.

StatusCode: 400 → Transaction with the same transactionId has been already processed - It was already processed, use different transactionId

Response example:

```
{
    "message":"Duplicate Transaction"
}
```

3.3. MOTO PREAUTHORIZATION

Requesting MOTO Preauthorization transaction. Moto Preauthorization is a transaction where a user has to insert card information manually on the terminal.

The state of the transaction should be queried periodically by calling **/status**. Once the cash register receives a **"Finished"** state from the status request, the app must request the result of the transaction through **/result** request.

The cash register must confirm that the transaction result from **/result** was received correctly by calling **/confirm.** Otherwise, the transaction will be **automatically reversed**. There is a **60 seconds countdown** for a transaction to be confirmed.

This endpoint is available for Switchio Pay version 4.7.0 and higher.

3.3.1. Request

GSA v7 - POST ip:port/api/switchio/pay/v7/payment/moto/preauth
GSA v8 - POST ip:port/api/switchio/pay/v8/payment/moto/preauth

Request example:

```
{
    "secureString":"password",
    "request":{
        "transactionId":"1e6a1251-762f-45fc-85fb-227a1c3290b6",
        "amount":10053,
        "currencyCode":203,
        "invoiceNumber":"",
        "daughterCompanyId":""
    }
}
```

PARAMETER		TYPE	DESCRIPTION
secureString		string	Password of the terminal
request		obj	Request object
	transactionId	string	ID of the transaction. Example: *1e6a1251-762f-45fc-85fb-227a1c3290b6*
	amount	int	Amount of the preauthorization, in cents 10053 is 100.53 euro
	currencyCode	int	Code of the currency
	invoiceNumber	string	Variable symbol length is 20 char
	daughterCompanyId	string	ID of the daughter company. Required only when virtualization is enabled.

3.3.2. Response

StatusCode: 200 → transaction has started, cash register is polling **/status** request until it receives the **"Finished"** state

StatusCode: 202 → Request was processed, but there is some transaction on the terminal that needs to be handled first - server busy.

Response example:

```
{
    "transactionId":"1e6a1251-762f-45fc-85fb-227a1c3290b6",
    "isStarted":false,
    "status":"Other payment in progress"
}
```

PARAMETER	TYPE	DESCRIPTION
transactionId	string	ID of the transaction. Example: *1e6a1251-762f-45fc-85fb-227a1c3290b6*
isStarted	bool	Information whether the preauth completion has started.
status	string	Message of the payment status.

If you send a request with a transaction ID that was already processed, you will get a different response.

3.4. REFUND POLLING

The state of the transaction should be queried periodically by calling **/status**. Once the cash register receives a **"Finished"** state from the status request, the app must request the result of the transaction through **/result** request.

The cash register must confirm that the transaction result from **/result** was received correctly by calling **/confirm.** Otherwise, the transaction will be **automatically reversed**. There is a **60 seconds countdown** for a transaction to be confirmed.

3.4.1. Request

GSA v1 - POST ip:port/paya/refund
GSA v2 - POST ip:port/api/switchio/pay/v2/refund
GSA v4 - POST ip:port/api/switchio/pay/v4/refund
GSA v5 - POST ip:port/api/switchio/pay/v5/refund
GSA v6 - POST ip:port/api/switchio/pay/v6/refund
GSA v7 - POST ip:port/api/switchio/pay/v7/refund
GSA v8 - POST ip:port/api/switchio/pay/v8/refund

Request example:

```
{
    "secureString":"password",
    "request":{
        "transactionId":"1e6a1251-762f-45fc-85fb-227a1c3290b6",
        "amount":10053,
        "currencyCode":203,
        "invoiceNumber":"",
        "daughterCompanyId":""
    }
}
```

PARAMETER		TYPE	DESCRIPTION
secureString		string	Password of the terminal
request		obj	Request object
	transactionId	string	ID of the transaction. Example: *1e6a1251-762f-45fc-85fb-227a1c3290b6*
	amount	int	Amount of the refund, in cents 10053 is 100.53 euro
	currencyCode	int	Code of the currency
	invoiceNumber	string	Variable symbol, length is 20 char
	daughterCompanyId	string	ID of the daughter company. Required only when virtualization is enabled.

3.4.2. Request with callback

This endpoint is not available for GSA over Bluetooth.
Cash register calls only **/confirm**. There is no need to call **/status** and **/result**.

GSA v1 - POST ip:port/paya/cbrefund
GSA v2 - POST ip:port/api/switchio/pay/v2/cbrefund
GSA v4 - POST ip:port/api/switchio/pay/v4/cbrefund
GSA v5 - POST ip:port/api/switchio/pay/v5/cbrefund
GSA v6 - POST ip:port/api/switchio/pay/v6/cbrefund
GSA v7 - POST ip:port/api/switchio/pay/v7/cbrefund
GSA v8 - POST ip:port/api/switchio/pay/v8/cbrefund

Request example:

```
{     "secureString":"password",
      "request":{
         "transactionId":"1e6a1251-762f-45fc-85fb-227a1c3290b6",
         "amount":10053,
         "currencyCode":203,
         "invoiceNumber":"",
         "daughterCompanyId":""
     },
     "progressCallback":{
         "ip":"172.25.11.14",
         "port":7777,
         "endpoint":"/gsa/progress"
     },
     "resultCallback":{
         "ip":"172.25.11.14",
         "port":7777,
         "endpoint":"/gsa/result"
     }
}
```

PARAMETER	TYPE	DESCRIPTION
progressCallback	obj	Object of progress to callback
ip	string	IP to callback
port	int	Port to callback
endpoint	string	Endpoint of method to callback
resultCallback	obj	Object of result to callback
ip	string	IP to callback
port	int	Port to callback
endpoint	string	Endpoint of the method to callback

3.4.3. Response

StatusCode: 200 → transaction has started, the cash register is polling **/status** request until it receives the **"Finished"** state

StatusCode: 202 → Request was processed, but there is some transaction on the terminal that needs to be handled first - server busy.

Response example:

```
{
    "transactionId":"1e6a1251-762f-45fc-85fb-227a1c3290b6",
    "isStarted":true,
    "status":"Other payment in progress"
}
```

PARAMETER	TYPE	DESCRIPTION
transactionId	string	ID of the transaction. Example: *1e6a1251-762f-45fc-85fb-227a1c3290b6*
isStarted	bool	Information whether the refund has started.
status	string	Message of the payment status.

StatusCode: 400 → A transaction with the same ID has already been processed.

Response example:

```
{
    "message":"Duplicate Transaction"
}
```

3.5. REFERRAL REFUND POLLING - 3.2.0

The state of the transaction should be queried periodically by calling **/status**. Once the cash register receives a **"Finished"** state from the status request, the app must request the result of the transaction through **/result** request.

The cash register must confirm that the transaction result from **/result** was received correctly by calling **/confirm.** Otherwise, the transaction will be **automatically reversed**. There is a **60 seconds countdown** for a transaction to be confirmed.

3.5.1. Request

GSA v1 - POST ip:port/paya/referral_refund
GSA v2 - POST ip:port/api/switchio/pay/v2/referral_refund
GSA v4 - POST ip:port/api/switchio/pay/v4/referral_refund
GSA v5 - POST ip:port/api/switchio/pay/v5/referral_refund
GSA v6 - POST ip:port/api/switchio/pay/v6/referral_refund
GSA v7 - POST ip:port/api/switchio/pay/v7/referral_refund
GSA v8 - POST ip:port/api/switchio/pay/v8/referral_refund

Request example:

```
{
    "secureString":"password",
    "request":{
        "transactionId":"1e6a1251-762f-45fc-85fb-227a1c3290b6",
        "amount":10053,
        "currencyCode":203,
        "invoiceNumber":"",
        "daughterCompanyId":"",
        "refundId":""    //ID printed on the ticket
    }
}
```

PARAMETER	TYPE	DESCRIPTION
transactionId	string	Identifier of transport payment transaction, which the cash register wants to create
secureString	string	Secured string, which allows users to access data from the terminal
amount	int	refund amount
currencyCode	int	numeric code for currency code from ISO 4217
invoiceNumber	int	invoice number is a unique, sequential code that is systematically assigned to invoices
daughterCompanyId	string	virtualization identifier which the cash register can use to specify which company should be used for the transaction (for available daughter companies you can call /api/switchio/pay/v2/daughter_companies

refundId	string	ID printed on the ticket

3.5.2. Request with callback

This endpoint is not available for GSA over Bluetooth.
Cash register calls only **/confirm**. There is no need to call **/status** and **/result**.

GSA v1 - POST ip:port/paya/cbreferral_refund
GSA v2 - POST ip:port/api/switchio/pay/v2/cbreferral_refund
GSA v4 - POST ip:port/api/switchio/pay/v4/cbreferral_refund
GSA v5 - POST ip:port/api/switchio/pay/v5/cbreferral_refund
GSA v6 - POST ip:port/api/switchio/pay/v6/cbreferral_refund
GSA v7 - POST ip:port/api/switchio/pay/v7/cbreferral_refund
GSA v8 - POST ip:port/api/switchio/pay/v8/cbreferral_refund

Request example:

```
{     "secureString":"password",
      "request":{
         "transactionId":"1e6a1251-762f-45fc-85fb-227a1c3290b6",
         "amount":10053,
         "currencyCode":203,
         "invoiceNumber":"",
         "daughterCompanyId":""
     },
     "progressCallback":{
         "ip":"172.25.11.14",
         "port":7777,
         "endpoint":"/gsa/progress"
     },
     "resultCallback":{
         "ip":"172.25.11.14",
         "port":7777,
         "endpoint":"/gsa/result"
     }
}
```

PARAMETER	TYPE	DESCRIPTION
progressCallback	obj	Object of progress to callback
ip	string	IP to callback
port	int	Port to callback
endpoint	string	Endpoint of method to callback
resultCallback	obj	Object of result to callback
ip	string	IP to callback
port	int	Port to callback
endpoint	string	Endpoint of the method to callback

3.5.3. Response

StatusCode: 200 → Transaction has started, the cash register is polling **/status** request until it receives the **"Finished"** state.

StatusCode: 202 → Request was processed, but there is some transaction on the terminal that needs to be handled first - server busy.

Response example:

```
{
    "transactionId":"1e6a1251-762f-45fc-85fb-227a1c3290b6",
    "isStarted":true,
    "status":"Other payment in progress"
}
```

StatusCode: 405 → Referral refund is not supported on the side of the terminal app

StatusCode: 400 → Transaction with the same ID has already been processed

Response example:

```
{
    "message":"Duplicate Transaction"
}
```

3.6. REVERSAL POLLING

The Reversal method triggers the last payment return back to the customer's account. This method can only be used for the Payment and Refund operation.

The request starts the reversal of the transaction specified by **originalTransactionId** . If the cash register does not specify **originalTransactionId** , then SwitchioPay will reverse the last authorized transaction.

The state of the transaction should be queried periodically by calling **/status**. Once the cash register receives a **"Finished"** state from the status request, the app must request the result of the transaction through **/result** request.

3.6.1. Request

GSA v1 - POST ip:port/paya/reverse
GSA v2 - POST ip:port/api/switchio/pay/v2/reverse
GSA v4 - POST ip:port/api/switchio/pay/v4/reverse
GSA v5 - POST ip:port/api/switchio/pay/v5/reverse
GSA v6 - POST ip:port/api/switchio/pay/v6/reverse
GSA v7 - POST ip:port/api/switchio/pay/v7/reverse
GSA v8 - POST ip:port/api/switchio/pay/v8/reverse

Request example:

```
{
    "secureString":"password",
    "request":{
        "transactionId":"1e6a1251-762f-45fc-85fb-227a1c3290b6"
        "originalTransactionId":"d9ab5fc6-d8a4-4bbd-a135-114198dc21f5",
        "daughterCompanyId":""
    }
}
```

PARAMETER		TYPE	DESCRIPTION
secureString		string	Password of the terminal
request		obj	Request object
	transactionId	string	ID of the transaction. Example: *1e6a1251-762f-45fc-85fb-227a1c3290b6*
	originalTransactionId	string	ID of the original transaction. Example: *1e6a1251-762f-45fc-85fb-227a1c3290b6*
	daughterCompanyId	string	ID of the daughter company. Required only when virtualization is enabled.

3.6.2. Request with callback

This endpoint is not available for GSA over Bluetooth.
Cash register calls only **/confirm**. There is no need to call **/status** and **/result**.

GSA v2 - POST ip:port/paya/cbreverse
GSA v2 - POST ip:port/api/switchio/pay/v2/cbreverse
GSA v4 - POST ip:port/api/switchio/pay/v4/cbreverse
GSA v5 - POST ip:port/api/switchio/pay/v5/cbreverse
GSA v6 - POST ip:port/api/switchio/pay/v6/cbreverse
GSA v7 - POST ip:port/api/switchio/pay/v7/cvreverse
GSA v8 - POST ip:port/api/switchio/pay/v8/cvreverse

Request example:

```
{
    "secureString":"password",
    "request":{
        "transactionId":"1e6a1251-762f-45fc-85fb-227a1c3290b6"
        "originalTransactionId":"d9ab5fc6-d8a4-4bbd-a135-114198dc21f5",
        "daughterCompanyId":""
    },
    "progressCallback":{
        "ip":"172.25.11.14",
        "port":7777,
        "endpoint":"/gsa/progress"
    },
    "resultCallback":{
        "ip":"172.25.11.14",
        "port":7777,
        "endpoint":"/gsa/result"
    }
}
```

PARAMETER		TYPE	DESCRIPTION
progressCallback		obj	Object of progress to callback
	ip	string	IP to callback
	port	int	Port to callback
	endpoint	string	Endpoint of method to callback
resultCallback		obj	Object of result to callback
	ip	string	IP to callback
	port	int	Port to callback
	endpoint	string	Endpoint of the method to callback

3.6.3. Response

StatusCode: 200 → Transaction has started, cash register is polling **/status** request until it receives the **"Finished"** state.

StatusCode: 202 → Request was processed, but there is some transaction on the terminal that needs to be handled first - server busy.

Response example:

```
{
    "transactionId":"1e6a1251-762f-45fc-85fb-227a1c3290b6",
    "isStarted":true,
    "status":"Other payment in progress"
}
```

PARAMETER	TYPE	DESCRIPTION
transactionId	string	ID of the transaction. Example: *1e6a1251-762f-45fc-85fb-227a1c3290b6*
isStarted	bool	Information whether the reversal has started.
status	string	Message of the payment status.

StatusCode: 400 → Transaction with the same transactionId has been already processed - It was already processed, use different transactionId

StatusCode: 404 → Transaction has not been found with given originalTransactionId

Response example:

```
{
    "message":"Duplicate Transaction"
}
```

3.7. PREAUTHORIZATION POLLING

The request creates preauthorization transaction.

There is an option to cancel the transaction while the transaction is in "**WaitForCard**" state by using **/cancel** endpoint.

The state of the transaction should be queried periodically by calling **/status**. Once the cash register receives a **"Finished"** state from the status request, the app must request the result of the transaction through **/result** request.

The cash register must confirm that the transaction result from **/result** was received correctly by calling **/confirm.** Otherwise, the transaction will be **automatically reversed**. There is a **60 seconds countdown** for a transaction to be confirmed.

3.7.1. Request

GSA v1 - POST ip:port/paya/preauth
GSA v2 - POST ip:port/api/switchio/pay/v2/preauth
GSA v4 - POST ip:port/api/switchio/pay/v4/preauth
GSA v5 - POST ip:port/api/switchio/pay/v5/preauth
GSA v6 - POST ip:port/api/switchio/pay/v6/preauth
GSA v7 - POST ip:port/api/switchio/pay/v7/preauth
GSA v8 - POST ip:port/api/switchio/pay/v8/preauth

Request example:

```
{
    "secureString":"password",
    "request":{
        "transactionId":"1e6a1251-762f-45fc-85fb-227a1c3290b6",
        "amount":123,
        "currencyCode":203,
        "invoiceNumber":"",
        "daughterCompanyId":""
    }
}
```

PARAMETER	TYPE	DESCRIPTION
secureString	string	Password of the terminal
request	obj	Request object
transactionId	string	ID of the transaction. Example: *1e6a1251-762f-45fc-85fb-227a1c3290b6*
amount	int	Amount of the refund, in cents 10053 is 100.53 euro
currencyCode	int	Code of the currency
invoiceNumber	string	Variable symbol, length is 20 char
daughterCompanyId	string	ID of the daughter company. Required only when virtualization is

			enabled.

3.7.2. Request with callback

This endpoint is not available for GSA over Bluetooth.
Cash register calls only **/confirm**. There is no need to call **/status** and **/result**.

GSA v1 - POST ip:port/paya/cbpreauth
GSA v2 - POST ip:port/api/switchio/pay/v2/cbpreauth
GSA v4 - POST ip:port/api/switchio/pay/v4/cbpreauth
GSA v5 - POST ip:port/api/switchio/pay/v5/cbpreauth
GSA v6 - POST ip:port/api/switchio/pay/v6/cbpreauth
GSA v7 - POST ip:port/api/switchio/pay/v7/cbpreauth
GSA v8 - POST ip:port/api/switchio/pay/v8/cbpreauth

Request example:

```
{     "secureString":"password",
      "request":{
         "transactionId":"1e6a1251-762f-45fc-85fb-227a1c3290b6",
         "amount":10053,
         "currencyCode":203,
         "invoiceNumber":"",
         "daughterCompanyId":""
     },
     "progressCallback":{
         "ip":"172.25.11.14",
         "port":7777,
         "endpoint":"/gsa/progress"
     },
     "resultCallback":{
         "ip":"172.25.11.14",
         "port":7777,
         "endpoint":"/gsa/result"
     }
}
```

PARAMETER		TYPE	DESCRIPTION
progressCallback		obj	Object of progress to callback
	ip	string	IP to callback
	port	int	Port to callback
	endpoint	string	Endpoint of method to callback
resultCallback		obj	Object of result to callback
	ip	string	IP to callback
	port	int	Port to callback
	endpoint	string	Endpoint of the method to callback

3.7.3. Response

StatusCode: 200 → transaction has started, cash register is polling **/status** request until it receives the **"Finished"** state

StatusCode: 202 → Request was processed, but there is some transaction on the terminal that needs to be handled first - server busy.

Response example:


```
{
    "transactionId":"1e6a1251-762f-45fc-85fb-227a1c3290b6",
    "isStarted":false,
    "status":"Other payment in progress"
}
```


PARAMETER	TYPE	DESCRIPTION
transactionId	string	ID of the transaction. Example: *1e6a1251-762f-45fc-85fb-227a1c3290b6*
isStarted	bool	Information whether the preauth has started.
status	string	Message of the payment status.

StatusCode: 400 → Transaction with the same transactionId has been already processed - It was already processed, use different transactionId

Response example:


```
{
    "message":"Duplicate Transaction"
}
```

3.8. PREAUTH COMPLETE - POLLING

The request completes the selected preauthorization identified by **originalTransactionId** in the request.

It is not possible to cancel the transaction by using the **/cancel** endpoint**.**

The state of the transaction should be queried periodically by calling **/status**. Once the cash register receives a **"Finished"** state from the status request, the app must request the result of the transaction through **/result** request.

The cash register must confirm that the transaction result from **/result** was received correctly by calling **/confirm.** Otherwise, the transaction will be **automatically reversed**. There is a **60 seconds countdown** for a transaction to be confirmed.

3.8.1. Request

GSA v1 - POST ip:port/paya/preauth/complete
GSA v2 - POST ip:port/api/switchio/pay/v2/preauth/complete
GSA v4 - POST ip:port/api/switchio/pay/v4/preauth/complete
GSA v5 - POST ip:port/api/switchio/pay/v5/preauth/complete
GSA v6 - POST ip:port/api/switchio/pay/v6/preauth/complete
GSA v7 - POST ip:port/api/switchio/pay/v7/preauth/complete
GSA v8 - POST ip:port/api/switchio/pay/v8/preauth/complete

Request example:

```
{
    "secureString":"password",
    "request":{
        "transactionId":"1e6a1251-762f-45fc-85fb-227a1c3290b6",
        "originalTransactionId":"d9ab5fc6-d8a4-4bbd-a135-114198dc21f5",
        "finalAmount":123,
        "invoiceNumber":"",
        "daughterCompanyId":""
    }
}
```

PARAMETER	TYPE	DESCRIPTION
secureString	string	Password of the terminal
request	obj	Request object
transactionId	string	ID of the transaction. Example: *1e6a1251-762f-45fc-85fb-227a1c3290b6*
originalTransactionId	string	ID of the original transaction. Example: *1e6a1251-762f-45fc-85fb-227a1c3290b6*
invoiceNumber	string	OPTIONAL Variable symbol, length is 20 char

	finalAmount	int	Final amount of the preauth, in cents 10053 is 100.53 euro
	daughterCompanyId	string	OPTIONAL ID of the daughter company. Required only when virtualization is enabled.

3.8.2. Request with callback

This endpoint is not available for GSA over Bluetooth.
Cash register calls only **/confirm**. There is no need to call **/status** and **/result**.

GSA v1 - POST ip:port/paya/cbpreauth/complete
GSA v2 - POST ip:port/api/switchio/pay/v2/cbpreauth/complete
GSA v4 - POST ip:port/api/switchio/pay/v4/cbpreauth/complete
GSA v5 - POST ip:port/api/switchio/pay/v5/cbpreauth/complete
GSA v6 - POST ip:port/api/switchio/pay/v6/cbpreauth/complete
GSA v7 - POST ip:port/api/switchio/pay/v7/cbpreauth/complete
GSA v8 - POST ip:port/api/switchio/pay/v8/cbpreauth/complete

Request example:

```
{
    "secureString":"password",
    "request":{
        "transactionId":"1e6a1251-762f-45fc-85fb-227a1c3290b6",
        "originalTransactionId":"d9ab5fc6-d8a4-4bbd-a135-114198dc21f5",
        "finalAmount":123,
        "invoiceNumber":"",
        "daughterCompanyId":""
    },
    "progressCallback":{
        "ip":"172.25.11.14",
        "port":7777,
        "endpoint":"/gla/progress"
    },
    "resultCallback":{
        "ip":"172.25.11.14",
        "port":7777,
        "endpoint":"/gla/result"
    }
}
```

PARAMETER		TYPE	DESCRIPTION
progressCallback		obj	Object of progress to callback
	ip	string	IP to callback
	port	int	Port to callback
	endpoint	string	Endpoint of method to callback

resultCallback		obj	Object of result to callback
	ip	string	IP to callback
	port	int	Port to callback
	endpoint	string	Endpoint of the method to callback

3.8.3. Response

StatusCode: 200 → transaction has started, cash register is polling **/status** request until it receives the **"Finished"** state

StatusCode: 202 → Request was processed, but there is some transaction on the terminal that needs to be handled first - server busy.

Response example:

```
{
    "transactionId":"1e6a1251-762f-45fc-85fb-227a1c3290b6",
    "isStarted":false,
    "status":"Other payment in progress"
}
```

PARAMETER	TYPE	DESCRIPTION
transactionId	string	ID of the transaction. Example: *1e6a1251-762f-45fc-85fb-227a1c3290b6*
isStarted	bool	Information whether the preauth completion has started.
status	string	Message of the payment status.

If you send a request with a transaction ID that was already processed, you will get a different response.

StatusCode: 400 → Transaction with the same transactionId has been already processed - It was already processed, use different transactionId

Response example:

```
{
    "message":"Duplicate Transaction"
}
```

If you send a request with a transaction ID that was not found, you will get a different response.

StatusCode: 404 → Transaction has not been found with given originalTransactionId

Response example:

```
{
```

```
    "message":"Not found"
}
```

3.9. PREAUTH EXTERNAL COMPLETE - POLLING - since Switchio PAY 3.3.0

It allows completing a pre-authorization transaction from a different terminal. The pre-authorization transaction is identified by **terminalId, authCode, and sequenceNumber** in the request.

It is not possible to cancel the transaction by using the **/cancel** endpoint**.**

The state of the transaction should be queried periodically by calling **/status**. Once the cash register receives a **"Finished"** state from the status request, the app must request the result of the transaction through **/result** request.

The cash register must confirm that the transaction result from **/result** was received correctly by calling **/confirm.** Otherwise, the transaction will be **automatically reversed**. There is a **60 seconds countdown** for a transaction to be confirmed.

3.9.1. Request

GSA v1 - POST ip:port/paya/preauth/external/complete
GSA v2 - POST ip:port/api/switchio/pay/v2/preauth/external/complete
GSA v4 - POST ip:port/api/switchio/pay/v4/preauth/external/complete
GSA v5 - POST ip:port/api/switchio/pay/v5/preauth/external/complete
GSA v6 - POST ip:port/api/switchio/pay/v6/preauth/external/complete

GSA v7 - POST ip:port/api/switchio/pay/v7/preauth/external/complete
GSA v8 - POST ip:port/api/switchio/pay/v8/preauth/external/complete

Request example:

```
{
   "secureString":"password",
   "request":{
      "transactionId":"1e6a1251-762f-45fc-85fb-227a1c3290b6",
      "originalTransactionData":{
         "terminalId":"TERMINALID",
         "authCode":"123456",
         "seqNumber":"001001234"
      },
      "finalAmount":123,
      "invoiceNumber":"",
      "daughterCompanyId":""
   }
}
```

PARAMETER	TYPE	DESCRIPTION
secureString	string	Password of the terminal

request		obj	Request object
	transactionId	string	ID of the transaction. Example: *1e6a1251-762f-45fc-85fb-227a1c3290b6*
	finalAmount	int	Final amount of the preauth, in cents 10053 is 100.53 euro
	invoiceNumber	string	Variable symbol, length is 20 char
	daughterCompanyId	string	ID of the daughter company. Required only when virtualization is enabled.
originalTransactionData		string	Object
	terminalID	string	Name of the terminal
	authCode	string	Code of the authorization
	seqNumber	string	Transaction sequence number

3.9.2. Response

StatusCode: 200 → transaction has started, cash register is polling **/status** request until it receives the **"Finished"** state

StatusCode: 202 → Request was processed, but there is some transaction on the terminal that needs to be handled first - server busy.

Response example:

```
{
    "transactionId":"1e6a1251-762f-45fc-85fb-227a1c3290b6",
    "isStarted":false,
    "status":"Other payment in progress"
}
```

PARAMETER	TYPE	DESCRIPTION
transactionId	string	ID of the transaction. Example: *1e6a1251-762f-45fc-85fb-227a1c3290b6*
isStarted	bool	Information whether the preauth completion has started.
status	string	Message of the payment status.

If you send a request with a transaction ID that was already processed, you will get a different response.

StatusCode: 400 → Transaction with the same transactionId has been already processed - It was already processed, use different transactionId

Response example:

```
{
    "message":"Duplicate Transaction"
}
```

If you send a request with a transaction ID that was not found, you will get a different response.

StatusCode: 404 → Transaction has not been found with given originalTransactionId

Response example:

```
{
    "message":"Not found"
}
```

3.10. PREAUTH INCREMENT - POLLING

The endpoint increment opened preauthorization identified by **originalTransactionId** with an amount specified in the request.

It is not possible to cancel the transaction by using the **/cancel** endpoint**.**

The state of the transaction should be queried periodically by calling **/status**. Once the cash register receives a **"Finished"** state from the status request, the app must request the result of the transaction through **/result** request.

The cash register must confirm that the transaction result from **/result** was received correctly by calling **/confirm.** Otherwise, the transaction will be **automatically reversed**. There is a **60 seconds countdown** for a transaction to be confirmed.

3.10.1. Request

GSA v1 - POST ip:port/paya/preauth/increment
GSA v2 - POST ip:port/api/switchio/pay/v2/preauth/increment
GSA v4 - POST ip:port/api/switchio/pay/v4/preauth/increment
GSA v5 - POST ip:port/api/switchio/pay/v5/preauth/increment
GSA v6 - POST ip:port/api/switchio/pay/v6/preauth/increment
GSA v7 - POST ip:port/api/switchio/pay/v7/preauth/increment
GSA v8 - POST ip:port/api/switchio/pay/v8/preauth/increment

Request example:

```
{
    "secureString":"password",
    "request":{
        "transactionId":"1e6a1251-762f-45fc-85fb-227a1c3290b6",
        "originalTransactionId":"d9ab5fc6-d8a4-4bbd-a135-114198dc21f5",
        "incrementAmount":123,
        "invoiceNumber":"",
        "daughterCompanyId":""
```

```
    }
}
```

PARAMETER	TYPE	DESCRIPTION
secureString	string	Password of the terminal
request	obj	Request object
transactionId	string	ID of the transaction. Example: *1e6a1251-762f-45fc-85fb-227a1c3290b6*
originalTransactionId	string	ID of the original transaction. Example: *1e6a1251-762f-45fc-85fb-227a1c3290b6*
incrementAmount	int	Incremental amount of the preauth, in cents 10053 is 100.53 euro
invoiceNumber	string	OPTIONAL Variable symbol, length is 20 char
daughterCompanyId	string	OPTIONAL ID of the daughter company. Required only when virtualization is enabled.

3.10.2. Request with callback

This endpoint is not available for GSA over Bluetooth.
Cash register calls only **/confirm**. There is no need to call **/status** and **/result**.

GSA v1 - POST ip:port/paya/cbpreauth/increment
GSA v2 - POST ip:port/api/switchio/pay/v2/cbpreauth/increment
GSA v4 - POST ip:port/api/switchio/pay/v4/cbpreauth/increment
GSA v5 - POST ip:port/api/switchio/pay/v5/cbpreauth/increment
GSA v6 - POST ip:port/api/switchio/pay/v6/cbpreauth/increment
GSA v7 - POST ip:port/api/switchio/pay/v7/cbpreauth/increment
GSA v8 - POST ip:port/api/switchio/pay/v8/cbpreauth/increment

Request example:

```
{
    "secureString":"password",
    "request":{
        "transactionId":"1e6a1251-762f-45fc-85fb-227a1c3290b6",
        "originalTransactionId":"d9ab5fc6-d8a4-4bbd-a135-114198dc21f5",
        "incrementAmount":123,
        "invoiceNumber":"",
        "daughterCompanyId":""
    },
    "progressCallback":{
        "ip":"172.25.11.14",
```

```
        "port":7777,
        "endpoint":"/gla/progress"
    },
    "resultCallback":{
        "ip":"172.25.11.14",
        "port":7777,
        "endpoint":"/gla/result"
    }
}
```

PARAMETER	TYPE	DESCRIPTION
progressCallback	obj	Object of progress to callback
ip	string	IP to callback
port	int	Port to callback
endpoint	string	Endpoint of method to callback
resultCallback	obj	Object of result to callback
ip	string	IP to callback
port	int	Port to callback
endpoint	string	Endpoint of the method to callback

3.10.3. Response

StatusCode: 200 → transaction has started, cash register is polling **/status** request until it receives the **"Finished"** state

StatusCode: 202 → Request was processed, but there is some transaction on the terminal that needs to be handled first - server busy.

Response example:

```
{
    "transactionId":"1e6a1251-762f-45fc-85fb-227a1c3290b6",
    "isStarted":false,
    "status":"Other payment in progress"
}
```

PARAMETER	TYPE	DESCRIPTION
transactionId	string	ID of the transaction. Example: *1e6a1251-762f-45fc-85fb-227a1c3290b6*
isStarted	bool	Information whether the preauth increment has started.

status	string	Message of the payment status.

If you send a request with a transaction ID that was already processed, you will get a different response.

StatusCode: 400 → Transaction with the same transactionId has been already processed - It was already processed, use different transactionId

Response example:

```
{
    "message":"Duplicate Transaction"
}
```

If you send a request with a transaction ID that was not found, you will get a different response.

StatusCode: 404 → Transaction has not been found with given originalTransactionId

Response example:

```
{
    "message":"Duplicate Transaction"
}
```

3.11. PREAUTH CANCEL - POLLING

The endpoint cancels the opened preauthorization, which is identified by **originalTransactionId**.

It is not possible to cancel the transaction by using the **/cancel** endpoint**.**

The state of the transaction should be queried periodically by calling **/status**. Once the cash register receives a **"Finished"** state from the status request, the app must request the result of the transaction through **/result** request.

No need to call **/confirm** after receiving **Finished** state.

3.11.1. Request

GSA v1 - POST ip:port/paya/preauth/cancel
GSA v2 - POST ip:port/api/switchio/pay/v2/preauth/cancel
GSA v4 - POST ip:port/api/switchio/pay/v4/preauth/cancel
GSA v5 - POST ip:port/api/switchio/pay/v5/preauth/cancel
GSA v6 - POST ip:port/api/switchio/pay/v6/preauth/cancel
GSA v7 - POST ip:port/api/switchio/pay/v7/preauth/cancel
GSA v8 - POST ip:port/api/switchio/pay/v8/preauth/cancel

Request example:

```
{
    "secureString":"password",
    "request":{
        "transactionId":"1e6a1251-762f-45fc-85fb-227a1c3290b6",
        "originalTransactionId":"d9ab5fc6-d8a4-4bbd-a135-114198dc21f5",
        "daughterCompanyId":""
    }
}
```

PARAMETER	TYPE	DESCRIPTION
secureString	string	Password of the terminal
request	obj	Request object
transactionId	string	ID of the transaction. Example: *1e6a1251-762f-45fc-85fb-227a1c3290b6*
originalTransactionId	string	ID of the original transaction. Example: *1e6a1251-762f-45fc-85fb-227a1c3290b6*
daughterCompanyId	string	OPTIONAL ID of the daughter company. Required only when virtualization is enabled.

3.11.2. Request with callback

This endpoint is not available for GSA over Bluetooth.
There is no need to call **/status**, **/result** and **/confirm**.

GSA v1 - POST ip:port/paya/cbpreauth/cancel
GSA v2 - POST ip:port/api/switchio/pay/v2/cbpreauth/cancel
GSA v4 - POST ip:port/api/switchio/pay/v4/cbpreauth/cancel
GSA v5 - POST ip:port/api/switchio/pay/v5/cbpreauth/cancel
GSA v6 - POST ip:port/api/switchio/pay/v6/cbpreauth/cancel
GSA v7 - POST ip:port/api/switchio/pay/v7/cbpreauth/cancel
GSA v8 - POST ip:port/api/switchio/pay/v8/cbpreauth/cancel

Request example:

```
{
    "secureString":"password",
    "request":{
        "transactionId":"1e6a1251-762f-45fc-85fb-227a1c3290b6"
        "originalTransactionId":"d9ab5fc6-d8a4-4bbd-a135-114198dc21f5",
        "daughterCompanyId":""
    },
    "progressCallback":{
        "ip":"172.25.11.14",
        "port":7777,
        "endpoint":"/gsa/progress"
    },
    "resultCallback":{
        "ip":"172.25.11.14",
        "port":7777,
        "endpoint":"/gsa/result"
    }
}
```

PARAMETER	TYPE	DESCRIPTION
progressCallback	obj	Object of progress to callback
ip	string	IP to callback
port	int	Port to callback
endpoint	string	Endpoint of method to callback
resultCallback	obj	Object of result to callback
ip	string	IP to callback
port	int	Port to callback
endpoint	string	Endpoint of the method to callback

3.11.3. Response

StatusCode: 200 → transaction has started, cash register is polling **/status** request until it receives the **"Finished"** state

StatusCode: 202 → Request was processed, but there is some transaction on the terminal that needs to be handled first - server busy.

Response example:

```
{
    "transactionId":"1e6a1251-762f-45fc-85fb-227a1c3290b6",
    "isStarted":false,
    "status":"Other payment in progress"
}
```

PARAMETER	TYPE	DESCRIPTION
transactionId	string	ID of the transaction. Example: *1e6a1251-762f-45fc-85fb-227a1c3290b6*
isStarted	bool	Information whether the preauth cancel has started.
status	string	Message of the payment status.

If you send a request with a transaction ID that was already processed, you will get a different response.

StatusCode: 400 → Transaction with the same transactionId has been already processed - It was already processed, use different transactionId

Response example:

```
{
    "message":"Duplicate Transaction"
}
```

If you send a request with a transaction ID that was not found, you will get a different response.

StatusCode: 404 → Transaction has not been found with given originalTransactionId

Response example:

```
{
    "message":"Not found"
}
```

3.12. PREAUTH EXTERNAL CANCEL

It allows external canceling of a pre-authorization transaction from a different terminal. The pre-authorization transaction is identified by **terminalId, authCode, and sequenceNumber** in the request.

It is not possible to cancel the transaction by using the **/cancel** endpoint**.**

The state of the transaction should be queried periodically by calling **/status**. Once the cash register receives a **"Finished"** state from the status request, the app must request the result of the transaction through **/result** request.

No need to call **/confirm** after receiving **Finished** state.

3.12.1. Request

GSA v1 - POST ip:port/paya/preauth/external/cancel
GSA v2 - POST ip:port/api/switchio/pay/v2/preauth/external/cancel
GSA v4 - POST ip:port/api/switchio/pay/v4/preauth/external/cancel
GSA v5 - POST ip:port/api/switchio/pay/v5/preauth/external/cancel
GSA v6 - POST ip:port/api/switchio/pay/v6/preauth/external/cancel
GSA v7 - POST ip:port/api/switchio/pay/v7/preauth/external/cancel
GSA v8 - POST ip:port/api/switchio/pay/v8/preauth/external/cancel

Request example:

```
{
   "secureString":"password",
   "request":{
      "transactionId":"1e6a1251-762f-45fc-85fb-227a1c3290b6",
      "originalTransactionData":{
         "terminalId":"TERMINALID",
         "authCode":"123456",
         "seqNumber":"001001234"
      },
      "daughterCompanyId":""
   }
}
```

PARAMETER	TYPE	DESCRIPTION
secureString	string	Password of the terminal
request	obj	Request object
transactionId	string	ID of the transaction. Example: *1e6a1251-762f-45fc-85fb-227a1c3290b6*
daughterCompanyId	string	OPTIONAL ID of the daughter company. Required only when virtualization is enabled.

originalTransactionData		string	Object
	terminalID	string	Name of the terminal
	authCode	string	Code of the authorization
	seqNumber	string	Transaction sequence number

3.12.2. Response

StatusCode: 200 → transaction has started, cash register is polling **/status** request until it receives the **"Finished"** state

StatusCode: 202 → Request was processed, but there is some transaction on the terminal that needs to be handled first - server busy.

Response example:

```
{
    "transactionId":"1e6a1251-762f-45fc-85fb-227a1c3290b6",
    "isStarted":false,
    "status":"Other payment in progress"
}
```

PARAMETER	TYPE	DESCRIPTION
transactionId	string	ID of the transaction. Example: *1e6a1251-762f-45fc-85fb-227a1c3290b6*
isStarted	bool	Information whether the preauth completion has started.
status	string	Message of the payment status.

If you send a request with a transaction ID that was already processed, you will get a different response.

StatusCode: 400 → Transaction with the same transactionId has been already processed - It was already processed, use different transactionId

Response example:

```
{
    "message":"Duplicate Transaction"
}
```

If you send a request with a transaction ID that was not found, you will get a different response.

StatusCode: 404 → Transaction has not been found with given originalTransactionId

Response example:

```
{
    "message":"Not found"
}
```

3.13. PREAUTH EXTERNAL INCREMENT

It allows external increment of a pre-authorization transaction from a different terminal. The pre-authorization transaction is identified by **terminalId, authCode, and sequenceNumber** in the request.

It is not possible to cancel the transaction by using the **/cancel** endpoint**.**

The state of the transaction should be queried periodically by calling **/status**. Once the cash register receives a **"Finished"** state from the status request, the app must request the result of the transaction through **/result** request.

The cash register must confirm that the transaction result from **/result** was received correctly by calling **/confirm.** Otherwise, the transaction will be **automatically reversed**. There is a **60 seconds countdown** for a transaction to be confirmed.

This endpoint is available for Switchio Pay version 4.7.0 and higher.

3.13.1. Request

GSA v7 - POST ip:port/api/switchio/pay/v7/preauth/external/increment
GSA v8 - POST ip:port/api/switchio/pay/v8/preauth/external/increment

Request example:

```
{
   "secureString":"password",
   "request":{
      "transactionId":"1e6a1251-762f-45fc-85fb-227a1c3290b6",
      "originalTransactionData":{
         "terminalId":"TERMINALID",
         "authCode":"123456",
         "seqNumber":"001001234"
      },
      "incrementAmount":"123456",
      "invoiceNumber":"",
      "daughterCompanyId":""
   }
}
```

PARAMETER		TYPE	DESCRIPTION
secureString		string	Password of the terminal
request		obj	Request object
	transactionId	string	ID of the transaction. Example: *1e6a1251-762f-45fc-85fb-227a1c3290b6*
	incrementAmount	int	Incremental amount of the preauth, in cents

			10053 is 100.53 euro
	invoiceNumber	string	OPTIONAL Variable symbol, length is 20 char
	daughterCompanyId	string	OPTIONAL ID of the daughter company. Required only when virtualization is enabled.
originalTransactionData		obj	Original Transaction Data object
	terminalID	string	Name of the terminal
	authCode	string	Code of the authorization
	seqNumber	string	Transaction sequence number

3.13.2. Response

StatusCode: 200 → transaction has started, cash register is polling **/status** request until it receives the **"Finished"** state

StatusCode: 202 → Request was processed, but there is some transaction on the terminal that needs to be handled first - server busy.

Response example:

```
{
    "transactionId":"1e6a1251-762f-45fc-85fb-227a1c3290b6",
    "isStarted":false,
    "status":"Other payment in progress"
}
```

PARAMETER	TYPE	DESCRIPTION
transactionId	string	ID of the transaction. Example: *1e6a1251-762f-45fc-85fb-227a1c3290b6*
isStarted	bool	Information whether the preauth completion has started.
status	string	Message of the payment status.

If you send a request with a transaction ID that was already processed, you will get a different response.

StatusCode: 400 → Transaction with the same transactionId has been already processed - It was already processed, use different transactionId

Response example:

```
{
    "message":"Duplicate Transaction"
}
```

If you send a request with a transaction ID that was not found, you will get a different response.

StatusCode: 404 → Transaction has not been found with given originalTransactionData

Response example:

```
{
      "message":"Not found"
}
```

3.14. PREAUTH EXTERNAL DECREMENT

It allows external decrement of a pre-authorization transaction from a different terminal. The pre-authorization transaction is identified by **terminalId, authCode, and sequenceNumber** in the request.

It is not possible to cancel the transaction by using the **/cancel** endpoint**.**

The state of the transaction should be queried periodically by calling **/status**. Once the cash register receives a **"Finished"** state from the status request, the app must request the result of the transaction through **/result** request.

The cash register must confirm that the transaction result from **/result** was received correctly by calling **/confirm.** Otherwise, the transaction will be **automatically reversed**. There is a **60 seconds countdown** for a transaction to be confirmed.

This endpoint is available for Switchio Pay version 4.7.0 and higher.

3.14.1. Request

GSA v7 - POST ip:port/api/switchio/pay/v7/preauth/external/decrement
GSA v8 - POST ip:port/api/switchio/pay/v8/preauth/external/decrement

Request example:

```
{
   "secureString":"password",
   "request":{
      "transactionId":"1e6a1251-762f-45fc-85fb-227a1c3290b6",
      "originalTransactionData":{
         "terminalId":"TERMINALID",
         "authCode":"123456",
         "seqNumber":"001001234"
      },
      "decrementAmount":"123456",
      "invoiceNumber":"",
      "daughterCompanyId":""
   }
}
```

PARAMETER		TYPE	DESCRIPTION
secureString		string	Password of the terminal
request		obj	Request object
	transactionId	string	ID of the transaction. Example: *1e6a1251-762f-45fc-85fb-227a1c3290b6*
	decrementAmount	int	Decremental amount of the preauth, in cents 10053 is 100.53 euro

	invoiceNumber	string	OPTIONAL Variable symbol, length is 20 char
	daughterCompanyId	string	OPTIONAL ID of the daughter company. Required only when virtualization is enabled.
originalTransactionData		obj	Original Transaction Data object
	terminalID	string	Name of the terminal
	authCode	string	Code of the authorization
	seqNumber	string	Transaction sequence number

3.14.2. Response

StatusCode: 200 → transaction has started, cash register is polling **/status** request until it receives the **"Finished"** state

StatusCode: 202 → Request was processed, but there is some transaction on the terminal that needs to be handled first - server busy.

Response example:

```
{
    "transactionId":"1e6a1251-762f-45fc-85fb-227a1c3290b6",
    "isStarted":false,
    "status":"Other payment in progress"
}
```

PARAMETER	TYPE	DESCRIPTION
transactionId	string	ID of the transaction. Example: *1e6a1251-762f-45fc-85fb-227a1c3290b6*
isStarted	bool	Information whether the preauth completion has started.
status	string	Message of the payment status.

If you send a request with a transaction ID that was already processed, you will get a different response.

StatusCode: 400 → Transaction with the same transactionId has been already processed - It was already processed, use different transactionId

Response example:

```
{
    "message":"Duplicate Transaction"
}
```

If you send a request with a transaction ID that was not found, you will get a different response.

StatusCode: 404 → Transaction has not been found with given originalTransactionData

Response example:

```
{
      "message":"Not found"
}
```

3.15. GET TRANSACTION STATUS

Requesting verification of an authorized transaction with the authorization server. After calling transaction_status, it is necessary to call the TRANSACTION RESULT endpoint which will return the TransactionResult of the transaction.

3.15.1. Request

GSA v1 - POST ip:port/paya/transaction_status
GSA v2 - POST ip:port/api/switchio/pay/v2/transaction_status
GSA v4 - POST ip:port/api/switchio/pay/v4/transaction_status
GSA v5 - POST ip:port/api/switchio/pay/v5/transaction_status
GSA v6 - POST ip:port/api/switchio/pay/v6/transaction_status
GSA v7 - POST ip:port/api/switchio/pay/v7/transaction_status
GSA v8 - POST ip:port/api/switchio/pay/v8/transaction_status

Request example:

```
{
    "secureString":"password",
    "transactionId":"48a3ea7d-b82f-4b42-abbb-5043365e4faa"
}
```

PARAMETER	TYPE	DESCRIPTION
transactionId	string	ID of the transaction. Example: *1e6a1251-762f-45fc-85fb-227a1c3290b6*
secureString	string	Password of the terminal

3.15.2. Response

Response example:

```
{
    "isStarted": true,
    "transactionId" : "48a3ea7d-b82f-4b42-abbb-5043365e4faa",
    "status": "OK"
}
```

PARAMETER	TYPE	DESCRIPTION
isStarted	bool	Indicates whether the transaction processing started.
transactionId	string	ID of the transaction. Example: *1e6a1251-762f-45fc-85fb-227a1c3290b6*
status	string	Additional information about the transaction status.

3.16. GET AVAILABLE PREAUTHORIZATION

Requesting a list of opened preauthorization transactions for incrementing, canceling, or completing.

3.16.1. Request

GSA v1 - POST ip:port/paya/preauth/transactions
GSA v2 - POST ip:port/api/switchio/pay/v2/preauth/transactions
GSA v4 - POST ip:port/api/switchio/pay/v4/preauth/transactions
GSA v5 - POST ip:port/api/switchio/pay/v5/preauth/transactions
GSA v6 - POST ip:port/api/switchio/pay/v6/preauth/transactions
GSA v7 - POST ip:port/api/switchio/pay/v7/preauth/transactions
GSA v8 - POST ip:port/api/switchio/pay/v8/preauth/transactions

Request example:

```
{
      "secureString":"password",
      "daughterCompanyId":""
}
```

PARAMETER	TYPE	DESCRIPTION
secureString	string	Password of the terminal
daughterCompanyId	string	OPTIONAL ID of the daughter company. Default is "", this is the merchant identifier to be used to get preath

3.16.2. Response

For response structure and examples for each version see TRANSACTION RESULT
The response will come as a list of transaction results (link above).

Response example:

```
[
 {transaction result 1}, {transaction result 2}, {transaction result 3}
]
```

3.17. PREAUTH DECREMENT - POLLING

This endpoint is available for Switchio Pay version 4.4.0 and higher.

Decrements the transaction amount of the original preauthorized transaction, which is identified by the originalTransactionId in the request. It is not possible to cancel this operation using the **/cancel** endpoint. If the Preauth decrement amount is higher than the originally authorized amount, the transaction is declined by the authorization host.

3.17.1. Request

GSA v5 - POST /api/switchio/pay/v5/preauth/decrement
GSA v6 - POST /api/switchio/pay/v6/preauth/decrement
GSA v7 - POST /api/switchio/pay/v7/preauth/decrement
GSA v8 - POST /api/switchio/pay/v8/preauth/decrement

Request example:

```
{
    "secureString":"password",
    "request":{
        "transactionId":"1e6a1251-762f-45fc-85fb-227a1c3290b6",
        "originalTransactionId":"d9ab5fc6-d8a4-4bbd-a135-114198dc21f5",
        "decrementAmount":123,
        "invoiceNumber":"",
        "daughterCompanyId":""
    }
}
```

PARAMETER	TYPE	DESCRIPTION
secureString	string	Password of the terminal
request	obj	Request object
transactionId	string	ID of the transaction. Example: *1e6a1251-762f-45fc-85fb-227a1c3290b6*
originalTransactionId	string	ID of the original transaction. Example: *d9ab5fc6-d8a4-4bbd-a135-114198dc21f5*
decrementAmount	int	Decremental amount of the preauth, in cents. Example: 10053 (100.53 EUR)
invoiceNumber	string	OPTIONAL Variable symbol, length is 20 char
daughterCompanyId	string	OPTIONAL ID of the daughter company. Required only when virtualization is enabled.

3.17.2. Response

Response example:

```
{
    "transactionId":"1e6a1251-762f-45fc-85fb-227a1c3290b6",
    "isStarted":false,
    "status":"Other payment in progress"
}
```

3.18. QR PAYMENT - POLLING

This endpoint is available for Switchio Pay version 4.4.0 and higher.

Performs a QR payment. It is necessary to call the CONFIRM TRANSACTION endpoint after getting the Finished status. If the CONFIRM TRANSACTION endpoint is not called after finishing the payment, the REVERSAL method is triggered.

3.18.1. Request

GSA v5 - POST /api/switchio/pay/v5/qrpayment
GSA v6 - POST /api/switchio/pay/v6/qrpayment
GSA v7 - POST /api/switchio/pay/v7/qrpayment
GSA v8 - POST /api/switchio/pay/v8/qrpayment

Request example:

```
{
    "secureString":"password",
    "request":{
        "transactionId":"1e6a1251-762f-45fc-85fb-227a1c3290b6",
        "amount":10053,
        "tipAmount":100,
        "currencyCode":203,
        "invoiceNumber":"",
        "daughterCompanyId":""
    }
}
```

PARAMETER		TYPE	DESCRIPTION
secureString		string	Password of the terminal
request		obj	Request object
	transactionId	string	ID of the transaction. Example: *1e6a1251-762f-45fc-85fb-227a1c3290b6*
	amount	int	Amount of the payment, in cents 10053 is 100.53 euro

	tipAmount	int	OPTIONAL Tip amount of the payment
	currencyCode	int	Code of the currency
	invoiceNumber	string	OPTIONAL Variable symbol length is 20 char
	daughterCompanyId	string	OPTIONAL ID of the daughter company. Required only when virtualization is enabled.

3.18.2. Response

Response example:

```
{
    "transactionId":"1e6a1251-762f-45fc-85fb-227a1c3290b6",
    "isStarted":false,
    "status":"Other payment in progress"
}
```

3.19. CASHBACK

This endpoint is available for Switchio Pay version 4.6.0 and higher.

This method allows customers to withdraw cash from a merchant during a card transaction.

NOTE: Only domestic cards can be used for the CASHBACK operation. This means that the terminal country code needs to be the same as the payment card country code. If the card country code differs from the terminal country code, the result of the transaction returns DOMESTIC_CARD_ONLY_ERROR - see Terminal Error Codes for more detail.

3.19.1. Request

GSA v1 - POST ip:port/paya/cashback
GSA v2 - POST ip:port/api/switchio/pay/v2/cashback
GSA v4 - POST ip:port/api/switchio/pay/v4/cashback
GSA v5 - POST ip:port/api/switchio/pay/v5/cashback
GSA v6 - POST ip:port/api/switchio/pay/v6/cashback
GSA v7 - POST ip:port/api/switchio/pay/v7/cashback
GSA v8 - POST ip:port/api/switchio/pay/v8/cashback

Request example:

```
{
    "secureString":"password",
    "request":{
        "transactionId":"1e6a1251-762f-45fc-85fb-227a1c3290b6",
        "saleAmount":10053,
        "cashbackAmount":100,
        "currencyCode":203,
        "invoiceNumber":"",
        "daughterCompanyId":""
    }
}
```

PARAMETER	TYPE	DESCRIPTION
secureString	string	Password of the terminal
request	obj	Request object
transactionId	string	ID of the transaction. Example: *1e6a1251-762f-45fc-85fb-227a1c3290b6*
saleAmount	int	Amount of the payment, in cents 10053 is 100.53 euro
cashbackAmount	int	Amount to be withdrawn in cash from the merchant.
currencyCode	int	Code of the currency

	invoiceNumber	string	OPTIONAL Variable symbol length is 20 char
	daughterCompanyId	string	OPTIONAL ID of the daughter company. Required only when virtualization is enabled.

3.19.2. Response

Response example:

```
{
    "transactionId":"1e6a1251-762f-45fc-85fb-227a1c3290b6",
    "isStarted":false,
    "status":"Other payment in progress"
}
```

4. TRANSPORT ENDPOINTS

TRANSPORT ENDPOINTS ARE SUPPORTED ONLY FOR NEXGO TERMINAL PLATFORM

4.1. TRANSPORT INIT

By using this request, the cash register initializes transport mode by downloading the initial deny list to SwitchioPay. The result will be available after the consumer device receives "**FinishedPlain**" in **/status** status polling.

4.1.1. Request

GSA v1 - POST ip:port/paya/transport/init
GSA v2 - POST ip:port/api/switchio/pay/v2/transport/init
GSA v4 - POST ip:port/api/switchio/pay/v4/transport/init
GSA v5 - POST ip:port/api/switchio/pay/v5/transport/init
GSA v6 - POST ip:port/api/switchio/pay/v6/transport/init
GSA v7 - POST ip:port/api/switchio/pay/v7/transport/init
GSA v8 - POST ip:port/api/switchio/pay/v8/transport/init

Request example:

```
{
    "transactionId":"6c1c7e21-7805-49cc-8045-07cd8d9e3a3c",
    "secureString":"123456"
}
```

PARAMETER	TYPE	DESCRIPTION
transactionId	string	Identifier of transport payment transaction, which the cash register wants to create
secureString	string	Secured string, which allows users to access data from the terminal

Parameters of the Transport Init request

4.1.2. Response

StatusCode: 200 → Transaction has started, cash register is polling **/status** request until it receives "**FinishedPlain**" state

StatusCode: 202 → Request was processed, but there is some transaction on the terminal that needs to be handled first - server busy.

Response example:

```
{
```

```
    "transactionId":"1e6a1251-762f-45fc-85fb-227a1c3290b6",
    "isStarted":true,
    "status":"Other payment in progress"
}
```

After cash register received "**FinishedPlain",** cash register should request result of the transaction with using endpoint **/result**, SwitchioPay will respond with following response:

Result Response for transport init

```
{"resultCode":0,"message":"ok"}

const val SUCCESS = 0
const val PARAM_ERROR = 2 // Internal parameter error
const val DOWNLOAD_ERROR = 3
```

4.2. TRANSPORT PAYMENT

Requesting transport payment from SwitchioPay**.** The result will be available after the consumer device receives "**FinishedPlain**" in **/status** status polling.

4.2.1. Request

GSA v1 - POST ip:port/paya/transport/payment
GSA v2 - POST ip:port/api/switchio/pay/v2/transport/payment
GSA v4 - POST ip:port/api/switchio/pay/v4/transport/payment
GSA v5 - POST ip:port/api/switchio/pay/v5/transport/payment
GSA v6 - POST ip:port/api/switchio/pay/v6/transport/payment
GSA v7 - POST ip:port/api/switchio/pay/v7/transport/payment
GSA v8 - POST ip:port/api/switchio/pay/v8/transport/payment

Request example:

```
{
    "secureString":"123456",
    "request":{
        "transactionId":"6c1c7e21-7805-49cc-8045-07cd8d9e3a3c",
        "transportData":"ZGF0YQ==", //Base64 from string "data"
        "transportAmount":10053,
        "transportCurrencyCode":203,
        "processingMode":"KFT",
        "invoiceNumber":"VS",
        "virtualTerminalId":"TERM004"
    }
}
```

PARAMETER	TYPE	DESCRIPTION
transactionId	string	Identifier of transport payment transaction, which the cash register wants to create

secureString	string	Secured string, which allows users to access data from the terminal
transportData	string	additional information cash register wants to provide for transport payment in Base64
transportAmount	int	OPTIONAL Transaction amount shown on the terminal/validator display when the card is tapped. The amount is sent in the minimum currency unit. For example, if the currency is Kč 10,00, amount = „1000“; EUR 1,50 = „150“ (without decimal point).
transportCurrencyCode	int	OPTIONAL Currency code visualized on the terminal/validator display when the card is tapped.
invoiceNumber	string	OPTIONAL Variable symbol, max length is 20 char
processsingMode	string	OPTIONAL Processing mode of transaction, e.g: KFT
virtualTerminalId	string	OPTIONAL ID of the virtual terminal id. Required only when virtualization is enabled.

Parameters of the Transport Payment request

4.2.2. Response

StatusCode: 200 → Transaction has started, cash register is polling **/status** request until it receives "**FinishedPlain**" state

StatusCode: 202 → Request was processed, but there is some transaction on the terminal that needs to be handled first - server busy.

Response example:

```
{
    "transactionId":"1e6a1251-762f-45fc-85fb-227a1c3290b6",
    "isStarted":true,
    "status":"Other payment in progress"
}
```

After cash register received "**FinishedPlain",** cash register should request result of the transaction with using endpoint **/result**, SwitchioPay will respond with following response:

Result Response for transport payment

```
{
    "processingResultCode":0,
    "message":"ok",
    "fareTransactionRequest":{
        "device":{
            "type":"NEXGO",
```

```
            "reader_id":"TN86RTA1",
            "reader_sn":"N860W006694",
            "deny_list_version":""
        },
        "transaction":{

"token":"11688528C822DAA6C0757EA0F5E647055452A83E5F90109ADD226471D89DBEB960",
            "id":"6c1c7e21-7805-49cc-8045-07cd8d9e3a3c",
            "dttm":"2022-10-03T12:53:01Z",
            "result":"NO_DENYLIST"
        },
        "fare_data":"test_transport_data",
        "emv_data":{
        "tags":[
            {"key":"9F06","value":"A0000000031010"},
            {"key":"5F24","value":"2208"}
        ],
        "encrypted":{
            "encryption_method":"MNSP",

"encrypted_data":"49727E9FEECAC356BA58B6C6267AACFDED3AA2030EFB7E4BA148F92DA2D
3E4D1DECAB5D32EF457FCB2BE4CDD3FE5D6A0A0E0C1A242EDD24F"
        }
    }
}
```


Refer to classes definition for parsing result: TRANSPORT RESULT

4.3. TRANSPORT REFUND

Requesting transport refund from SwitchioPay**.** The result will be available after the consumer device receives **"FinishedPlain"** in **/status** status polling. Supported from version 4.7.0

4.3.1. Request

GSA v7 - POST ip:port/api/switchio/pay/v7/transport/refund
GSA v8 - POST ip:port/api/switchio/pay/v8/transport/refund

Request example:


```
{
   "secureString":"123456",
   "request":{
      "transactionId":"88bb2366-716d-11ed-a1eb-0242ac120002",
      "transportAmount":100,
      "transportCurrencyCode":978,
      "invoiceNumber":"invoice",
      "transportData": "q100-12_123",
      "additionalText":"text",
```

```
      "virtualTerminalId":"TERMINAL"
   }
}
```

PARAMETER	TYPE	DESCRIPTION
secureString	string	Secured string, which allows users to access data from the terminal
transactionId	string	Identifier of transport payment transaction, which the cash register wants to create
invoiceNumber	string	OPTIONAL Variable symbol, max length is 20 char
transportAmount	int	OPTIONAL Transaction amount shown on the terminal/validator display when the card is tapped. The amount is sent in the minimum currency unit. For example, if the currency is Kč 10,00, amount = „1000“; EUR 1,50 = „150“ (without decimal point).
transportData	string	Optional, Base64 encoded string data from validator
transportCurrencyCode	int	OPTIONAL Currency code visualized on the terminal/validator display when the card is tapped.
additionalText	string	OPTIONAL Additional text
virtualTerminalId	string	OPTIONAL ID of the virtual terminal id. Required only when virtualization is enabled.

Parameters of the Transport refund request

4.3.2. Response

StatusCode: 200 → Transaction has started, cash register is polling **/status** request until it receives "**FinishedPlain**" state

StatusCode: 202 → Request was processed, but there is some transaction on the terminal that needs to be handled first - server busy.

Response example:

```
{
    "transactionId":"1e6a1251-762f-45fc-85fb-227a1c3290b6",
    "isStarted":true,
    "status":"Other payment in progress"
}
```

After cash register received "**FinishedPlain",** cash register should request result of the transaction with using endpoint **/result**, SwitchioPay will respond with following response:

Result Response for transport payment


```
{
    "processingResultCode":0,
    "message":"ok",
    "fareTransactionRequest":{
        "device":{
            "type":"NEXGO",
            "reader_id":"TN86RTA1",
            "reader_sn":"N860W006694",
            "deny_list_version":""
        },
        "transaction":{

"token":"11688528C822DAA6C0757EA0F5E647055452A83E5F90109ADD226471D89DBEB960",
            "id":"6c1c7e21-7805-49cc-8045-07cd8d9e3a3c",
            "dttm":"2022-10-03T12:53:01Z",
            "result":"NO_DENYLIST"
        },
        "fare_data":"test_transport_data",
        "emv_data":{
        "tags":[
            {"key":"9F06","value":"A0000000031010"},
            {"key":"5F24","value":"2208"}
        ],
        "encrypted":{
            "encryption_method":"MNSP",

"encrypted_data":"49727E9FEECAC356BA58B6C6267AACFDED3AA2030EFB7E4BA148F92DA2D
3E4D1DECAB5D32EF457FCB2BE4CDD3FE5D6A0A0E0C1A242EDD24F"
        }
    }
}
```


Refer to classes definition for parsing result: TRANSPORT RESULT

4.4. TRANSPORT REVERSAL

Requesting transport reversal from SwitchioPay**.** The result will be available after the consumer device receives **"FinishedPlain"** in **/status** status polling. Supported from version 4.7.0 of SwitchioPay

The request starts the reversal of the transaction specified by **originalTransactionId** .

The state of the transaction should be queried periodically by calling **/status**. Once the cash register receives a **"Finished"** state from the status request, the app must request the result of the transaction through **/result** request.

4.4.1. Request

GSA v7 - POST ip:port/api/switchio/pay/v7/transport/reversal
GSA v8 - POST ip:port/api/switchio/pay/v8/transport/reversal

Request example:

```
{
    "secureString":"123456",
    "request":{
        "transactionId":"6c1c7e21-7805-49cc-8045-10000001",
        "originalTransactionId":"6c1c7e21-7805-49cc-8045-1000000",
        "additionalText":"additional text",
        "virtualTerminalId":"TERM0004",
    }
}
```

PARAMETER	TYPE	DESCRIPTION
secureString	string	Secured string, which allows users to access data from the terminal
transactionId	string	Identifier of transport payment transaction, which the cash register wants to create
originalTransactionID	string	Identifier of original transport transaction to be reversed
additionalText	string	OPTIONAL Additional text
virtualTerminalId	string	OPTIONAL ID of the virtual terminal id. Required only when virtualization is enabled.

Parameters of the Transport refund request

4.4.2. Response

Result Response for transport reversal

```
{"resultCode":0,"message":"ok"}
```

Result Response for transport reversal

```
@Serializable data class TransportReversalResult(
      val resultCode: Int,
      val message: String
) {
```

```
        companion object {
                const val CODE_SUCCESS = 0
                const val CODE_INVALID_REQUEST = 1
        }
}
```

4.5. TRANSPORT TAP

Method for creating transport tap transaction, which will check the token from the tapped card with a deny list. The result will be available after the consumer device receives "**FinishedPlain**" in **/status** status polling.

4.5.1. Request

GSA v2 - POST ip:port/paya/transport/tap
GSA v2 - POST ip:port/api/switchio/pay/v2/transport/tap
GSA v4 - POST ip:port/api/switchio/pay/v4/transport/tap
GSA v5 - POST ip:port/api/switchio/pay/v5/transport/tap
GSA v6 - POST ip:port/api/switchio/pay/v6/transport/tap
GSA v7 - POST ip:port/api/switchio/pay/v7/transport/tap
GSA v8 - POST ip:port/api/switchio/pay/v8/transport/tap

Request example:

```
{
    "secureString":"123456",
    "request": {
        "transactionId":"6c1c7e21-7805-49cc-8045-07cd8d9e3a3c"
    }
}
```

PARAMETER	TYPE	DESCRIPTION
transactionId	string	Identifier of transport payment transaction, which the cash register wants to create
secureString	string	Secured string, which allows users to access data from the terminal

Parameters of the Transport Tap request

4.5.2. Response

StatusCode: 200 → Transaction has started, cash register is polling **/status** request until it receives "**FinishedPlain**" state

StatusCode: 202 → Request was processed, but there is some transaction on the terminal that needs to be handled first - server busy.

Response example:

```
{
    "transactionId":"1e6a1251-762f-45fc-85fb-227a1c3290b6",
    "isStarted":true,
    "status":"Other payment in progress"
}
```

After cash register received "**FinishedPlain",** cash register should request result of the transaction with using endpoint **/result**, SwitchioPay will respond with following response:

Result Response for transport tap

```
{"processingResultCode":8,"message":"Invalid or missing parameters"}
```

Class definition for parsing result of transport TAP method:

Result Response for transport tap

```
@Serializable
data class CustomerTapResult(
    val processingResultCode: Int,
    val message: String,
    val response: PairTokenResponse? = null
) {
    companion object {
        const val CODE_SUCCESS = 0
        const val CODE_INVALID_INPUT_DATA = 1
        const val CODE_CARD_READ_ERROR = 3
        const val CODE_TOKEN_KEY_ERROR = 4
        const val CODE_TIMEOUT = 5
        const val CODE_INVALID_RESPONSE = 6
        const val CODE_GENERAL_ERROR = 7
        const val CODE_INVALID_PARAMETERS = 8

        val INVALID_DATA: CustomerTapResult =
            CustomerTapResult(CODE_INVALID_INPUT_DATA, "invalid input data")
           val TIMEOUT: CustomerTapResult = CustomerTapResult(CODE_TIMEOUT,
"Timeout")
        val INVALID_RESPONSE: CustomerTapResult =
            CustomerTapResult(CODE_INVALID_RESPONSE, "Invalid Response")
          val GENERAL_ERROR = CustomerTapResult(CODE_GENERAL_ERROR, "General
error")
    }
}


@Serializable
data class PairTokenResponse(
    val result: String,
    val resultMessage: String = "",
    val code: String,
```

```
)
```

4.6. TRANSPORT UPLOAD

Uploading transport transactions to the server and downloading the most recent deny list from the server to SwitchioPay. The result will be available after the consumer device receives "**FinishedPlain**" in **/status** status polling.

4.6.1. Request

GSA v1 - POST ip:port/paya/transport/upload
GSA v2 - POST ip:port/api/switchio/pay/v2/transport/upload
GSA v4 - POST ip:port/api/switchio/pay/v4/transport/upload
GSA v5 - POST ip:port/api/switchio/pay/v5/transport/upload
GSA v6 - POST ip:port/api/switchio/pay/v6/transport/upload
GSA v7 - POST ip:port/api/switchio/pay/v7/transport/upload
GSA v8 - POST ip:port/api/switchio/pay/v8/transport/upload

Request example:

```
{
   "transactionId":"6c1c7e21-7805-49cc-8045-07cd8d9e3a3c",
   "secureString":"asdf"
}
```

PARAMETER	TYPE	DESCRIPTION
transactionId	string	Identifier of transport payment transaction, which the cash register wants to create

Parameters of the Transport Upload request

4.6.2. Response

StatusCode: 200 → transaction has started, cash register is polling **/status** request until it receives "**FinishedPlain**" state

StatusCode: 202 → Request was processed, but there is some transaction on the terminal that needs to be handled first - server busy.

Response example:

```
{
    "transactionId":"1e6a1251-762f-45fc-85fb-227a1c3290b6",
    "isStarted":true,
    "status":"Other payment in progress"
}
```

After cash register received "**FinishedPlain",** cash register should request result of the transaction with using endpoint **/result**, SwitchioPay will respond with following response:

Result Response for transport upload

```
{"resultCode":0,"message":"ok"}
```

Class definition for parsing result of transport TAP method:

Result Response for transport upload

```
@Serializable
data class UploadResult(val resultCode: Int, val message: String) {

    companion object {
        const val CODE_SUCCESS = 0
        const val CODE_UPLOAD_ERROR = 1
        const val CODE_PARAM_ERROR = 2
           const val CODE_BLACKLIST_ERROR = 3 // Transactions were uploaded
though!
        const val CODE_UPLOAD_NO_TRANSACTIONS_FOUND = 4
        const val CODE_UPLOAD_CANCELLED = 5
    }
}
```

4.7. TOKEN

Requesting for a list of tokens of the tapped card**.** The result will be available after the consumer device receives **"FinishedPlain"** in **/status** status polling.

4.7.1. Request

GSA v1 - POST ip:port/paya/token
GSA v2 - POST ip:port/api/switchio/pay/v2/token
GSA v4 - POST ip:port/api/switchio/pay/v4/token
GSA v5 - POST ip:port/api/switchio/pay/v5/token
GSA v6 - POST ip:port/api/switchio/pay/v6/token
GSA v7 - POST ip:port/api/switchio/pay/v7/token
GSA v8 - POST ip:port/api/switchio/pay/v8/token

Request example without amount:

```
{
  "transactionId": "55627324-dd68-465c-9f34-a2854398e468",
  "secureString": "asdf"
}
```

Request example with amount:

```
{
  "transactionId": "55627324-dd68-465c-9f34-a2854398e468",
  "secureString": "asdf",
  "amount":  1000,
  "currencyCode": 203, //CZK
}
```

PARAMETER	TYPE	DESCRIPTION
transactionId	string	Identifier of transport payment transaction, which the cash register wants to create
secureString	String	Secured string, which allows users to access data from the terminal
amount	Int	OPTIONAL Transaction amount shown on the terminal/validator display when the card is tapped. The amount is sent in the minimum currency unit. For example, if the currency is Kč 10,00, amount = „1000“; EUR 1,50 = „150“ (without decimal point).
currencyCode	Int	OPTIONAL Currency code visualized on the terminal/validator display when the card is tapped.

Parameters of the Token request

4.7.2. Response

StatusCode: 200 → transaction has started, cash register is polling **/status** request until it receives **"FinishedPlain"** state

StatusCode: 202 → Request was processed, but there is some transaction on the terminal that needs to be handled first - server busy.

Response example:

```
{
    "transactionId":"1e6a1251-762f-45fc-85fb-227a1c3290b6",
    "isStarted":true,
    "status":"Other payment in progress"
}
```

After cash register received "**FinishedPlain",** cash register should request result of the transaction with using endpoint **/result**, SwitchioPay will respond with following response:

Result Response for token

```
{
   "resultCode":0,
   "tokens":[
      "11688528C822DAA6C0757EA0F5E647055452A83E5F90109ADD226471D89DBEB960"
   ],
   "maskedCardNumber":"451161******4821",
   "expiration":"2208"
}
```

Class definition for parsing result of transport TAP method:

Result Response for token

```
@Serializable
data class TokenResponse(
    val resultCode: Int, //TokenResponseCode#value
    val tokens: List<String>,
    val maskedCardNumber: String,
    val expiration: String,
)

enum class TokenResponseCode(val value: Int){
      SUCCESS(0),
      UNKNOWN_ERROR(1),
      UNKNOWN_CARD(2),
      TIMEOUT(3),
      CANCELLED(4),
      DISCOVERY_ERROR(5),
}
```

4.8. SCAN

Request for scanning the QR/Barcode code with a terminal camera. It can be canceled by **/cancel** endpoint while state is "**WaitForUserInput**".

4.8.1. Request

GSA v1 - POST ip:port/paya/scan
GSA v2 - POST ip:port/api/switchio/pay/v2/scan
GSA v4 - POST ip:port/api/switchio/pay/v4/scan
GSA v5 - POST ip:port/api/switchio/pay/v5/scan
GSA v6 - POST ip:port/api/switchio/pay/v6/scan
GSA v7 - POST ip:port/api/switchio/pay/v7/scan
GSA v8 - POST ip:port/api/switchio/pay/v8/scan

Request example:

```
{
   "transactionId":"55627324-dd68-465c-9f34-a2854398e468",
   "secureString":"asdf"
}
```

PARAMETER	TYPE	DESCRIPTION
transactionId	string	Identifier of transport payment transaction, which the cash register wants to create
secureString	string	Secured string, which allows users to access data from the terminal

Parameters of the Scan request

4.8.2. Response

StatusCode: 200 → scanning has started, cash register is polling **/status** request until cash register received **"FinishedPlain"** state

StatusCode: 202 → Request was processed, but there is some transaction on the terminal that needs to be handled first - server busy.

Response example:

```
{
    "transactionId":"1e6a1251-762f-45fc-85fb-227a1c3290b6",
    "isStarted":true,
    "status":"Other payment in progress"
}
```

After cash register received "**FinishedPlain",** cash register should request result of the transaction with using endpoint **/result**, SwitchioPay will respond with following response:

4.9. TRANSPORT RESULT

Definition of classes that are required to parse result from transport payment, refund and reversal endpoints

Result Response

```
@Serializable
data class TransportTransactionResult(
    val processingResultCode: Int,
    val message: String,
    val fareTransactionRequest: FareTransactionRequest? = null,
) {
    companion object {
        const val CODE_SUCCESS = 0
        const val CODE_INVALID_INPUT_DATA = 1
        const val CODE_PARAM_ERROR = 2
        const val CODE_BLACKLIST_ERROR = 3 // not used
        const val CODE_TRANSACTION_NOT_FOUND = 4
        const val CODE_CARD_READ_ERROR = 5
        const val CODE_TOKEN_KEY_ERROR = 6
        const val CODE_BLACKLIST_NOT_FOUND = 7
    }
}

@Serializable
data class FareTransactionRequest(
    val device: Device,
    val transaction: FareTransaction,
    @SerialName("fare_data")
    val fareData: String,
    @SerialName("emv_data")
    val emvData: FareEmvData,
)

@Serializable
data class FareTransaction(
    val token: String,
    @SerialName("id")
    val transactionId: String,
    @SerialName("dttm")
    @Serializable(FareTransactionRequestDateSerializer::class)
    val dateTime: Date,
    val result: TokenDenylistResult,
    @SerialName("detail_result")
    val detailResult: String = "",
)
```

```
enum class TokenDenylistResult {
    OK,
```

```
    FAIL,
    EXPIRED,
    DENYLISTED,
    NOTIFY_DENYLIST,
    NO_DENYLIST,
}
```

```
@Serializer(forClass = Date::class)
class FareTransactionRequestDateSerializer : KSerializer<Date> {

    private val dateFormat: SimpleDateFormat =
        SimpleDateFormat("yyyy-MM-dd'T'HH:mm:ss'Z'", Locale.getDefault())

    override val descriptor: SerialDescriptor =
        PrimitiveSerialDescriptor("DenylistDate", PrimitiveKind.STRING)

    override fun serialize(encoder: Encoder, value: Date) =
        encoder.encodeString(dateFormat.format(value))

              override    fun    deserialize(decoder:    Decoder):    Date    =
checkNotNull(dateFormat.parse(decoder.decodeString()))
}

@Serializable
data class FareEmvData(
    val tags: List<Tag>,
    val encrypted: FareEncryptedData,
)


@Serializable
data class Tag(val key: String, val value: String)

@Serializable
data class FareEncryptedData(
    @SerialName("encryption_method")
    val encryptionMethod: EncryptionMethod,
    @SerialName("encrypted_data")
    val encryptionData: String,
    @SerialName("encryption_seed")
    val encryptionSeed: String? = null,
)
```

```
enum class EncryptionMethod {
    DUKPT,
    MNSP,
    TDES;
}
```

4.10. SCAN RESULT

Scan result has to be used to obtain results from the scanner, it has to be requested by the cash register when the cash register receives the **"FinishedPlain"** state of the scanning process from **/status** endpoint.

4.10.1. Request

GSA v1 - POST ip:port/paya/scan_result
GSA v2 - POST ip:port/api/switchio/pay/v2/scan_result
GSA v4 - POST ip:port/api/switchio/pay/v4/scan_result
GSA v5 - POST ip:port/api/switchio/pay/v5/scan_result
GSA v6 - POST ip:port/api/switchio/pay/v6/scan_result
GSA v7 - POST ip:port/api/switchio/pay/v7/scan_result
GSA v8 - POST ip:port/api/switchio/pay/v8/scan_result

Request example:

```
{
  "transactionId": "55627324-dd68-465c-9f34-a2854398e468",
  "secureString": "asdf"
}
```

4.10.2. Response

StatusCode: 200 → Plain data

StatusCode: 404 → Transaction with given transactionId is not recognized by SwitchioPay

Response example:

```
"data_from_qr"
```

4.11. GET TAPS (from v5)

Method returns list of taps stored in the terminal which are ready to be uploaded.
Since Switchio PAY 4.2.0.

4.11.1. Request

GSA v5 - POST ip:port/api/switchio/pay/v5/get_taps
GSA v6 - POST ip:port/api/switchio/pay/v6/get_taps
GSA v7 - POST ip:port/api/switchio/pay/v7/get_taps
GSA v8 - POST ip:port/api/switchio/pay/v8/get_taps

Request example:

```
{
    "secureString":"password"
}
```

PARAMETER	TYPE	DESCRIPTION
secureString	string	Password of the terminal

Parameters of the Get taps V5 request

4.11.2. Response - transaction with 200

List of taps. If there are no available taps, the list is empty

Response example:

```
{
   "token":"token12345",
   "id":"7d1c083d-19f3-4d7b-aa14-f615ed500a85",
   "dttm":"2023-09-05T08:49:55Z",
   "result":"OK",
   "detail_result":"result",
   "masked_pan":"1234****1234"
}
```

PARAMETER	TYPE	DESCRIPTION
token	string	Card token
id	string	Tap ID
dttm	date	Date and time of the tap
result	string	TokenDenylistResult
detail_result	string	Result message

masked_pan	string	Masked PAN of the payment card

Data object of Tap

```
@Serializable
data class FareTransaction(
    val token: String,
    @SerialName("id")
    val transactionId: String,
    @SerialName("dttm")
                    @Serializable(FareTransactionRequestDateSerializer::class)
//"yyyy-MM-dd'T'HH:mm:ss'Z'"
    val dateTime: Date,
    val result: TokenDenylistResult,
    @SerialName("detail_result")
    val detailResult: String = "",
    @SerialName("masked_pan")
    val maskedPan: String = "",
)

enum class TokenDenylistResult {
    OK,
    FAIL,
    EXPIRED,
    DENYLISTED,
    NOTIFY_DENYLIST,
    NO_DENYLIST,
}
```

4.12. GET DENYLIST LAST UPDATE (from v5)

4.12.1. Request

GSA v5 - POST ip:port/api/switchio/pay/v5/denylist_last_update
GSA v6 - POST ip:port/api/switchio/pay/v6/denylist_last_update
GSA v7 - POST ip:port/api/switchio/pay/v7/denylist_last_update
GSA v8 - POST ip:port/api/switchio/pay/v8/denylist_last_update

Method returns object with the date of last denylist update. Date is in format "yyyy-MM-dd HH:mm:ss".
Since Switchio PAY 4.2.0.

Request example:

```
{
   "secureString":"password"
}
```

PARAMETER	TYPE	DESCRIPTION
secureString	string	Password of the terminal

Parameters of the Denylist last update V5 request

4.12.2. Response - transaction with 200

Object with single attribute date. If there was no denylist update, the date is null.

Response example:

```
{
   "date":"2023-09-06 10:01:02"
}
```

Data object of result

```
@Serializable
data class GsaDenylistLastUpdateResponse(
    val date: String?,
)
```

5. ENUMS

5.1. STATUS CODES

StatusCode

```
200 - OK
201 - CREATED
202 - ACCEPTED (BUT SERVER IS BUSY)
400 - ALREADY PROCESSED
401 - UNAUTHORIZED
404 - NOT FOUND
405 - METHOD NOT ALLOWED
408 - REQUEST TIMEOUT
500 - INTERNAL SERVER ERROR
501 - NOT IMPLEMENTED
503 - SERVICE NOT AVAILABLE
```

5.2. CARD INPUT

CardInputType

```
MAGNETIC
MAGNETIC_FALLBACK
CONTACT
CLESS
CLESS_MAGSTRIPE
```

5.3. CVM TYPE

CvmTypeList

```
NO
SIGNATURE
PIN_ONLINE
PIN_OFFLINE
CONSUMER_DEVICE
```

5.4. TRANSACTION TYPES

TransactionType

```
SALE_ONLINE
SALE_OFFLINE
PREAUTHORIZATION
PREAUTHORIZATION_COMPLETION
PREAUTHORIZATION_INCREMENT
PREAUTHORIZATION_DECREMENT
EXTERNAL_PREAUTHORIZATION_INCREMENT
EXTERNAL_PREAUTHORIZATION_DECREMENT
EXTERNAL_PREAUTHORIZATION_COMPLETION
RETURN_ONLINE
RETURN_OFFLINE
REFERRAL_RETURN
```

```
MOTO_SALE
MOTO_PREAUTHORIZATION
CLOSE_BATCH
SUBTOTAL_BATCH
HANDSHAKE
UNDEFINED
CARD_VERIFY
TRANSACTION_STATUS
QR_SALE
```

5.5. TRANSACTION STATUS

```
WaitForCard
WaitForPin
WaitForUserInput
Connecting
Connected
Disconnected
Sending
Receiving
Finished
Idle
FinishedPlain
```

5.6. TERMINAL ERROR CODES

RESULTCODE			
ID	**RESPONSE CODE**	**DESCRIPTION (RESPONSE MESSAGE)**	**RESULT**
0	OK	All processing is successful	✔
1	KERNEL_FAILED	Initialization of kernel failed	X
2	KERNEL_NOT_ACCESS	Kernel isn't accessible	X
3	TEXT_KERNEL_NOT_INIT	Kernel not initialized	X
4	BLUETOOTH_NOT_SUPPORTED	Bluetooth not supported	X
5	PINBLOCK_TIMEOUT	Timeout for read pinblock	X
6	PINBLOCK_USER_CANCELLED	User cancel a pinblock	X
7	CANCELLED	Default user cancel	X
8	FAILED	Default failure	X
9	NO_PAPER	No paper available	X
10	UNKNOWN_MERCHANT	Unknown merchant	X
11	UNKNOWN	Transaction has not been	X

		completed	
501	SCA_PIN_REQUIRED	Passcode and biometric capable. It is not possible to insert a contact card, it is just contactless.	**X**
1000	SPDH_DEFINE		**-**
1050	DeclinedGeneral	General decline.	**X**
1050	DeclinedProductGroup	Product group not allowed.	**X**
1050	DeclinedCardBlocked	The card is blocked.	**X**
1050	DeclinedCardBlockedByRange	The card is blocked by range.	**X**
1050	DeclinedAccountNotAllowedOnStation	The use of an account card is not allowed on the given gas station.	**X**
1050	DeclinedProductGroupForStation	Product group not allowed for the given gas station.	**X**
1050	DeclinedCardDailyTxCountLimitExceed	The card has exceeded the daily maximum number of transactions.	**X**
1050	DeclinedCardWeeklyTxCountLimitExceed	The card has exceeded the weekly maximum number of transactions.	**X**
1050	DeclinedCardMonthlyTxCountLimitExceed	The card has exceeded the monthly maximum number of transactions.	**X**
1050	DeclinedCardDailyFuelLimitExceed	The card has exceeded the daily fuel limit.	**X**
1050	DeclinedCardWeeklyFuelLimitExceed	The card has exceeded the weekly fuel limit.	**X**
1050	DeclinedCardMonthlyFuelLimitExceed	The card has exceeded the monthly fuel limit.	**X**
1050	DeclinedCardDailyMoneyLimitExceed	The card has exceeded the daily money limit.	**X**
1050	DeclinedCardWeeklyMoneyLimitExceed	The card has exceeded the weekly money limit.	**X**
1050	DeclinedCardMonthlyMoneyLimitExceed	The card has exceeded its monthly cash limit.	**X**
1050	DeclinedCardTxPriceLimitExceed	The card has exceeded the maximum cash limit per transaction.	**X**
1050	DeclinedCardDailyOtherLimitExceed	The card has exceeded the daily money limit on dry goods.	**X**
1050	DeclinedCardWeeklyOtherLimitExceed	The card has exceeded the weekly money limit on dry goods.	**X**
1050	DeclinedCardMonthlyOtherLimitExce	The card has exceeded the	**X**

	ed	monthly money limit on dry goods.	
1050	DeclinedCardTxFuelAmountLimitExceed	The card has exceeded the maximum amount of fuel per transaction.	X
1050	DeclinedCardTxOtherPriceLimitExceed	The card has exceeded the maximum price limit for dry goods per transaction.	X
1051	DeclinedExpiredCard	Expired card.	X
1052	DeclinedNumberOfPINTriesExceeded	Number of PIN tries exceeded.	X
1053	DeclinedNoSharingAllowed	No sharing allowed.	X
1054	DeclinedNoSecurityModule	No security module.	X
1055	DeclinedInvalidTransaction	Invalid transaction.	X
1056	DeclinedNotSupportedByInstitution	Transaction not supported by institution.	X
1057	DeclinedLostOrStolenCard	Lost or stolen card.	X
1058	DeclinedInvalidCardStatus	Invalid card status.	X
1059	DeclinedRestrictedStatus	Restricted status.	X
1060	DeclinedAccountNotFoundInCardholderDatabase	Account not found in cardholder database.	X
1061	DeclinedPositiveBalanceAccountRecordNotFound	Positive balance account record not found.	X
1061	DeclinedPositiveBalanceAccountRecordNotFound	Positive balance account update error.	X
1063	DeclinedInvalidAuthorizationTypeInInstitutionDatabase	Invalid authorization type in institution database.	X
1064	DeclinedBadTrackInformation	Bad track information.	X
1065	DeclinedAdjustmentNotAllowedInInstitutionDatabase	Adjustment not allowed in the institution database.	X
1066	DeclinedInvalidCreditCardAdvanceIncrement	Invalid credit card advance increment.	X
1067	DeclinedInvalidTransactionDate	Invalid transaction date.	X
1068	DeclinedTransactionLogFileError	Transaction log file error.	X
1069	DeclinedBadMessageEdit	Bad message edit.	X
1070	DeclinedNoInstitutionDatabaseRecord	No institution database record.	X
1071	DeclinedInvalidRoutingToHostAppplication	Invalid routing to host application.	X

1072	DeclinedCardOnNationalNegativeFile	Card on national negative file.	X
1073	DeclinedInvalidRoutingAuthorizationService	Invalid routing authorization service.	X
1074	DeclinedUnableToAuthorize	Unable to authorize.	X
1074	DeclinedUnableToAuthorizeRepeat	Unable to authorize, retry transaction.	X
1075	DeclinedInvalidPANLength	Invalid PAN length.	X
1076	DeclinedInsufficientFundsInPositiveBalanceAccount	Insufficient funds in a positive balance account.	X
1077	DeclinedPreauthorizationFull	Preauthorization full.	X
1078	DeclinedDuplicateTransactionReceived	Duplicate transaction received.	X
1079	DeclinedMaximumOnlineRefundReached	Maximum online refund reached.	X
1080	DeclinedMaximumOfflineRefundReached	Maximum offline refund reached.	X
1081	DeclinedMaximumCreditPerRefundReached	Maximum credit per refund reached.	X
1082	DeclinedMaximumNumberOfTimesUsed	Maximum number of times used.	X
1083	DeclinedMaximumRefundCreditReached	Maximum refund credit reached.	X
1084	DeclinedCustomerSelectedNegativeCardFileReason	Customer selected negative card file reason.	X
1085	DeclinedInquiryNotAllowedNoBalances	Inquiry not allowed—no balances.	X
1086	DeclinedOverFloorLimit	Over floor limit.	X
1087	DeclinedMaximumNumberRefundCreditsReached	Maximum number refund credits reached.	X
1088	DeclinedPlaceCall	Place call.	X
1089	DeclinedCardStatusIsInactiveOrClosed	Card status equals 0 (inactive) or 9 (closed).	X
1090	DeclinedReferralFileFull	Referral file full.	X
1091	DeclinedProblemAccessingNegativeCardFile	Problem accessing negative card file.	X
1092	DeclinedAdvanceLessThanMinimum	Advance less than minimum.	X
1093	DeclinedDelinquent	Delinquent.	X

1094	DeclinedOverLimitTableOrExceedsAmountAvailable	Over limit table or exceeds the amount available.	X
1095	DeclinedAmountOverMaximum	Amount over maximum.	X
1096	DeclinedPINRequired	PIN required.	X
1097	DeclinedMod10Check	Mod 10 check.	X
1098	DeclinedForcePost	Force post.	X
1099	DeclinedCouldNotAccessPositiveBalanceAccountInDatabase	Could not access positive balance account in database	X
1100	ReferralUnableToProcessTransaction	Unable to process a transaction.	X
1101	ReferralUnableToAuthorizeIssueCall	Unable to authorize—issue call.	X
1102	ReferralCall	Call.	X
1103	ReferralProblemAccessingNegativeCardFile	Problem accessing negative card file.	X
1104	ReferralProblemAccessingCardholderAccount	Problem accessing cardholder account.	X
1105	ReferralCardNotSupported	Card not supported.	X
1105	ReferralCardNotFound	Card not found.	X
1105	ReferralCardReplenishmentSupported	Card replenishment not supported.	X
1106	ReferralAmountOverMaximum	Amount over maximum.	X
1107	ReferralOverDailyLimit	Over daily limit.	X
1108	ReferralCardAuthorizationParametersNotFound	Card authorization parameters not found.	X
1109	ReferralAdvanceLessThanMinimum	Advance less than minimum.	X
1110	ReferralNumberTimesUsed	Number times used.	X
1111	ReferralDelinquent	Delinquent.	X
1112	ReferralOverLimitTable	Over limit table	X
1113	ReferralTimeout	Timeout.	X
1115	ReferralransactionLogFileFull	Transaction log file full.	X
1120	ReferralProblemAccessingCardholderUsageAccumulationData	Problem accessing cardholder usage accumulation data.	X
1121	ProblemAccessingAdministrativeCardData	Problem accessing administrative card data.	X
1122	ReferralUnableToValidatePIN	Unable to validate PIN; security module is down.	X

1130	ReferralAuthorizationRequestCryptogram	Authorization request cryptogram (ARQC) referral.	X
1131	ReferralCardVerificationResults	Card verification results (CVR) referral.	X
1132	ReferralTerminalVerificationResults	Terminal verification results (TVR) referral.	X
1133	ReferralReasonOnlineCode	Reason online code referral.	X
1134	ReferralFallback	Fallback referral.	X
1150	ServiceMerchantNotOnFile	Merchant not on file.	X
1150	ServiceMerchantTerminalDisabled	Blocked terminal.	X
1150	ServiceMerchantAcquirerDisabled	Blocked terminal owner.	X
1200	TransactionInvalidAccount	Invalid account.	X
1201	TransactionIncorrectPIN	Incorrect PIN.	X
1202	TransactionCashAdvanceIsLessThanMinimum	Cash advance is less than minimum.	X
1203	TransactionAdministrativeCardNeeded	Administrative card needed.	X
1204	TransactionEnterLesserAmount	Enter a lesser amount (This code can also be used when the transaction amount exceeds the retailer ceiling limits.).	X
1205	TransactionInvalidAdvanceAmount	Invalid advance amount.	X
1206	TransactionCardholderAuthorizationDataNotFound	Cardholder authorization data not found.	X
1207	TransactionInvalidTransactionDate	Invalid transaction date.	X
1208	TransactionInvalidExpirationDate	Invalid expiration date.	X
1209	TransactionInvalidTransactionCode	Invalid transaction code.	X
1251	TransactionCashBackExceedsDailyLimit	Cash back exceeds daily limit.	X
1400	TransactionAuthorizationRequestCryptogramFailure	Authorization request cryptogram (ARQC) failure.	X
1401	TransactionHardwareSecurityModuleParameterError	Hardware security module parameter error.	X
1402	TransactionHardwareSecurityModuleFailure	Hardware security module failure.	X
1403	TransactionIntegratedCircuitCardKeyInformationNotFound	Integrated circuit card key information not found.	X
1404	TransactionApplicationTransactionC	Application transaction counter	X

	ounterCheckFailure	(ATC) check failure.	
1405	TransactionCardVerificationResultsDecline	Card verification results (CVR) decline.	X
1406	TransactionTerminalVerificationResultsDecline	Terminal verification results (TVR) decline.	X
1407	TransactionReasonOnlineCodeDecline	Reason online code decline.	X
1408	TransactionFallbackDecline	Fallback decline.	X
1500	ProductDenied	Unauthorized product.	X
1557	SuspectedFraudCallWag	WAG antifraud. Suspected fraud, call WAG	X
1558	LockedCardSendSMS	WAG antifraud. Card locked, for unlock card send SMS to WAG	X
1800	TransactionFormatError	Format error.	X
1801	TransactionInvalidData	Invalid data.	X
1802	TransactionInvalidEmployeeNumber	Invalid employee number.	X
1809	TransactionInvalidCloseTransaction	Invalid close transaction.	X
1810	TransactionTimeout	Transaction timeout.	X
1811	TransactionSystemError	System error.	X
1811	SuperiorAuthorizationSystemError	Authorization failed on the master authorization system.	X
1820	TransactionInvalidTerminalIdentifier	Invalid terminal identifier.	X
1821	TransactionInvalidResponseLength	Invalid response length.	X
1882	DownloadAborted	Download aborted (call for service).	X
1878	DeclineIncorrectPINLengthError	Incorrect PIN length error.	X
1889	DeclineMACCommunicationsKeySynchronizationError	MAC communications key (KMAC) synchronization error.	X
1898	DeclineInvalidMAC	Invalid MAC.	X
1899	DeclineSequenceErrorResync	Sequence error—resync.	X
1900	POSCaptureNumberOfPINTriesExceeded	Number of PIN tries exceeded.	X
1901	POSCaptureExpiredCard	Expired card.	X
1902	POSCaptureNegativeCardFileCaptureCode	Negative card file capture code.	X
1903	POSCaptureCardStatusIsStolen	Card status is 3 (stolen).	X

1904	POSCaptureAdvanceLessThanMinimum	Advance less than minimum.	X
1905	POSCaptureNumberTimesUsedExceeded	Number times used exceeded.	X
1906	POSCaptureDelinquent	Delinquent.	X
1907	POSCaptureOverLimitTable	Over limit table.	X
1908	POSCaptureAmountOverMaximum	Amount over maximum.	X
1950	DeclinedAdministrativeCardNotFound	Administrative card not found.	X
1951	DeclinedAdministrativeCardNotAllowed	Administrative card not allowed.	X
1959	DeclinedAdministrativeTransactionsNotSupported	Administrative transactions not supported.	X
1956	ChargebackCustomerFileUpdatedAcquirerNotFound	Chargeback—customer file updated—acquirer not found.	X
1957	ChargebackIncorrectPrefixNumber	Chargeback—incorrect prefix number.	X
1958	ChargebackIncorrectResponseCodeOrCardPrefixConfiguration	Chargeback—incorrect response code or card prefix configuration.	X
1960	ChargebackApprovedCustomerFileNotUpdated	Chargeback—approved customer file not updated.	X
1961	ChargebackApprovedCustomerFileNotUpdatedAcquirerNotFound	Chargeback—approved customer file not updated, acquirer not found.	X
1962	ChargebackAcceptedIncorrectDestination	Chargeback—accepted, incorrect destination.	X
1963	NoCardTimeout	No card timeout	X
1964	InvalidSignature	Invalid signature	X
2000	PAX_DEFINE	ReturnCodes by PAX	-
2001	ICC_RESET_ERR	IC cardreset is failed	X
2002	ICC_CMD_ERR	IC card command is failed	X
2003	ICC_BLOCK	IC card is blocked	X
2004	EMV_RSP_ERR	Status Code returned by IC card is not 9000	X
2005	EMV_APP_BLOCK	The application selected is blocked	X
2006	EMV_NO_APP	There is no AID matched between ICC and terminal	X
2007	EMV_USER_CANCEL	The current operation or	X

		transaction was canceled by user	
2008	EMV_TIME_OUT	User's operation is timeout	X
2009	EMV_DATA_ERR	Data error is found	X
2010	EMV_NOT_ACCEPT	Transaction is not accepted	X
2011	EMV_DENIAL	Transaction is denial	X
2012	EMV_KEY_EXP	Certification Authority Public Key is Expired	X
2013	EMV_NO_PINPAD	PIN enter is required, but PIN pad is not present or not working	X
2014	EMV_NO_PASSWORD	PIN enter is required, PIN pad is present, but there is no PIN entered	X
2015	EMV_SUM_ERR	Checksum of CAPK is error	X
2016	EMV_NOT_FOUND	Appointed Data element can't be found	X
2017	EMV_NO_DATA	The length of the appointed Data Element is 0	X
2018	EMV_OVERFLOW	Memory is overflow	X
2019	NO_TRANS_LOG	There is no Transaction log	X
2020	RECORD_NOTEXIST	Appointed log is not existed	X
2021	LOGITEM_NOTEXIST	Appointed Label is not existed in current log record	X
2022	ICC_RSP_6985	Status Code returned by IC card for GPO is 6985	X
2023	CLSS_USE_CONTACT	Must use other interface for the transaction	X
2024	EMV_FILE_ERR	There is file error found	X
2025	CLSS_TERMINATE	Must terminate the transaction	X
2026	CLSS_FAILED	Contactless transaction is failed	X
2027	CLSS_DECLINE	Transaction should by declined	X
2028	CLSS_TRY_ANOTHER_CARD	Try another card (DPAS Only)	X
2030	CLSS_PARAM_ERR	Parameter is error = EMV_PARAM_ERR	X
2031	CLSS_WAVE2_OVERSEA	International transaction (for VISA AP payWaveLevel2 IC card use)	X
2033	CLSS_WAVE2_US_CARD	US card (for VISA AP payWaveL2 IC card use)	X

2034	CLSS_WAVE3_INS_CARD	Need to use IC card for the transaction (for VISA payWaveIC card use)	X
2035	CLSS_RESELECT_APP	Select the next AID in candidate list	X
2036	CLSS_CARD_EXPIRED	IC card is expired	X
2037	EMV_NO_APP_PPSE_ERR	Select PPSE command returns error code, or these is no matching Application	X
2038	CLSS_USE_VSDC	Switch to Contactless PBOC	X
2039	CLSS_CVMDECLINE	DVM result is decline	X
2040	CLSS_REFER_CONSUMER_DEVICE	Status Code returned by IC card is 6986, please see phone	X
2041	CLSS_LAST_CMD_ERR	Terminal did not receive the response of the last command of Read Record. (qPBOC only)	X
2042	CLSS_API_ORDER_ERR	APIs are called in wrong order. Please call Clss_GetDebugInfo_xxx to get error codes.	X
2043	CLSS_TORN_CARDNUM_ERR	Torn log's PAN is different with the reselect card's pan	X
2044	CLSS_TRON_AID_ERR	Torn log's AID is different with the reselect card's AID	X
2045	CLSS_TRON_AMT_ERR	Torn log's amount is different with the current transaction's amount	X
2046	CLSS_CARD_EXPIRED_REQ_ONLINE	Card expired and the transaction should go online	X
2047	CLSS_FILE_NOT_FOUND	ICC return 6A82 (File not found) in response to the SELECT command	X
2048	CLSS_TRY_AGAIN	Try again	X
2200	CLSS_PAYMENT_NOT_ACCEPT	Payment Type Not Accepted for flash	X
2301	CLSS_ISSERTED_ICCCARD	IC card is detected during contactless transaction	X
2302	CLSS_SWIPED_MAGCARD	Magnetic stripe card is detected during contactless transaction	X
3000	MONEMV_CLS_KERNEL	Monet kernel	-
3257	MONEMV_CLS_KERNEL_STATUS_SERVICE_NOT_AVAILABLE	The kernel to be accessed is not loaded in the terminal.	X
3258	MONEMV_CLS_KERNEL_STATUS_DATABASE_ERROR	A database error occurred.	X

3259	MONEMV_CLS_KERNEL_STATUS_INVALID_INPUT_DATA	One of the provided data is incorrect.	X
3260	MONEMV_CLS_KERNEL_STATUS_NOT_SUPPORTED	The called function is not supported by the kernel.	X
3261	MONEMV_CLS_KERNEL_STATUS_LACK_OF_MEMORY	There is not enough memory to complete the transaction.	X
3262	MONEMV_CLS_KERNEL_STATUS_COMMUNICATION_ERROR	A communication error occurred with the contactless card.	X
3263	MONEMV_CLS_KERNEL_STATUS_MISSING_INPUT_DATA	A mandatory data is missing to perform the transaction.	X
3264	MONEMV_CLS_KERNEL_STATUS_ICC_MISSING_DATA	A mandatory ICC data is missing to perform the transaction.	X
3265	MONEMV_CLS_KERNEL_STATUS_ICC_INVALID_DATA	The data returned by the card is not correctly formatted.	X
3266	MONEMV_CLS_KERNEL_STATUS_ICC_REDUNDANT_DATA	Card returned redundant data.	X
3267	MONEMV_CLS_KERNEL_STATUS_ICC_DATA_FORMAT_ERROR	The card response is not correctly formatted (parsing error, empty response, etc).	X
3268	MONEMV_CLS_KERNEL_STATUS_TERM_MISSING_DATA	A mandatory terminal data is missing to perform the transaction.	X
3269	MONEMV_CLS_KERNEL_STATUS_CARD_BLOCKED	A Card is blocked and the transaction cannot be performed.	X
3270	MONEMV_CLS_KERNEL_STATUS_APPLICATION_BLOCKED	The application in the card is blocked.	X
3271	MONEMV_CLS_KERNEL_STATUS_REMOVE_AID	The AID shall be removed from the candidate list.	X
3272	MONEMV_CLS_KERNEL_STATUS_UNKNOWN_SW	Unexpected response code from the ICC.	X
3273	MONEMV_CLS_KERNEL_STATUS_COND_OF_USE_NOT_SATISFIED	Conditions of use not satisfied" in the GPO response.	X
3275	MONEMV_CLS_KERNEL_STATUS_OFFLINE_DECLINED	Transaction is offline declined.	X
3276	MONEMV_CLS_KERNEL_STATUS_ONLINE_AUTHORISATION	Transaction succeeded and needs an online authorisation to be completed.	-
3277	MONEMV_CLS_KERNEL_STATUS_CANCELLED	Transaction has been cancelled.	X
3278	MONEMV_CLS_KERNEL_STATUS_USE_CONTACT_INTERFACE	The transaction has to be conducted over another interface (chip or swipe for example)	X
3279	MONEMV_CLS_KERNEL_STATUS_NOT	This function cannot be called as	X

	_ALLOWED	no transaction is in process with this kernel or cancellation is not allowed according to the transaction progress.	
3280	MONEMV_CLS_KERNEL_STATUS_CONTINUE	The transaction flow must continue.	-
3281	MONEMV_CLS_KERNEL_STATUS_SUSPEND	The transaction flow must be stopped.	X
3282	MONEMV_CLS_KERNEL_STATUS_INTERNAL_ERROR	A kernel internal error occurred.	X
3283	MONEMV_CLS_KERNEL_STATUS_LIB_INTERFACE_ERROR	An error occurred into the kernel interface (linked to the custom application).	X
3284	MONEMV_CLS_KERNEL_STATUS_EXPIRED_CERTIFICATE	The certificate used for Offline data authentication is expired.	X
3285	MONEMV_CLS_KERNEL_STATUS_REVOKED_CERTIFICATE	The certificate used for Offline data authentication is revoked.	X
3286	MONEMV_CLS_KERNEL_STATUS_CARD_UNKNOWN	The contactless card accessed isn't the same as at the beginning of the transaction. This error occurs for a new card presentation as Issuer Script Processing ...")	X
3287	MONEMV_CLS_KERNEL_STATUS_MOBILE	A mobile phone GPO response has been detected. This error allows to restart a selection application cycle with new user message (for mobile phone) ...")	X
3383	MONEMV_CLS_KERNEL_STATUS_UNKNOWN	Unknown	X
4000	FEITIAN_READER	Feitian reader	-
8005	DOMESTIC_CARD_ONLY_ERROR	This error code is used for the Cashback operation and indicates that for this operation, only domestic cards can be used. This means that the terminal country code has to be the same as the card country code. If the card country code differs from the terminal country code, this error occurs.	X
8008	READING_CARD_FAILED	Error when card was not processed correctly	X
8009	SECOND_TAP_CARD_MISMATCH	Error occurred when a different card was used for the required second tap of the transaction.	X
2147483647	UNDEFINED	Undefined	-