Payment Platform Rafinita

Integration API (POST)

version 2.4

Table of Content

1 Introduction

This document describes integration procedures and POST protocol usage for e-commerce merchants.

POST protocol implements acquiring payments (purchases) with specific API interaction using.

2 Integration process

2.1 Merchant registration

Before you get an account to access Payment Platform, you must provide the following data to the Payment Platform administrator.

Data	Description
IP list	List of your IP addresses, from which requests to Payment Platform will be sent
Notification URL	URL which will be receiving the notifications of the processing results of your request to Payment Platform. It is mandatory if your account supports 3D-Secure. The length of Notification URL should not be more than 255 symbols.
Contact email	Email address of Responsible Person who will monitor transactions, conduct refunds, etc.

With all Payment Platform POST requests at Notification URL the Merchant must return the string "OK" if he/she successfully received data or return "ERROR".

You should get the following information from administrator to begin working with the Payment Platform.

Data	Description
CLIENT_KEY	Unique key to identify the account in Payment Platform (used as request parameter) In the administration platform this parameter corresponds to the “Public key” field
CLIENT_PASS	Password for Client authentication in Payment Platform (used for calculating hash parameter) In the administration platform this parameter corresponds to the “Public secret” field
PAYMENT_URL	URL to request the Payment Platform

2.2 Brief description of the interaction with Payment Platform

For the transaction, you must send the server to server HTTPS POST request with fields listed below to Payment Platform URL (PAYMENT_URL). In response Payment Platform will return the JSON (http://json.org/) encoded string.

If your account supports 3D-Secure and credit card supports 3D-Secure, then Payment Platform will return the link to the 3D-Secure Access Control Server to perform 3D-Secure verification. In this case, you need to redirect the cardholder at this link. If there are also some parameters except the link in the result, you will need to redirect the cardholder at this link together with the parameters using the method of data transmitting indicated in the same result.

In the case of 3D-Secure after verification on the side of the 3D-Secure server, the owner of a credit card will come back to your site using the link you specify in the sale request, and Payment Platform will return the result of transaction processing to the Notification URL action.

2.3 List of possible actions in Payment Platform

When you make request to Payment Platform, you need to specify action that needs to be done. Possible actions are:

Action	Description

Action	Description
SALE	Creates SALE or AUTH transaction
CAPTURE	Creates CAPTURE transaction
CREDITVOID	Creates REVERSAL or REFUND transaction
GET_TRANS_STATUS	Gets status of transaction in Payment Platform
GET_TRANS_DETAILS	Gets details of the order from Payment platform
RECURRING_SALE	Creates SALE or AUTH transaction using previously used cardholder data

Following actions cannot be made by request, they are initiated by Payment Platform in certain circumstances (e.g. issuer initiated chargeback) and you receive callback as a result.

Action	Description
CHARGEBACK	CHARGEBACK transaction was created in Payment Platform

2.4 List of possible transaction results and statuses

Result – value that system returns on request. Possible results are:

Result	Description
SUCCESS	Action was successfully completed in Payment Platform
DECLINED	Result of unsuccessful action in Payment Platform
REDIRECT	Additional action required from requester (Redirect to 3ds)
ACCEPTED	Action was accepted by Payment Platform, but will be completed later
ERROR	Request has errors and was not validated by Payment Platform

Status – actual status of transaction in Payment Platform. Possible statuses are:

Status	Description
3DS	The transaction awaits 3D-Secure validation
REDIRECT	The transaction is redirected
PENDING	The transaction awaits CAPTURE
PREPARE	Status is undetermined, final status will be sent in callback
SETTLED	Successful transaction
REVERSAL	Transaction for which reversal was made
REFUND	Transaction for which refund was made
CHARGEBACK	Transaction for which chargeback was made
DECLINED	Not successful transaction

3 Card transactions requests

3.1 SALE request

Payment Platform supports two main operation type: Single Message System (SMS) and Dual Message System (DMS).

SMS is represented by SALE transaction. It is used for authorization and capture at a time. This operation is commonly used for immediate payments.

DMS is represented by AUTH and CAPTURE transactions. AUTH is used for authorization only, without capture. This operation used to hold the funds on card account (for example to check card validity).

SALE request is used to make both SALE and AUTH transactions.

If you want to make AUTH transaction, you need to use parameter auth with value Y.

If you want to send a payment for the specific sub-account (channel), you need to use channel_id, that specified in your Payment Platform account settings.

This request is sent by POST in the background (eg, through PHP CURL).

3.1.1 Request parameters

Parameter	Description	Values	Required field
action	Sale	SALE	+
client_key	Unique key (CLIENT_KEY)	UUID format value	+
channel_id	Payment channel (Sub-account)	String up to 16 characters	-
order_id	Transaction ID in the Merchants system	String up to 255 characters	+
order_amount	The amount of the transaction	Numbers in the form XXXX.XX (without leading zeros)	+
order_currency	Currency	3-letter code	+
order_description	Description of the transaction (product name)	String up to 1024 characters	+
card_number	Credit Card Number		+
card_exp_month	Month of expiry of the credit card	Month in the form XX	+
card_exp_year	Year of expiry of the credit card	Year in the form XXXX	+
card_cvv2	CVV/CVC2 credit card verification code	3-4 symbols	+
payer_first_name	Customer's name	String up to 32 characters	+
payer_last_name	Customer's surname	String up to 32 characters	+
payer_middle_name	Customer's middle name	String up to 32 characters	-
payer_birth_date	Customer's birthday	format yyyy-MM-dd, e.g. 1970-02-17	-
payer_address	Customer's address	String up to 255 characters	+

Parameter	Description	Values	Required field
payer_address2	The adjoining road or locality (if required) of the customer's address	String up to 255 characters	-
payer_country	Customer's country	2-letter code	+
payer_state	Customer's state	String up to 32 characters	-
payer_city	Customer's city	String up to 32 characters	+
payer_zip	ZIP-code of the Customer	String up to 10 characters	+
payer_email	Customer's email	String up to 256 characters	+
payer_phone	Customer's phone	String up to 32 characters	+
payer_ip	IP-address of the Customer	XXX.XXX.XXX.XXX	+
term_url_3ds	URL to which Customer should be returned after 3D-Secure	String up to 1024 characters	+
recurring_init	Initialization of the transaction with possible following recurring	Y or N (default N)	-
auth	Indicates that transaction must be only authenticated, but not captured	Y or N (default N)	-
hash	Special signature to validate your request to Payment Platform	* (Appendix A)	+

3.1.2 Response parameters

You will get JSON encoded string (see an example on Appendix B) with transaction result. If your account supports 3D-Secure, transaction result will be sent to your Notification URL.

3.1.2.1 Synchronous mode

Successful sale response

Parameter	Description
action	SALE
result	SUCCESS
status	PENDING / PREPARE / SETTLED; only PENDING when auth=Y
order_id	Transaction ID in the Merchant's system
trans_id	Transaction ID in the Payment Platform
trans_date	Transaction date in the Payment Platform
descriptor	Descriptor from the bank, the same as cardholder will see in the bank statement
recurring_token	Recurring token (get if account support recurring sales and was initialization transaction for following recurring)
amount	Order amount

Parameter	Description
currency	Currency

Unsuccessful sale response

Parameter	Description
action	SALE
result	DECLINED
status	DECLINED
order_id	Transaction ID in the Merchant’s system
trans_id	Transaction ID in the Payment Platform
trans_date	Transaction date in the Payment Platform
descriptor	Descriptor from the bank, the same as cardholder will see in the bank statement
amount	Order amount
currency	Currency
decline_reason	The reason why the transaction was declined

3D-Secure transaction response

Parameter	Description
action	SALE
result	REDIRECT
status	3DS / REDIRECT
order_id	Transaction ID in the Merchant’s system
trans_id	Transaction ID in the Payment Platform
trans_date	Transaction date in the Payment Platform
descriptor	Descriptor from the bank, the same as cardholder will see in the bank statement
amount	Order amount
currency	Currency
redirect_url	URL to which the Merchant should redirect the Customer
redirect_params	Object of specific 3DS parameters
redirect_method	The method of transferring parameters (POST/GET)

3.1.2.2 Callback parameters

Successful sale response

Parameter	Description
action	SALE
result	SUCCESS
status	PENDING / PREPARE / SETTLED
order_id	Transaction ID in the Merchant’s system
trans_id	Transaction ID in the Payment Platform
trans_date	Transaction date in the Payment Platform
descriptor	Descriptor from the bank, the same as cardholder will see in the bank statement
recurring_token	Recurring token (get if account support recurring sales and was initialization transaction for following recurring)
amount	Order amount
currency	Currency
hash	Special signature, used to validate callback **(Appendix A)

Unsuccessful sale response

Parameter	Description
action	SALE
result	DECLINED
status	DECLINED
order_id	Transaction ID in the Merchant’s system
trans_id	Transaction ID in the Payment Platform
trans_date	Transaction date in the Payment Platform
decline_reason	Description of the cancellation of the transaction
hash	Special signature, used to validate callback **(Appendix A)

3D-Secure transaction response

Parameter	Description
action	SALE
result	REDIRECT
status	3DS/ REDIRECT
order_id	Transaction ID in the Merchant’s system
trans_id	Transaction ID in the Payment Platform
trans_date	Transaction date in the Payment Platform

Parameter	Description
redirect_url	URL to which the Merchant should redirect the Customer
descriptor	Descriptor from the bank, the same as cardholder will see in the bank statement
amount	Order amount
currency	Currency
redirect_params	Object with the parameters
redirect_method	The method of transferring parameters (POST/GET)
hash	Special signature, used to validate callback **(Appendix A)

3.2 CAPTURE request

CAPTURE request is used to submit previously authorized transaction (created by SALE request with parameter auth = Y). Hold funds will be transferred to Merchants account.

This request is sent by POST in the background (eg, through PHP CURL).

3.2.1 Request parameters

Parameter	Description	Values	Required field
action	Capture previously authenticated transaction	CAPTURE	+
client_key	Unique key (CLIENT_KEY)	UUID format value	+
trans_id	Transaction ID in the Payment Platform	UUID format value	+
amount	The amount for capture. Only one partial capture is allowed	Numbers in the form XXXX.XX (without leading zeros)	-
hash	Special signature to validate your request to payment platform	**(Appendix A)	+

3.2.2 Response parameters

3.2.2.1 Synchronous mode

Successful capture response

Parameter	Description
action	CAPTURE
result	SUCCESS
status	SETTLED
amount	Amount of capture
order_id	Transaction ID in the Merchant’s system
trans_id	Transaction ID in the Payment Platform

Parameter	Description
trans_date	Transaction date in the Payment Platform
descriptor	Descriptor from the bank, the same as cardholder will see in the bank statement
currency	Currency

Unsuccessful capture response

Parameter	Description
action	CAPTURE
result	DECLINED
status	PENDING
order_id	Transaction ID in the Merchant's system
trans_id	Transaction ID in the Payment Platform
trans_date	Transaction date in the Payment Platform
descriptor	Descriptor from the bank, the same as cardholder will see in the bank statement
amount	Amount of capture
currency	Currency
decline_reason	The reason why the capture was declined

3.2.2.2 Callback parameters

Successful capture response

Parameter	Description
action	CAPTURE
result	SUCCESS
status	SETTLED
order_id	Transaction ID in the Merchant's system
trans_id	Transaction ID in the Payment Platform
amount	Amount of capture
trans_date	Transaction date in the Payment Platform
descriptor	Descriptor from the bank, the same as cardholder will see in the bank statement
currency	Currency
hash	Special signature, used to validate callback **(Appendix A)

Unsuccessful capture response

Parameter	Description
action	CAPTURE
result	DECLINED
status	PENDING
order_id	Transaction ID in the Merchant's system
trans_id	Transaction ID in the Payment Platform
decline_reason	The reason why the capture was declined
hash	Special signature, used to validate callback **(Appendix A)

3.3 CREDITVOID request

CREDITVOID request is used to complete both REFUND and REVERSAL transactions.

REVERSAL transaction is used to cancel hold from funds on card account, previously authorized by AUTH transaction.

REFUND transaction is used to return funds to card account, previously submitted by SALE or CAPTURE transactions.

This request is sent by POST in the background (eg, through PHP CURL).

3.3.1 Request parameters

Parameter	Description	Values	Required field
action	CREDITVOID	CREDITVOID	+
client_key	Unique key (CLIENT_KEY)	UUID format value	+
trans_id	Transaction ID in the Payment Platform	UUID format value	+
amount	The amount of full or partial refund. If amount is not specified, full refund will be issued. If case of partial refund this parameter is required. Several partial refunds are allowed	Numbers in the form XXXX.XX (without leading zeros)	-
hash	Special signature to validate your request to Payment Platform	**(Appendix A)	+

3.3.2 Response parameters

3.3.2.1 Synchronous mode

Parameter	Description
action	CREDITVOID
result	ACCEPTED
order_id	Transaction ID in the Merchant's system

Parameter	Description
trans_id	Transaction ID in the Payment Platform

3.3.2.2 Callback parameters

Successful refund/reversal response

Parameter	Description
action	CREDITVOID
result	SUCCESS
status	REFUND/REVERSAL
order_id	Transaction ID in the Merchant's system
trans_id	Transaction ID in the Payment Platform
creditvoid_date	Date of the refund/reversal
amount	Amount of refund
hash	Special signature, used to validate callback**(Appendix A)

Unsuccessful refund/reversal response

Parameter	Description
action	CREDITVOID
result	DECLINED
order_id	Transaction ID in the Merchant's system
trans_id	Transaction ID in the Payment Platform
decline_reason	Description of the cancellation of the transaction
hash	Special signature, used to validate callback **(Appendix A)

3.4 GET_TRANS_STATUS request

Gets order status from Payment Platform. This request is sent by POST in the background (eg, through PHP CURL).

3.4.1 Request parameters

Parameter	Description	Values	Required field
action	GET_TRANS_STATUS	GET_TRANS_STATUS	+
client_key	Unique key (CLIENT_KEY)	UUID format value	+
trans_id	Transaction ID in the Payment Platform	UUID format value	+
hash	Special signature to validate your request to Payment Platform	**(Appendix A)	+

3.4.2 Response parameters

Parameter	Description
action	GET_TRANS_STATUS
result	SUCCESS
status	3DS / REDIRECT / PENDING / PREPARE / DECLINED / SETTLED / REVERSAL / REFUND / CHARGEBACK
order_id	Transaction ID in the Merchant's system
trans_id	Transaction ID in the Payment Platform
decline_reason	Reason of transaction decline. It shows for the transactions with the "DECLINED" status
recurring_token	Token for recurring. It shows when the next conditions are met for the SALE transaction: • transaction is successful • SALE request contained "recurring_init" parameter with the value "Y" • SALE request contained card data which was used for the first time

3.5 GET_TRANS_DETAILS request

Gets all history of transactions by the order. This request is sent by POST in the background (eg, through PHP CURL).

3.5.1 Request parameters

Parameter	Description	Values	Required field
action	GET_TRANS_DETAILS	GET_TRANS_DETAILS	+
client_key	Unique key (CLIENT_KEY)	UUID format value	+
trans_id	Transaction ID in the Payment Platform	UUID format value	+
hash	Special signature to validate your request to Payment Platform	**(Appendix A)	+

3.5.2 Response parameters

Parameter	Description
action	GET_TRANS_DETAILS
result	SUCCESS
status	3DS / REDIRECT / PENDING / PREPARE / DECLINED / SETTLED / REVERSAL / REFUND / CHARGEBACK
order_id	Transaction ID in the Merchant's system
trans_id	Transaction ID in the Payment Platform
name	Payer name

Parameter	Description
mail	Payer mail
ip	Payer IP
amount	Order amount
currency	Currency
card	Card in the format XXXXXX****XXXX
decline_reason	Reason of transaction decline. It shows for the transactions with the “DECLINED” status
recurring_token	Token for recurring. It shows when the next conditions are met for the SALE transaction: • transaction is successful • SALE request contained “recurring_init” parameter with the value “Y” • SALE request contained card data which was used for the first time
transactions [1]	Array of transactions with the parameters: • date • type (sale, 3ds, auth, capture, chargeback, reversal, refund) • status (success, waiting, fail) • amount

[1] **Example:**

```
{"transactions":[
{"date":"2012-01-01
01:10:25","type":"AUTH","status":"success","amount":"1.95"},{"date":"2012-01-01
01:11:30","type":"CAPTURE","status":"success","amount":"1.95"},{"date":"2012-02-
06 10:25:06","type":"REFUND","status":"success","amount":"1.95"}
```

4 RECURRING card transactions requests

4.1 RECURRING_SALE request

Recurring payments are commonly used to create new transactions based on already stored cardholder information from previous operations.

RECURRING_SALE request has same logic as SALE request, the only difference is that you need to provide primary transaction id, and this request will create a secondary transaction with previously used cardholder data from primary transaction.

This request is sent by POST in the background (eg, through PHP CURL).

4.1.1 Request parameters

Parameter	Description	Values	Required field
action	Recurring sale	RECURRING_SALE	+
client_key	Unique key (CLIENT_KEY)	UUID format value	+
order_id	Transaction ID in the Merchant's system	String up to 255 characters	+
order_amount	The amount of the transaction	Numbers in the form XXXX.XX (without leading zeros)	+
order_description	Transaction description (product name)	String up to 1024 characters	+
recurring_first_trans_id	Transaction ID of the primary transaction in the Payment Platform	UUID format value	+
recurring_token	Value obtained during the primary transaction	UUID format value	+
auth	Indicates that transaction must be only authenticated, but not captured	Y or N (default N)	-
hash	Special signature to validate your request to payment platform	*(Appendix A)	+

4.1.2 Response parameters

Response from Payment Platform is the same as by SALE command, except for the value of the difference parameter "action=RECURRING_SALE". You will receive a JSON encoded string with the result of the transaction.

5 Dispute transactions processing

CHARGEBACK transactions are used to dispute already settled payment.

When processing these transactions Payment Platform sends notification to Merchant's Notification URL.

5.1 CHARGEBACK notification parameters

Parameter	Description
action	CHARGEBACK
result	SUCCESS
status	CHARGEBACK
order_id	Transaction ID in the Merchant’s system
trans_id	Transaction ID in the Payment Platform
amount	The amount of the chargeback
chargeback_date	System date of the chargeback
bank_date	Bank date of the chargeback
reason_code	Reason code of the chargeback
hash	Special signature to validate callback **(Appendix A)

6 Errors

In case of an error you get synchronous response from Payment Platform:

Parameter	Description
result	ERROR
error_message	Error message
error_code	Error code

The list of the error codes is shown below.

Code	Description
204002	Enabled merchant mappings or MIDs not found.
204003	Payment type not supported.
204004	Payment method not supported.
204005	Payment action not supported.
204006	Payment system/brand not supported.
204007	Day MID limit is not set or exceeded.
204008	Day Merchant mapping limit is not set or exceeded.
204009	Payment type not found.
204010	Payment method not found.
204011	Payment system/brand not found.
204012	Payment currency not found.
204013	Payment action not found.
204014	Month MID limit is exceeded.
204015	Week Merchant mapping limit is exceeded.
208001	Payment not found.
208002	Not acceptable to request the 3ds for payment not in 3ds status.
208003	Not acceptable to request the capture for payment not in pending status.
208004	Not acceptable to request the capture for amount bigger than auth amount.
208005	Not acceptable to request the refund for payment not in settled or pending status.
208006	Not acceptable to request the refund for amount bigger than payment amount.
208008	Not acceptable to request the reversal for amount bigger than payment amount.
208009	Not acceptable to request the reversal for partial amount.
208010	Not acceptable to request the chargeback for amount bigger than payment's amount.

Code	Description
400	Duplicate request.
400	Previous payment not completed.

Sample of error-response


```
{
"result": "ERROR",
    "error_code": 100000,
    "error_message": "Request data is invalid.",
    "errors": [
        {
            "error_code": 100000,
            "error_message": "card_number: This value should not be blank."
        },
        {
            "error_code": 100000,
            "error_message": "card_exp_month: This value should not be blank."
        },
        {
            "error_code": 100000,
            "error_message": "card_exp_year: This value should not be blank."
        },
        {
            "error_code": 100000,
            "error_message": "card_cvv2: This value should not be blank."
        },
        {
            "error_code": 100000,
            "error_message": "order_id: This value should not be blank."
        },
        {
            "error_code": 100000,
            "error_message": "order_amount: This value should not be blank."
        },
        {
            "error_code": 100000,
            "error_message": "order_amount: This value should be greater than
0."
        },
        {
            "error_code": 100000,
            "error_message": "order_currency: This value should not be blank."
        },
        {
            "error_code": 100000,
            "error_message": "order_description: This value should not be
blank."
        },
        {
            "error_code": 100000,
            "error_message": "payer_first_name: This value should not be blank."
        },
        {
            "error_code": 100000,
            "error_message": "payer_last_name: This value should not be blank."
        },
        {
            "error_code": 100000,
            "error_message": "payer_address: This value should not be blank."
        },
        {
            "error_code": 100000,
```

```
            "error_message": "payer_country: This value should not be blank."
        },
        {
            "error_code": 100000,
            "error_message": "payer_city: This value should not be blank."
        },
        {
            "error_code": 100000,
            "error_message": "payer_zip: This value should not be blank."
        },
        {
            "error_code": 100000,
            "error_message": "payer_email: This value should not be blank."
        },
        {
            "error_code": 100000,
            "error_message": "payer_phone: This value should not be blank."
        },
        {
            "error_code": 100000,
            "error_message": "payer_ip: This value should not be blank."
        },
        {
            "error_code": 100000,
            "error_message": "term_url_3ds: This value should not be blank."
        }
    ]
}
```

7 Testing

You can make test requests using data below. Please note, that all transactions will be processed using Test engine.

Card number	Card expiration date (MM/YYYY)	Testing / Result
4111111111111111	01/2025	This card number and card expiration date must be used for testing successful sale Response on successful SALE request: "action": "SALE", "result": "SUCCESS", "status": "SETTLED" "action": "SALE", "result": "SUCCESS", "status": "SETTLED" Response on successful AUTH request: "action": "SALE", "result": "SUCCESS", "status": "PENDING"
4111111111111111	02/2025	This card number and card expiration date must be used for testing unsuccessful sale Response on unsuccessful SALE request: "action": "SALE", "result": "DECLINED", "status": "DECLINED" Response on unsuccessful AUTH request: "action": "SALE", "result": "DECLINED", "status": "DECLINED"
4111111111111111	03/2025	This card number and card expiration date must be used for testing unsuccessful CAPTURE after successful AUTH Response on successful AUTH request: "action": "SALE", "result": "SUCCESS", "status": "PENDING" Response on unsuccessful CAPTURE request: "action": "CAPTURE", "result": "DECLINED", "status": "PENDING"
4111111111111111	05/2025	This card number and card expiration date must be used for testing successful sale after 3DS verification Response on VERIFY request: "action": "SALE", "result": "REDIRECT", "status": "3DS" After return from ACS:

Card number	Card expiration date (MM/YYYY)	Testing / Result
		"action": "SALE", "result": "SUCCESS", "status": "SETTLED"
4111111111111111	06/2025	This card number and card expiration date must be used for testing unsuccessful sale after 3DS verification Response on VERIFY request: "action": "SALE", "result": "REDIRECT", "status": "3DS" After return from ACS: "action": "SALE", "result": "DECLINED", "status": "DECLINED"

8 Appendix A (Hash)

Hash is signature rule used either to validate your requests to payment platform or to validate callback from payment platform to your system. It must be md5 encoded string calculated by rules below:

* hash is calculated by the formula:

md5(strtoupper(strrev(email).CLIENT_PASS.

strrev(substr(card_number,0,6).substr(card_number,-4))))

** hash is calculated by the formula:

md5(strtoupper(strrev(email).CLIENT_PASS.

trans_id.strrev(substr(card_number,0,6).substr(card_number,-4))))

9 Appendix B (Examples)

Please review carefully the list of parameters before using the examples. Some parameters may be missing in the examples.

Requests examples are for reference only. If you will use them unchanged you will receive an error in the response.

You have to set your own values for parameters ("client_key" and "hash" in particular).

9.1 SALE request

9.1.1 Sample data of the sale request

Parameter	Valid value
action	SALE
client_key	c2b8fb04-110f-11ea-bcd3-0242c0a85004
order_id	ORDER-12345
order_amount	1.99
order_currency	USD
order_description	Product
card_number	4111111111111111
card_exp_month	01
card_exp_year	2025
card_cvv2	000
payer_first_name	John
payer_last_name	Doe
payer_address	Big street
payer_country	US
payer_state	CA
payer_city	City
payer_zip	123456
payer_email	doe@example.com
payer_phone	199999999
payer_ip	123.123.123.123
term_url_3ds [1]	http://client.site.com/return.php
hash	a1a6de416405ada72bb47a49176471dc

The hash above was calculated for CLIENT_PASS equal to *13a4822c5907ed235f3a068c76184fc3*.

Sample curl request

```
curl -d "action=SALE&client_key=c2b8fb04-110f-11ea-bcd3-
0242c0a85004&order_id=ORDER12345&order_amount=1.99&order_currency=USD&
order_description=Product&card_number=4111111111111111&card_exp_month=01&
card_exp_year=2025&card_cvv2=000&payer_first_name=John&payer_last_name=Doe&
payer_address=BigStreet&payer_country=US&payer_state=CA&payer_city=City&
payer_zip=123456&payer_email=doe@example.com&payer_phone=199999999&
payer_ip=123.123.123.123&term_url_3ds=http://client.site.com/return.php&
hash="a1a6de416405ada72bb47a49176471dc" https://test.apiurl.com -k
```

9.1.2 Sample response (synchronous mode)

The response if the sale is successful


```
{"action":"SALE","result":"SUCCESS","status":"SETTLED","order_id": "ORDER-
12345","trans_id":"aaaff66a-904f-11ea-833e-0242ac1f0007","trans_date":"2012-04-
03 16:02:01","descriptor":"test","amount":"0.02","currency":"USD"}
```


The response if the sale is unsuccessful


```
{"action":"SALE","result":"DECLINED","status":"DECLINED",
"order_id": "ORDER-12345","trans_id":"aaaff66a-904f-11ea-833e-0242ac1f0007",

"trans_date":"2012-04-03 16:02:01","decline_reason":"Declined by processing"}
```


The response if the transaction supports 3D-Secure


```
{"action":"SALE","result":"REDIRECT","status":"3DS","order_id":
"1588856266Intelligent","trans_id":"595ceeea-9062-11ea-aa1b-
0242ac1f0007","trans_date":2012-04-03
16:02:01","descriptor":"Descriptor","amount":"0.02","currency":"USD","redirect_u
rl":https://some.acs.endpoint.com","redirect_params":{"PaReq":"M0RTIE1hc3RlciBVU
0QgU1VDQ0VTUw==","MD":"595ceeea-9062-11ea-aa1b-
0242ac1f0007","TermUrl":"https://192.168.0.1:8101/verify/3ds/595ceeea-9062-11ea-
aa1b-0242ac1f0007/7d6b9b240ff2779b7209aef786f808d1"},"redirect_method":"POST"}
```


In case of an error (sample is shown in the “Errors” section)


```
{"result":"ERROR","error_message":"Error description"}
```


9.1.3 Sample response (asynchronous mode)

The response if the sale is successful


```
{"action":"SALE","result":""ACCEPTED","order_id":"ORDER-12345","trans_id":
"aaaff66a-904f-11ea-833e-0242ac1f0007","trans_date":"2012-04-03 16:02:01"}
```


In case of an error (sample is shown in the “Errors” section)


```
{"result":"ERROR","error_message":"Error description"}
```


9.2 Recurring sale request

9.2.1 Sample recurring sale request


```
curl -d "action=RECURRING_SALE&client_key=c2b8fb04-110f-11ea-bcd3-
0242c0a85004&order_id=ORDER-
12345&order_amount=1.99&order_description=Product&recurring_first_trans_id=aaaff
66a-904f-11ea-833e-0242ac&recurring_token=d6dcb9e0-96b6-11ea-bbd1-
0242ac120012&hash= a1a6de416405ada72bb47a49176471dc" https://test.apiurl.com -k
```


9.2.2 Sample response


```
{"ac-
tion":"RECURRING_SALE","result":"SUCCESS","status":"SETTLED","order_id":"ORDER-
12345","trans_id":"aaaff66a-904f-11ea-833e-0242ac1f0007","trans_date": "2012-04-
03 16:02:01", "descriptor":"test","amount":"0.02","currency":"USD"}
```


9.3 Creditvoid request


```
{"action=CREDITVOID&client_key=c2b8fb04-110f-11ea-bcd3-
0242c0a85004&trans_id=aaaff66a-904f-11ea-833e-
0242ac&amount=10.00&hash=6b957fca41c353ac344fcad47f0cbf97
"https://test.apiurl.com -k}
```


9.3.1 Sample response


```
{"action":"CREDITVOID","result":"ACCEPTED","trans_id":"aaaff66a-904f-11ea-833e-
0242ac","order_id":"ORDER-12345"}
```