

# myPOS Virtual API

---

*Accepting e-commerce payments for merchants*

*Version 1.0.4*

*Document version 1.0.4*

**myPOS™**

## Table of Contents

Table of Contents .....	2
Version control.....	3
Part 1: General Overview .....	4
Introduction .....	5
Scope .....	6
Structural features of the handbook.....	6
Terms and descriptions .....	7
Where to start.....	8
Payment Gateway details .....	8
Implementation basics .....	8
Use of Trademarks on Merchant's website .....	12
Currencies accepted at myPOS Virtual API .....	13
Security and availability .....	13
The functions of myPOS Virtual API .....	15
Business myPOS Account details .....	18
Integrating myPOS Virtual with online stores.....	18
Adding of an online store.....	18
E-commerce transactions .....	20
Reserve.....	20
Details.....	21
Frequently asked questions .....	22
Part 2: Technical integration .....	24
Overview .....	25
HTTP POST .....	25
Data type formats .....	25

---

Signature and authentications .....	25
Test environment .....	27
The process .....	30
Understanding transmission mechanism.....	30
Method standard properties.....	31
Response standard properties .....	31
The calls.....	31
Purchase with payment card (API call: IPCPurchase).....	32
Successful payment notification (API call: IPCPurchaseNotify / IPCPurchaseOK) .....	35
Cancellation of payment notification (API call: IPCPurchaseCancel).....	36
Rollback of previous notification (API call: IPCPurchaseRollback) .....	36
Get transaction log for previously executed payment (API call: IPCGetTxnLog) .....	36
Make a refund for previously executed payment (API call: IPCRefund) .....	43
Get transaction status for previously executed payment (API call: IPCGetTxnStatus).....	44
Appendix I – Error messages .....	45
Appendix II – Testing data.....	46
Test private key .....	46
myPOS test public certificate .....	46
Appendix III – Example for successful payment notification .....	47
Result POST data .....	47

## Version control

Date posted	API Version	Document revision	Description
19.08.2015	1.0	1	Version 1.0
13.11.2015	1.0	2	Review of POST requests examples
01.07.2016	1.0	3	
03.11.2016	1.0	4	

# Part 1: General Overview

## Introduction

myPOS Virtual API is a customer friendly interface for e-commerce payment solutions for your Online store. The API will let your customers to pay for goods and services quickly and securely using their debit or credit card.

The API will gain access to the entry point of iPayment Gateway managed by Satabank. By using hosted checkout secure pages, the merchant adheres to compliance rules for handling customer data in a secure way: data is stored on security servers so that it is not exposed to compromise.

The API enables you quickly and easily to integrate myPOS Virtual into all your online stores as a method of payment. The integration will give you the opportunity to receive money from your customers instantly, in multi-currencies, to track and manage your payments from all your online stores in a single place and to manage your online stores settings. Therewith the API will give convenience to all your customers to choose their preferred payment method from a single source.

Using the myPOS Virtual API your customers will be temporary redirected to <https://www.mypos.eu/vmypos/>. The iPayment Gateway will handle and guide the customer during the payment process, will check the card sensitive data and will process a payment transaction through card schemes (VISA, MasterCard, JCB).

Once the payment is complete, the customer will be returned to the merchant's website. You will receive notification about the payment along with the transaction details.

myPOS Virtual API will provide:

- Secured page and Secured communication channel with the merchant;
- Storing of merchant private data (shopping cart, amount, payment method, transaction details etc.);
- Financial transactions to VISA, MasterCard, JCB (if applicable);
- Operations for the front-end: Purchase transaction, Manage online stores settings;
- 3D secure processing for direct payments with Debit or Credit Cards.

## Scope

This handbook is aimed at the operators of e-commerce companies who would like to optimize their payment processes using an innovative and complete payment solution.

The purpose of this document is to specify the myPOS Virtual API Interface and demonstrate how it is used in the most common way, therewith to provide technical details about the system integration.

It is intended to be utilized by:

### *the merchant / commercial decision makers*

Certain sections describe the functions, main requirements and processes of the myPOS Virtual payment system from the stance of e-commerce Business development managers. The document provides guidance on what is a myPOS Virtual API, how it works, getting started, setting up a myPOS Account Number, main benefits of the API and FAQs.

### *the IT specialists and developers*

Other sections of this manual describe the technical background of the myPOS Virtual system. The document provides guidance on technical integration of the API with detailed working examples, use cases and error codes.

## Structural features of the handbook



Important notes are indicated by the pictogram on the left.



References to another point in the text or to another document are indicated by the pictogram on the left.

`<?xml .` Expressions or code examples are written in grey background with smaller font size.



## Terms and descriptions

**API** - Application Programming Interface for implementation of the myPOS Virtual system in the merchant e-commerce application.

**myPOS Account number** - Unique identifier for the Business Account of the merchant.

**Online store ID** – Unique Identifier for any online store (website) of the merchant which will use the myPOS Virtual API as a payment method.

**PAN** - Unique payment card number, which identifies the Issuer and the particular cardholder account.

**Call** – Call by the merchant's e-commerce system to myPOS Virtual API.

**Card Validation Code (CVC) or Card Verification Value (CVV)** – Three digits encrypted in the magnetic stripe on the back of the card. This is used as a method of verification during credit card processing.

**Request** - Query or request from the merchant e-commerce system to myPOS Virtual API

**Response** - Response from myPOS Virtual API to a Call from the e-commerce system.

**3-D Secure™ (3DS)** - Technical standard developed by Visa, MasterCard and JBC, designed to combat online credit card fraud. Cardholders who have registered for Verify by Visa®, MasterCard SecureCode® or J/Secure® use their password to validate their identity whenever they make a purchase on a participating site.

**XML** - XML (Extensible Markup Language) is a generally available data format which myPOS Virtual API uses for comfortable exchange with other systems.

## Where to start

In order to sign up as an myPOS Virtual API merchant and start using the iPayment Gateway, you need to have a Business myPOS Account first.

1. Sign in to your myPOS Account and go to the eCommerce / API.
2. Click on the link Activate.
3. Set up your online stores and generate the keys.



Important! Please have in mind that your stores need to be verified by Satabank which will take up to 5 business days. Until your online stores are verified, you can still process transactions but with certain limits.

## Payment Gateway details

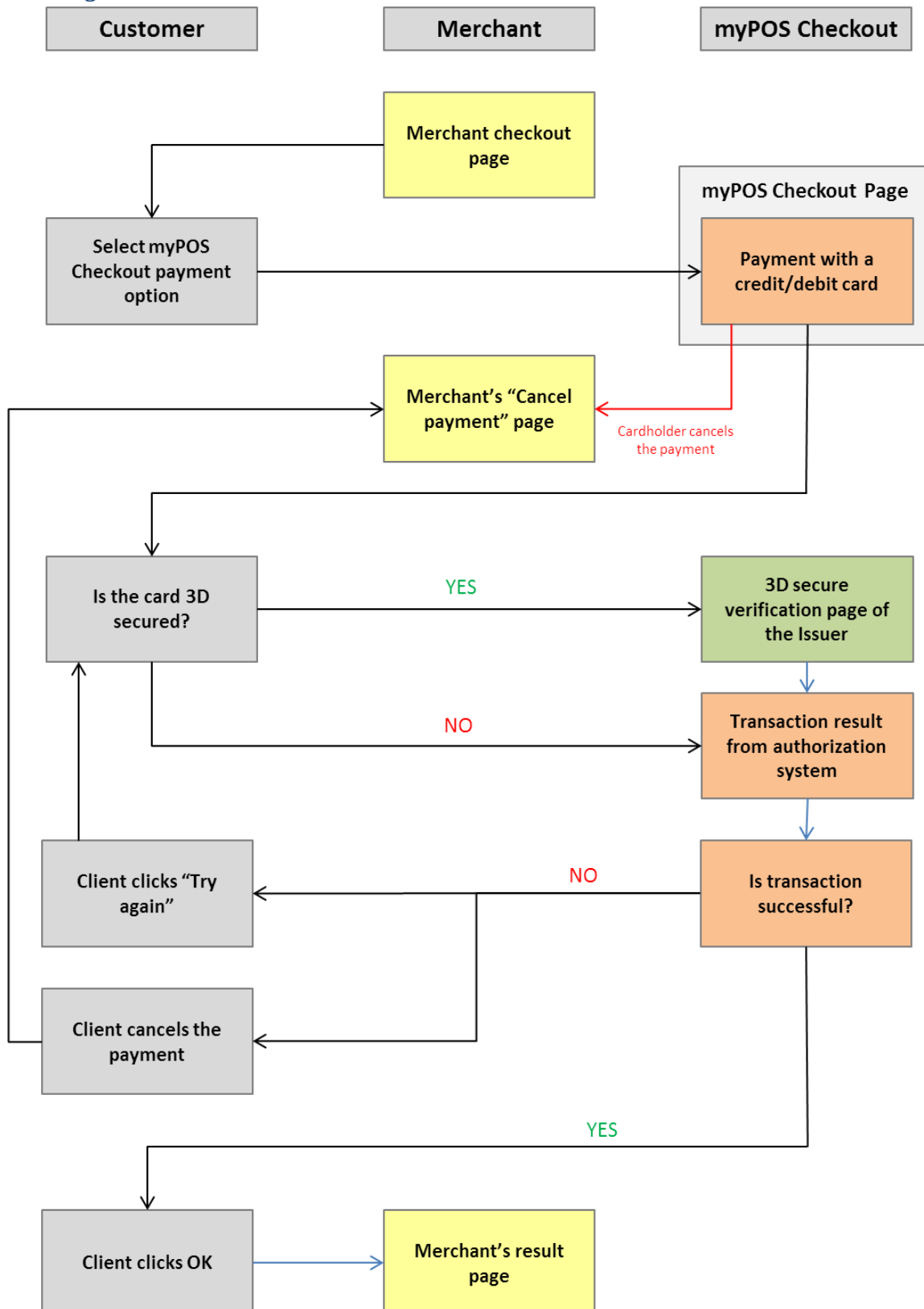
### Implementation basics

The API requires the merchants to modify their payment page at the shopping cart so to include myPOS Virtual as a payment option. When the customer chooses myPOS Virtual as a payment method he will actually submit a HTML form to the iPayment Gateway secure web servers using HTTPS protocol. The submitted form will contain all needed information about the payment (such as myPOS Account number, online store ID, shopping cart data, currency, amount, etc.)

Once the myPOS Virtual API processes the payment, the system will pass the result to merchant web server and the customer will be redirected to the online store “checkout result” page.



## Interaction Diagram





Detailed diagram with all communications and messages between the merchant website and myPOS Virtual API is presented in the [Part 2: Technical integration](#) of this document.

## *Payment process in steps*

1. Internet customer chooses myPOS Virtual as a payment method at the merchant's online store checkout page.
2. Merchant web server initiates payment through the iPayment Gateway.
3. Customer web browser is redirected to myPOS Virtual payment link along with the payment details.
4. If customer cancels the payment, he will be redirected to the merchant's "Cancel payment" page.
5. The customer needs to fill in his payment data and to click the button "Confirm".



Most of the customer data will be automatically filled in and the customer will need to complete just few of the fields.

6. iPayment gateway system receives the details for the payment – successful or declined.
7. iPayment gateway system passes the result to merchant web server and the customer web browser is directed to the myPOS Virtual result page.
8. If the customer clicks on the "Return to ..." button he will be redirected to the merchant's result page.

# Accepting e-commerce payments for merchants

myPOS Virtual API v.1.0.4



## myPOS Virtual payment page

The myPOS Virtual payment page will have the following main elements:

BANNER

### Order details

Order No: 1441116626

Item	Item price	Quantity	Amount
HP ProBook 6360b sticker	22.50 USD	2	45.00 USD
Delivery	5.00 USD	1	5.00 USD
Total:			50.00 USD

### Secure credit or debit card payment

Country:

Card number:

Valid Thru:  
 /

CVC:

Cardholder name:

ZIP / City:

Address:

Mobile number:

Example: +352XXXXXXX  
The country code and the phone number should be entered without spaces

Email:

\* All fields are mandatory

[Confirm](#)

[Cancel and return to dev.icards.eu](#)

If your debit or credit card support 3-D authentication, you may be taken to the 3-D secure verification page delivered by your bank.

+359 700 13 44

[support@mypos.eu](mailto:support@mypos.eu)



Most of the customer data fields will be pre-populated.

## ***Refund transactions***

The merchant could initiate a refund transaction for any previously executed payment. This functionality is available in his account pages in menu Transaction/ Transaction details.

In case where the customer has lodged a justified complaint or the merchant is unable to deliver the goods, he could refund any partial amount of the payment back to the customer.



Satabank always books the refund back in the originally used payment method.

The maximum refunded amount is 100 percent of the original amount.

## **Use of Trademarks on Merchant's website**

The Merchant needs to follow the instructions below:

### **1. General requirements**

The myPOS logo and all related with the myPOS Virtual API logos may not be altered, modified, or changed in any way, without the prior written permission of Satabank. These logos may not be used in a manner that refers disparagingly to Satabank or the myPOS Virtual service.

The myPOS Virtual payment option must be shown along with any other payment method on the merchant's website. The service must be presented in a manner which shows service in equality with all other payment methods.

The merchant may not use any of the Card scheme logos (VISA, MasterCard, JCB) separately without the myPOS logo, except in case where the merchant has personal agreement with a specific Card scheme.

The merchant must comply with Satabank any text descriptions of the myPOS Virtual service and/or the system before publishing them online.

The merchant may use on his website any logo available on mypos.eu website. If the merchant needs custom size or solution, he must contact our Customer service department.

### **2. Payment method page**

If the merchant has separate payment method page on his website, he needs to describe the myPOS Virtual service, complying with the above General requirements.

### **3. Checkout page**

The myPOS Virtual payment method must be presented on the merchant's checkout page complying with the above General requirements.

### **4. Redirecting page**

If the merchant has a redirecting page between his checkout page and the myPOS Virtual payment page, the merchant needs to have the following attributes on this page:

- <https://www.mypos.eu>
- Payments processed by mypos.eu or myPOS logo

Example:

*Please wait, you are now being redirected to [https://www.mypos.eu/vmp/...](https://www.mypos.eu/vmp/)*

*You are about to pay via secure payments processed by Satabank using a debit or credit card.*

*If you are not redirected within 5 seconds, please [click here](#).*

## 5. Other places on the merchant's website

The myPOS logos must be placed in clear and visible place on the homepage complying with the above General requirements.

The Verified by VISA & MasterCard SecureCode & J/Secure logos could be placed separately in accordance with the requirements of the Card scheme.

## Currencies accepted at myPOS Virtual API

myPOS Virtual API is accepting payments and making pay-outs in all currencies available for myPOS Accounts. Please refer to [www.mypos.eu](http://www.mypos.eu) in order to see the current list with all available currencies.

If the merchant needs to accept payments in any other currency, he will need to contact his customer service representative for more information.

## Security and availability

Connection between the merchant and the myPOS Virtual API is handled through internet using HTTPS protocol (SSL over HTTP). Requests and responses are digitally signed both. The host is located at Tier IV datacentre in Luxembourg. Public address for myPOS Virtual API is BGP enabled and available through all first level internet providers.

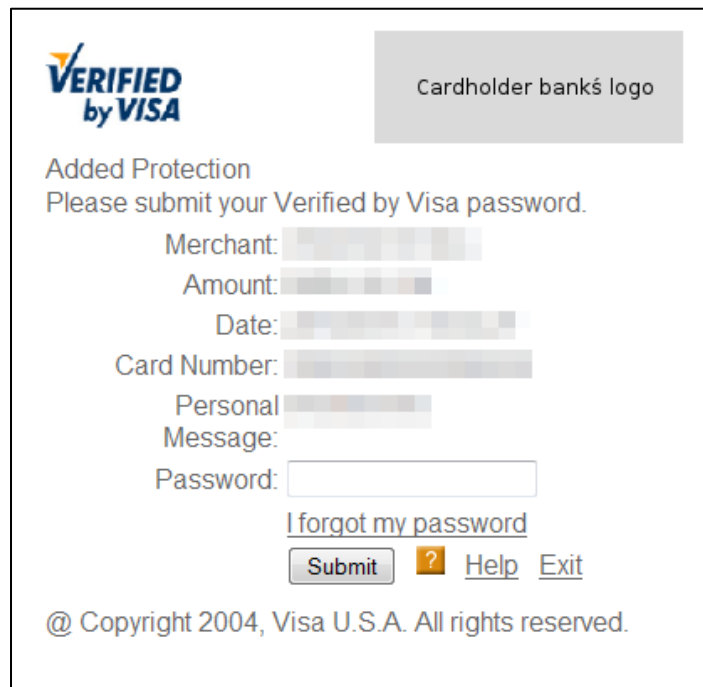
Our system supplies an emergency support line via e-mail or phone which is 7x24 enabled and reaches certified engineers.

## 3-D Secure payment

To make online transactions using credit cards safer and more secure, the system supports 3-D secure payments.

The service is available for merchant's accounts which support 3-D secure and in case the customer credit card is 3-D secure.

Depending on the Card scheme and the Issuing bank, the customer will see an additional step in the myPOS Virtual payment page. Please take a look at the VISA example below:



### ***Important security requirements for making requests to the myPOS Virtual API***

All requests to the API are standard HTTPS requests. The 'User-Agent' HTTP request header is required by myPOS Virtual API.

It is a means of verification of the program on the client host and if the client does not send this string, it cannot be verified nor logged and will result in myPOS Virtual error page with the following text: “The online store has sent myPOS a shopping cart with errors in it. We will contact the Merchant with a request to fix this problem. As this could be a temporary issue, you can go back to try checking out again.” and a link to the merchant’s website.

Sending the 'User-Agent' is one of the principle rules of our network security and is usually a simple setting in client programs. If you are against sending the header for tracking reasons, we inform you that this is used as a loophole by potential attackers.

### ***Security restrictions***

#### ***Enable/Disable payments for a specific merchant’s online store***

By default, the online payment processing for any approved merchant’s online store is disabled. To enable the store the merchant needs to log in his Business Account, to go to the eCommerce / Online stores menu and to click on the button “Enable” beside the particular online store.



The merchant could use the “Enable/ Disable” functionality at any convenient time.

## Request URLs

This myPOS feature aims to further increase the security level of the merchant's account, protecting it from unauthorized request attempts.

The merchant must specify at least one URL from which request to the myPOS Virtual API will be made.



All requests from any other URLs will be denied.

The merchant could add new URLs at any time, however all new URLs will be reviewed and approved first.

## Signature and public/private key pairs

In every message a signature is supplied.

For signing process, both myPOS Virtual API and the merchant generate public/private key pairs and exchange the public certificate. Key pairs are generated using RSA algorithm. The certificates must be PEM-encoded PKCS7 file. Every of the parties are using the private key to sign the message and the opposite side authenticate the sender with corresponding public certificate.

The myPOS Virtual API provides different myPOS public certificate to everyone online store of the merchant. They are available for download at eCommerce / Online stores / Keys menu.

myPOS Virtual API requires from merchant to upload his public certificate so that his digital signature can be verified from the system. The merchant can upload several public certificates. A key index is assigned to each certificate. For each of the merchant's public certificate there is a certain myPOS public certificate. The merchant can download each myPOS public certificate by clicking on Download in the myPOS public certificate column.



The online store public certificate can be changed at any time from the eCommerce / Online stores / Keys menu.

## The functions of myPOS Virtual API

### General

This chapter describes the main functions of the applications implemented in myPOS Virtual API. The function calls are the core elements of the payment API. The IT specialists and e-commerce system developers will be provided with an initial introduction to the technical background of the API.



The complete API reference is shown in the [Part 2: Technical integration](#) of this document.

API function call	Description
<b>MERCHANT TO MYPOS VIRTUAL API</b>	

<a href="#">IPCPurchase</a>	This is the standard method for checkout at web shop. Customer interaction.
<a href="#">IPCRefund</a>	Credit to customer, e.g. return money. No customer interaction needed.
<a href="#">IPCGetTxnLog</a>	Returns detailed information about a previously executed payment. No customer interaction needed.
<a href="#">IPCGetTxnStatus</a>	Returns the status and the parameters of a previously executed payment. No customer interaction needed.
<b>MYPOS VIRTUAL API TO MERCHANT</b>	
<a href="#">IPCPurchaseNotify</a>	IPC will respond with this method on successful payment. The call will be made on previously supplied URL_Notify.
<a href="#">IPCPurchaseOK</a>	IPC will redirect with this method on successful payment. The call will be made on previously supplied URL_OK.
<a href="#">IPCPurchaseCancel</a>	IPC will redirect with this method when the customer chooses cancel payment. The call will be made on previously supplied URL_Cancel.
<a href="#">IPCPurchaseRollback</a>	IPC will notify that a reversal is passed for previous successful authorization. The merchant should mark the order as not paid (in case, the merchant has received IPCPurchaseNotify method). This is used when IPC do not receive and HTTP OK from the merchant as a response for IPCPurchaseNotify method.

## ***Individual transaction / purchase (API call: IPCPurchase)***

This is the standard method for checkout. If the customer has decided to pay via myPOS Virtual, the merchant's e-commerce system transmits an IPCPurchase call to myPOS system. The customer is redirected to the myPOS Virtual payment page.

myPOS Virtual will check for:

- Valid myPOS Account number
- Valid Online Store ID corresponding with this myPOS Account number
- Valid status of the Online store (enabled)
- Valid currency and total amount
- Valid signature

## ***Successful payment notification (API call: IPCPurchaseNotify / IPCPurchaseOK)***

This call is used by myPOS Virtual API to notify the merchant for a successful payment and to pass all needed parameters for the payment on URL\_Notify. After successful response to this call the API will redirect the customer browser to URL\_OK and will pass the same parameters as IPCPurchaseOK call.

## ***Cancellation of payment notification (API call: IPCPurchaseCancel)***

This call is used by myPOS Virtual API to notify the merchant that the customer has cancelled the payment. The API will redirect with this call when the customer chooses cancel payment. The call will be made on previously supplied URL\_Cancel.





### ***Notification for Rollback of previous executed payment (API call: IPCPurchaseRollback)***

This call is used by myPOS Virtual API to notify that a reversal is passed for previous successful authorization. The merchant should mark the order as not paid (in case, the merchant has received IPCPurchaseNotify call). This is used when myPOS Virtual API does not receive an HTTP OK from the merchant as a response from IPCPurchaseNotify call. The call will be posted to URL\_Notify.

### ***Get transaction log for previously executed payment (API call: IPCGetTxnLog)***

This call is used by the Merchant to get information about previously executed payment. The myPOS Virtual API will return an xml with detailed information about a specific OrderID. This call is intended to be utilized by the Merchant in his website back-end. The merchant could decide whether or not to use this method.

### ***Make a refund of a previously executed payment (API call: IPCRefund)***

This call is used by the Merchant to initiate a refund of previously executed payment. The myPOS Virtual API will return an xml with the result. This call is intended to be utilized by the merchant in his website back-end. The merchant could decide whether or not to use this method.

### ***Get transaction status for previously executed payment (API call: IPCGetTxnStatus)***

This call is used by the Merchant to get the current status of previously executed payment. The myPOS Virtual API will return an xml with detailed information about a specific OrderID. This call is intended to be utilized by the Merchant in his website back-end. The merchant could decide whether or not to use this method.

## Business myPOS Account details

### Integrating myPOS Virtual with online stores

From menu eCommerce / API / the Merchant can activate the service.

### Adding of an online store

Once the service is activated the Merchant can start adding the information of the online stores in menu eCommerce / Online Stores / Add new store:

**Add new store**Cancel ✕

**Store details**

Store name: \*

Activity: \*

Please, choose... ▾

Billing descriptor: \*

MYPOS \*

Maximum 22 symbols, including MYPOS \*. Use Latin letters only.

Website URL: \*

http://

Store logo: \* ⓘ

Browse

Maximum banner size: 940 x 105 px  
File formats: jpg, jpeg, png.  
Maximum file size: 64M

Request URLs: ⓘ

http://

Ⓜ Add

**Business contact details ⓘ**

Customer service URL: \*

http://

Customer service email: \*

Customer service phone: \*

**Transaction currencies ⓘ**

Currency \*      Settlement account \*

☐ EUR

☐ USD

☐ GBP

☐ HRK

☐ CHF

☐ RON

☐ JPY

☐ BGN

\* Required fields

Review and confirm

### Setting up the new online store

Once added, the new online store will be visible at the eCommerce / Online Stores / View all stores menu. The new online store will be with status “Disabled”. The merchant needs to:

1. activate the store once he is ready with the test integration with the myPOS Virtual API by clicking on the appropriate button beside the store;
2. read carefully and agree with the General terms and conditions and the Tariff for this store.
3. set key pairs for this particular website;
4. enable the store by clicking on the appropriate button beside the store.



Every online store could be enabled / disabled separately at any time by the merchant.

# Accepting e-commerce payments for merchants

myPOS Virtual API v.1.0.4



**myPOS** Helps my business grow.

08:13 EN Name Surname / [Logout](#)  
Current client: Client name / [Switch](#)

Accounts Devices Cards Transfers **eCommerce** Merchant services Profile

BUY Buttons and Links

**Online Stores**  
View all stores  
Add new store  
Transactions

API  
Shopping carts  
Integration Support

**My online stores**

Store ID	Billing descriptor	Store name	URL	Used Currencies	Tariff	Status	Actions
726	MYPOS *STORE1	Store name 1	www.website1.com	MYPOS *STORE1 *STORE1 Account MYPOS *STORE1 *STORE1 Account	<a href="#">View</a>	Enabled	<a href="#">Disable</a>   <a href="#">Keys</a>   <a href="#">Edit</a>
751	MYPOS *STORE2	Store name 2	www.website2.com	MYPOS *STORE2 *STORE2 Account MYPOS *STORE2 *STORE2 Account	<a href="#">View</a>	Disabled	<a href="#">Enable</a>   <a href="#">Keys</a>   <a href="#">Edit</a>

💡 If you need help setting up your integration, please visit [Integration Support](#).

[General terms and conditions](#) | [Tariff](#) | [Currency Exchange Rates](#) | [Return Policy](#) | [Privacy Policy](#) | [Acceptance Policy](#) | [Return Material Authorization](#)

[Contacts](#) | [sales@mypos.eu](#) | [support@mypos.eu](#)



Accounts Devices Cards Transfers **eCommerce** Merchant services Profile

BUY Buttons and Links

**Online Stores**  
View all stores  
Add new store  
Transactions

API  
Shopping carts  
Integration Support

**Keys for store "Store name 1"** [Back](#)

The myPOS™ Virtual API requires the use of key pairs' cryptography to ensure that the identity of the sender is guaranteed. It works through the use of public key and private certificate.

For signing process, both myPOS™ and the Merchant generate public and private key pairs and exchange the public certificate. Every of the parties are using the private key to sign the message and the opposite side authenticate the sender with corresponding public certificate. Key pairs are generated using RSA algorithm. The certificates must be PEM-encoded PKCS7 file.

**Generate and upload your Store name 1 myPOS™ Public Certificate**

The merchant can upload several public certificates. A key index is assigned to each certificate. The key index of the certificate used to sign the request should be supplied as a parameter in every transmission. For each of the merchant's public certificate there is a certain myPOS™ public certificate. The merchant can download each myPOS™ public certificate by clicking on Download in the myPOS™ public certificate column.

Keys						<a href="#">Add new certificate</a>
Key index	Added on	Custom name	Certificate name	Expiration date	Actions	myPOS™ public certificate
1	10.09.2015 15:08	Store name public key 1	/O=iPay	07.08.2025	<a href="#">Download</a>   <a href="#">Delete</a>	<a href="#">Download</a>

If you prefer, you could [Generate new key pair](#) for your online store here, save them and upload your public key above.

**If you use myPOS™ Virtual with a shopping cart you need to:**

1. Generate new key pair for your online store.
2. Save them to a text file or similar.
3. Copy your private key to the eCommerce platform » myPOS™ Virtual payment method settings.
4. Upload your public certificate above.
5. Copy the relevant myPOS™ public certificate to the eCommerce platform » myPOS™ Virtual payment method settings.

[Generate new keys](#)

💡 If you need help setting up your integration, please visit [Integration Support](#).

## Editing the online store data

The merchant can change the main information of every online store using the "Edit" functionality at the eCommerce / Online Stores.

## E-commerce transactions

### Overview

All e-commerce transactions will be visible in the merchant's myPOS Account under menu eCommerce / Online Stores / Transactions. The merchant could filter and see only the transactions which come from myPOS Virtual or from a specific online store.

### Details of e-commerce transactions

The merchant will see the following type of details for the e-commerce transactions:

- Payment details – such as payment origin and used payment method;
- Transaction details – transaction reference, exchange rate, amount, transaction fee;
- Reserve – reserve amount from the specific transaction and reimbursement date;
- Cart details – all data from the shopping cart shown to the customer during the checkout, i.e. item name, price, quantity, order total and note;
- Refund details / functionality – refund functionality with which the merchant could initiate partial or full amount refund transaction to a previously executed payment. If there is at least one refund transaction for the initial payment the merchant will see it here together with refund transaction reference and refund amount and fee.

### Making a refund

In case where the customer has lodged a justified complaint or the Merchant is unable to deliver the goods, he could refund any amount of the payment back to the customer. The merchant can refund up to 100 percent of the original amount.

When the merchant initiate a refund, a reference will be added to the original transaction details along with new row for the refund transaction.



Once the merchant issues a refund, it cannot be cancelled.

If there are insufficient funds in the merchant's myPOS Account, the refund transaction will be denied.



If the myPOS Account balance becomes negative as a result of deducted refunds or chargebacks, Satabank will collect funds from the merchant in accordance with the Legal agreements.

## Reserve

### Overview

The reserve is a percentage of money that must remain in the merchant's myPOS Account for a specific period of time to cover any payment reversals that may be received like chargebacks, claims, and disputes.

The reserve rate and period is specified in the merchant's tariff and depends on the risk assessment of the particular online store. They will be reviewed regularly and adjusted when necessary.



If the merchant Business myPOS Account is blocked by Issuer or closed for any reason, Satabank may hold the reserve for up to 180 days after the date of the last transaction with myPOS Virtual.

### Details

The merchant can track the accumulated currency reserve amount and how exactly they are calculated at any time. He can filter and see all transactions which were calculated in it and when the particular amount will be reimbursed.

## Frequently asked questions

Why choose myPOS Virtual API? What are the main benefits of the API?

The myPOS Virtual ensures secure transactions to be transmitted over the Internet.

- It is easy to be implemented in a website.
- Ability to support multiple payment methods and multi payment service providers (card schemes) with a single implementation.
- No need of additional organization, collection and storing of sensitive customer data on Security servers, respectively to meet all the requirements for these procedures. All this saves time and reduces the costs.
- Best way to provide convenience to your customers.

How can I integrate my e-commerce platform to accept payment with myPOS Virtual?

You could find all needed information for your integration in the myPOS Virtual section of the website. myPOS provides full Integration manual both with detailed working examples, usage cases, error codes and test environment.

Do I need to have a separate merchant account?

No, you don't. You need only to have a Business myPOS Account and to apply for e-commerce payment processing.

Do I need to have a separate Business myPOS Account in order to be able to accept online payments on different websites?

No, there are no restrictions about how many websites you will link to your Business myPOS Accounts.

When and where I will receive my money?

You will receive all your payments (from all your approved online stores) in your Business myPOS Account instantly.

Will I get the full amount of all my payments instantly?

Yes, you will get all payments into your Business myPOS Account instantly. However, a small amount will be unavailable for active operations for a specific period of time depending on your contract. This is your Reserve balance. These funds can be used to prevent transactions loss that may occur from payment reversals like chargebacks and disputes filed by the buyers.

Do I need to make a request for a pay-out from my myPOS Account or it is settled automatically?

Whenever you want to transfer money to your bank account, you need to use the "Transfers" functionalities in your myPOS Account. There will be no automatically set transfers from your myPOS Account.

What if I need to cancel the online payment service? Do I need to cancel the Business myPOS Account too?

You can terminate your use of myPOS Virtual at any time without any cancelation fees or long-term commitment. If you want to, you can keep using your Business myPOS Account.

How much does it cost to use myPOS Virtual API on my website?

The price could depend on the Risk Assessment of your company and the websites you want to be acquired. Please refer to our standard tariff or contact our Customer service department.

Will the myPOS Virtual API going to collect any additional fees from the customer for every transaction?

No. The myPOS Virtual API will not collect any fees from the customers.

Is it possible for me to cancel an order received from a customer? Are there any charges for such cancellation?

Yes, you could initiate a partial or full refund transaction for any previously processed transaction using the Refund functionality in your Business myPOS Account. Regarding the charges please refer to our tariff or to your contract.

How can I issue a Refund for my client?

Login to your Business myPOS Account, find the transaction you want to refund on the Transactions page and open the details. At the bottom of the details you will see a "Refund" button. If you click on it, you will see new window where you could specify the amount and reason of the refund and confirm the operation.

How long does it take for a transaction to take place?

Transactions submitted to the myPOS Virtual API are processed in real time and take approximately 1-3 seconds to return a response.

Are there any restrictions about the transactions per month which I can process?

No, there are no restrictions how many transactions you will process during a month. However, in some cases there could be some processing restrictions depending on the Risk assessment of your company and online stores.

Will I be able to see all transactions and turnover in my myPOS Account?

Yes, you will see all your transactions and account details (incl. statements) in your Business myPOS Account.

Is it possible to add new or edit the approved currencies in which I could accept payments on my online store?

Yes. You could change them at any time using your online store settings in the Business myPOS Account. However, they will be reviewed and verified too.

How do I manage my store settings?

You need to go to eCommerce / Online Stores menu and:

- set the keys for this store – to download the myPOS public certificate and to upload your public certificate.
- edit your store so to include all URLs from which you will send myPOS Virtual requests to the myPOS. Please have in mind that all changes of the online stores become active after an approval.

## Part 2: Technical integration



## Overview

### HTTP POST

Data transfer between Merchant and myPOS Virtual is made by HTTP POST. All the parameters for the requests are in the body in [parameter=value] form. Separator between tokens is [&]. The body is URL Encoded. Character encoding is UTF-8.

#### Example:

```
POST /somescript.php HTTP/1.1
Host: www.somesite.com
User-Agent: Mozilla/4.0
Content-Length: 39
Content-Type: application/x-www-form-urlencoded

userid=joe&password=guessme&user_type=1
```

### Data type formats

Data Type in document	Description	Example
int	Integer	1
String	String	This is a string
Date	ISO 8601 date string YYYY-MM-DD	2012-03-31
DateTime	ISO 8601 datetime string YYYY-MM-DD HH:mm:ss	2012-03-31 23:59:59
A(n)	Alpha string. [n] characters required	Alpha string
AN(n)	Alphanumeric string. [n] characters required	Alphanumeric string
N(n)	Numeric string. [n] characters required. Number is left-padded with zeroes.	000123
double	Numeric string with decimal point. Only point is used (no commas or other characters for decimal point)	34.56
BASE64	String used to pass binary data. The binary data should be converted to base64 standard.	YW55IGNhcm5hbCBwbGVhc3VyZQ==
XML	Simple in place XML array.	<pre>&lt;body&gt;   &lt;param&gt;1&lt;/param&gt;   &lt;value&gt;2&lt;/value&gt; &lt;/body&gt;</pre>

### Signature and authentications

Signatures are calculated using the following mechanism. All data in POST request without the Signature property are concatenated with dash and Base64 encoded. The string is signed with the private key using the SHA-256 algorithm. Then the signature needs to be Base64 encoded. The signature property is added to the POST request.

The opposite side should concatenate all data in the POST request without the Signature property, Base64-encode the string and then verify the obtained string with the sent signature property and the public key extracted from the myPOS public certificate.



The Merchant should always verify the signature when receiving a call from myPOS Virtual API!

## Example for PHP 5.x.x

```
<?php
```

```
// The POST data array
```

```
$postData = array('IPCmethod'=>'IPCPurchase', .....);
```

```
// This is an example of RSA private key
```

```
$privKey = '-----BEGIN RSA PRIVATE KEY-----
```

```
MIICXAIBAAKBgQCf0TdcTuphb7X+Zwekt1XKEWZDczSGecfo6vQfqvraf5VPzcnJ
2Mc5J72HBm0u98EJHan+nle2WOZMVGItTa/2k1FRWwbt7iQ5dzDh5PEeZASg2UWe
hoR8L8MpNBqH6h7ZITwVtFRS4LsBvIEfT7Pzhm5YJKfM+CdzDM+L9WVEGwIDAQAB
AoGAYfKxwUtEbq8uIVrD3nnWhF+hk1k6KejdUq0dLYN29w8WjbCMKb9laokmqWiQ
5iZGERYxh7G4BDP8AW/+M9HXM4oqm5SEkaxhbTlgks+E1s9dTpdFQvL76TvodqSy
l2E2BghVgLLgkdhRn9buaFzYta95JKfgyKGonNxsQA39PwECQQDKbG0Kp6KEkNgB
srCq3Cx2od5OfiPDG8g3RYZKx/O9dMy5CM160DwusVJpuywbpRhcWr3gkz0QgRMd
IRVwyxNbAkEAyh3sipmcgN7SD8xBG/MtBYpQWP1vxhSVYPfJzuPU3gS5MRJzQHBz
sVCLhTBY7hHSoqiqlqWYasi81JzBEwEuQQJBAKw9qGcZjyMH8JU5TDSGllr3jybx
FFMPj8TgJs346AB8ozqLL/ThvWPpxHttJbH8QAdNuyWdg6dIfVAa95h7Y+MCQEZg
jRDI1Bz7eWGO2c0Fq9OTz3IVLWpnmGwfw+HyaxizxFhV+FOj1GUVir9hylV7V0DU
Qjlajyv/oeDWhFQ9wQECQCydhJ6NaNQOCZh+6QTrH3TC5MeBA1Yeipoe7+BhsLNR
cFG8s9sTxRnltcZl1dXaBSemvpNvBizn0Kzi8G3ZAgc=
-----END RSA PRIVATE KEY-----';
```

```
// You need to concatenate all values from $postData and to Base64-encode the result
```

```
$concData = base64_encode(implode('-', $postData));
```

```
$privKeyObj = openssl_get_privatekey($privKey);
```

```
// Signed data in binary
```

```
openssl_sign($concData, $signature, $privKeyObj, OPENSSL_ALGO_SHA256);
```

```
// Base64 encoding of the signature
```

```
$signature = base64_encode($signature);
```

```
// Now you need to add the signature to the POST request
```

```
$postData['Signature'] = $signature;
```

```
openssl_free_key($privKeyObj);
```

```
?>
```

## Signature verification example for PHP 5.x.x

```
<?php
```

```
// Save POST request data in var $data
```

```
$data = $_POST;

// myPOS certificate
$cert = '-----BEGIN CERTIFICATE-----
MIIBkDCB+qADAgECAGAwDQYJKoZIhvcNAQEFBQAwDzENMAAGA1UEChMEaVBheTAe
Fw0xMzAzMTMxMTI1MTFaFw0yMzAzMTExMTI1MTFaMA8xDTALBgNVBAoTBGQYXkw
gZ8wDQYJKoZIhvcNAQEBBQADgY0AMIGJAoGBAML+VTmiY4yChoOTMZXAIg/mk+x
f/9mjwHxWzxtBJbNncNK0OLIOVXYKW2GgVklGHHQjvew1hTFkEGjnCJ7f5CDnbgx
evtyASDGst92a6xcAedEadP0nFXhUz+cYYIglcgfDcX3ZWeNEF5kscqy52kpD2O7
nFNCV+85vS4duJBNAgMBAAEwDQYJKoZIhvcNAQEFBQADgYEAfSfqJHH9Vp9Y4osJ
sLg1Um5LOoTgn6u4JepHMFoSiwYE0n/N3D3JlgqAzjdVJ+1rZV95Vaf/+TKzWzvP
V8L01LJ8aRFkUaPGenVsGvBT2mtsbu34QUOIPgzCi3huidwk0yIMX7zo8uxu1cXv
/bg5jBGe5SjvJP8Tq257QcAGgkA=
-----END CERTIFICATE-----';

// Save signature
$signature = $data['Signature'];
// Remove signature from POST data array
unset($data['Signature']);
// Concatenate all values
$concData = base64_encode(implode('-', $data));

// Extract public key from certificate
$pubKeyId = openssl_get_publickey($cert);

// Verify signature
$res = openssl_verify($concData, base64_decode($signature), $pubKeyId, OPENSSL_ALGO_SHA256);
//Free key resource
openssl_free_key ($privKeyObj);

if($res==1){
    //success
}else{
    //not success
}

?>
```

## Test environment

### Overview

The myPOS Virtual API test environment is a self-contained, virtual testing environment. Once you are ready with the API integration, you need to use it in order to make the appropriate tests of all your myPOS Virtual API calls and communication with the system. The test environment will let you test the API integration without affecting any real myPOS users or their real myPOS Wallets.

The myPOS Virtual API test environment mimics myPOS's production servers, so you can format and make all your Calls to the system as you will do in Production environment. The only difference between the Calls on both environments is that you need to use different user credentials and endpoints.

Before you start using myPOS Virtual into production, you should test the application with all case scenarios in myPOS Virtual API test Environment.

## Testing data

Test environment host: <https://www.mypos.eu/vmp/checkout-test>

SID: 0000000000000010

Wallet number: 61938166610

Customer Email: demo@ipay.eu

## POST parameters example

```
IPCmethod=IPCPurchase&
IPCVersion=1.0&
IPLanguage=EN&
WalletNumber= 61938166610&
SID=0000000000000010&
Amount=6.68&
Currency=EUR&
OrderID=1854&
URL_OK= http://site.ext/paymentOK&
URL_Cancel=http://site.ext/paymentNOK&
URL_Notify=https://site.ext/paymentNotify&
KeyIndex=1&
CustomerEmail=demo@ipay.eu&
CustomerPhone=+23568956958&
CustomerFirstNames= John Santamaria&
CustomerFamilyName=Smith&
CustomerCountry=DEU&
CustomerCity=Hamburg&
CustomerZIPCode=20095&
CustomerAddress= Kleine Bahnstr. 41&
Note=note&
CartItems=2&
Article_1=HP ProBook 6360b sticker&
Quantity_1=2&
Price_1=2.34&
Amount_1=4.68&
Currency_1=EUR&
Article_2=HP ProBook 6360b sticker&
Quantity_2=1&
Price_2=2.00&
Amount_2=2.00&
Currency_2=EUR
Signature=Y0Gy2zybnHq/EPVwhTsl+D3oJ1Oj1jKF3SK8li/XFAJ2JEuxwoqjmWI3aegLsP+6duM/6o10I6tBfAt67PjEqxuNzqlpcVAs
Yd9f0XjtaCUe46bjP995HHAWiISUTmnCtIFIDnKegwciiNhb/jKFH4Fc1LNlzxT7Ls84vSKfPYM=
```



Please, have in mind that you could test with every debit or credit card number. In test environment, all card transactions will be processed as successful payment. Your card will not be charged!

## *Test scenarios*

In order to start testing the myPOS Virtual API payments, you need to:

1. Implement the myPOS Virtual API.
2. Send a POST request with all the data to the test host address.
3. Test the following scenarios:
  - Error in the POST request – missing or invalid data;
  - Send wrong signature to myPOS Virtual API;
  - Check myPOS Virtual API message signature for authenticity;
  - The Customer terminates the process on the myPOS Virtual payment page;
  - Customer pays successfully;
  - Customer pays successfully, however the merchant's website does not return HTTP OK to myPOS Virtual API.

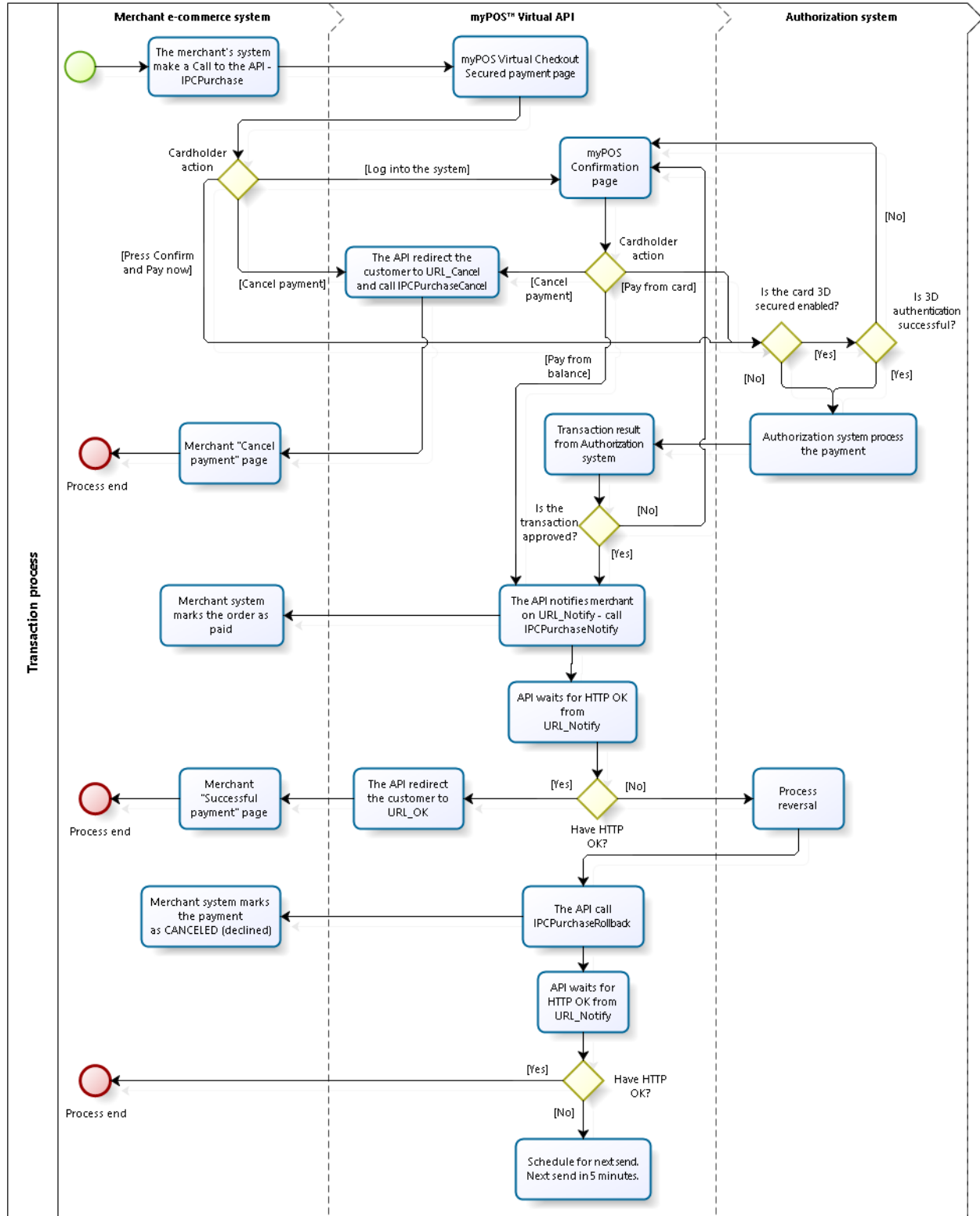
## *Production environment*

Once you have everything implemented and tested, you need to set your online store keys to production environment and change your test data.

Production environment host: <https://www.mypos.eu/vmp/checkout>

## The process

### Understanding transmission mechanism



## Method standard properties

In every request, there are several parameters that are always supplied. Below they are called 'standard properties'. Once defined below, they won't be described in every single command listed in the specification, they should be considered as existing to every command.

Property	Typical value	Type	Description
IPCmethod	IPCPurchase	String	Name of the method requested for execution from IPC.
Signature	Byte[]	BASE64	SHA-256 HASH for all properties in the command. Signature is <b>ALWAYS THE LAST PARAMETER IN THE POST</b> , as it is not used to calculate the hash.
KeyIndex	1	Int	Identifier of the private key used for signature (if more than 1).
IPCVersion	1.0	string	Version of protocol used for transition.
IPCLanguage	EN	A(2)	ISO 2-character code for the desired language on the payment page. If IPC cannot fulfill the requested language, it will set the English language as defaults. Currently supporting EN, FR, DE, BG, ES, RO.
WalletNumber	11111111111	N(11)	myPOS Account Number
SID	000000000000010	AN(15)	Site ID. Identifier of the website accepting payment.

## Response standard properties

Upon HTTP request, the party should respond with header HTTP 200 OK with the following body content: "OK". Every other response should be treated as communication error, call error, server error or system malfunctions.

Property	Typical value	Type	Description
IPCmethod	IPCPurchase	String	Name of the method requested for execution from IPC.
Status	0	N(3)	Code upon command completion. 0 is success otherwise error.
StatusMsg	Success	C(max)	Description for <status> code.
Signature	Byte[]	BASE64	SHA-256 HASH for all properties in the command. Signature is <b>ALWAYS THE LAST PARAMETER IN THE POST</b> , as it is not used to calculate the hash.

Once defined these parameters should be considered as existing to every response listed in the specification.

## The calls

API function call	Description
<b>MERCHANT TO MYPOS VIRTUAL</b>	
<a href="#">IPCPurchase</a>	This is the standard method for checkout at web shop. Customer interaction.
<a href="#">IPCRefund</a>	Credit to customer, e.g. return money. No customer interaction needed.

<a href="#">IPCGetTxnLog</a>	Returns detailed information about a previously executed payment. No customer interaction needed.
<a href="#">IPCGetTxnStatus</a>	Returns the status and the parameters of a previously executed payment. No customer interaction needed.
<b>MYPOS VIRTUAL TO MERCHANT</b>	
<a href="#">IPCPurchaseNotify</a>	myPOS Virtual will respond with this method on successful payment. The call will be made on the previously supplied URL_Notify.
<a href="#">IPCPurchaseOK</a>	myPOS Virtual will redirect with this method on successful payment. The call will be made on the previously supplied URL_OK.
<a href="#">IPCPurchaseCancel</a>	myPOS Virtual will redirect with this method when the customer chooses to cancel the payment. The call will be made on the previously supplied URL_Cancel.
<a href="#">IPCPurchaseRollback</a>	myPOS Virtual will notify that a reversal is passed for a previous successful authorization. The merchant should mark the order as not paid (in case, the merchant has received IPCPurchaseNotify method). This is used when IPC does not receive an HTTP OK from the merchant as a response for the IPCPurchaseNotify method.

All commands described below do not include the standard properties discussed in the previous topic. However, the standard properties are mandatory for all commands.

## Purchase with payment card (API call: IPCPurchase)

### Purpose

This method initiates the beginning of the payment process for a customer. The customer is placed on a page that requests entering payment card details.

myPOS Virtual will check for:

- Valid myPOS Account number
- Valid Online Store ID corresponding with this myPOS Account number
- Valid status of the Online store (enabled)
- Valid currency and total amount
- Valid signature

### Method properties

Property	Typical value	Type	Required	Description
Amount	23.45	Double	YES	The amount of the payment requested.
Currency	EUR	A(3)	YES	ISO 3-character currency code. The currency for the payment should be registered and approved.



OrderID	20120331999999	String	YES	Placeholder for the merchant. Used to put some data that will help the merchant to recognize for which order is the payment. Up to 255 characters.
URL_OK	http://site.ext/paymentOK	String	YES	The page where the cardholder should be redirected on successful payment.
URL_Cancel	http://site.ext/paymentNOK	String	YES	The page where the cardholder should be redirected when <Cancel> is pressed on the payment page.
URL_Notify*	https://site.ext/paymentNotify	String	YES	Address supplied by the partner, where the IPCPurchaseNotify API call will send the parameters for the successful payment.
CustomerEmail	<a href="mailto:name@website.com">name@website.com</a>	String	YES	This is a customer's email.
CustomerPhone	+23568956958	String	NO	This is a customer's phone.
CustomerFirstNames	John Santamaria	String	YES	All customer's names without the surname.
CustomerFamilyName	Smith	String	YES	The customer's surname.
CustomerCountry	DEU	String	NO	ISO country code
CustomerCity	Hamburg	String	NO	
CustomerZIPCode	20095	String	NO	
CustomerAddress	Kleine Bahnstr. 41	String	NO	Customer's address.
Note		String	NO	Text associated with the purchase.
CartItems	2	Int	YES	The number of rows (items) in the logical record Cart. If there will be some additional fees/taxes for the cardholder, they need to be added as new items.
<u>Cart</u>	Logical Holder	Logical Record	YES	Array provided by the Merchant. The array describes the content of the shopping cart. The content will be displayed on the IPC payment page.

\* **The URL\_Notify URL should be SSL-enabled address only (i.e. it must start with "https://"). Unsecure URLs will be treated as wrong.**

Same OrderID is used for the request of a partner, except in case where the order has been marked as paid. Then OrderID is a unique identifier and IPC will reject duplicated transmission.

## Cart Logical Record

Cart logical record consists of standard POST parameters with the form name=value. For every consequent item an index is added that shows the logical record number for the item (ex. Atricle\_1). Indexes are from 1 to <CartItems>.

Property	Typical value	Type	Description
Article	HP ProBook 6360b sticker	String	Name of an article in the shopping cart.
Quantity	2	Int	How many pieces of an article.
Price	2.34	Double	Price of a single unit.
Amount	4.68	Double	Quantity*Price for the article.
Currency	EUR	A(3)	Should be the same currency as in the purchase amount.

## Example

New lines and tabulators are included for better reading and do not exist in the POST request.

```
IPCmethod=IPCPurchase&
IPCVersion=1.0&
IPLanguage=EN&
WalletNumber=61938166610&
SID=0000000000000010&
Amount=6.68&
Currency=EUR&
OrderID=20120331999999&
URL_OK= http://site.ext/paymentOK&
URL_Cancel=http://site.ext/paymentNOK&
URL_Notify=https://site.ext/paymentNotify&
KeyIndex=1&
CustomerEmail=name@website.com&
CustomerPhone=+23568956958&
CustomerFirstNames= John Santamaria&
CustomerFamilyName=Smith&
CustomerCountry= DEU&
CustomerCity=Hamburg&
CustomerZIPCode=20095&
CustomerAddress= Kleine Bahnstr. 41&
Note=note&
CartItems=2&
Article_1=HP ProBook 6360b sticker&
Quantity_1=2&
Price_1=2.34&
Amount_1=4.68&
Currency_1=EUR&
Article_2=Delivery&
Quantity_2=1&
Price_2=2.00&
Amount_2=2.00&
Currency_2=EUR
Signature=CXTVYvva71NOHNCOxtneR6IQ0FyalFPvNF3B6sgTjaYhyYyCUmkqxfAdN/Fi+KWzxdSCWs9idL5VpW99ao
CU2R5+qr0Pf6XfsSVsP4HrWnSYbT9KAkxDACGDZ6/HyQcqQjA3ufGCqU0nAApZjxnquoNNUp6/R0Ty/fP0/nOaUU=
```

## Successful payment notification (API call: IPCPurchaseNotify / IPCPurchaseOK)

### Purpose

This method is used by myPOS Virtual to notify the merchant for a successful payment and to pass all needed parameters for the payment on URL\_Notify. After successful response for this method myPOS Virtual will redirect the customer browser to URL\_OK and will pass same parameters with IPCPurchaseOK method.

### Method properties

Property	Typical value	Type	Description
Amount	23.45	Double	Echo from IPCPurchase.
Currency	EUR	A(3)	Echo from IPCPurchase.
OrderID	201203319999999	string	Echo from IPCPurchase.
IPC_Trnref	12345678923	String	Used to uniquely identify a transaction in IPC. Used as a parameter for subsequent refund of reversal if needed.
RequestDateTime	2012-03-31 23:59:59	DateTime	Date/time of the request
RequestSTAN	123456	N(6)	Consequent number from 1 to 999999. Used for request unique match.
Signature	Byte[]	BASE64	SHA-256 HASH for all properties in the command. Signature is <b>ALWAYS THE LAST PARAMETER IN THE POST</b> , as it is not used to calculate the hash.

The IPCGetTxnStatus call will return the parameters of the response.

### Example

```
HTTP/1.1 200 OK
Date: Mon, 11 Mar 2013 08:04:50 GMT
Server: Apache/2.2.22 (Win32) PHP/5.2.17
X-Powered-By: PHP/5.2.17
Set-Cookie: PHPSESSID=a048e995f78f9e38b192f528f67d3cc3; path=/
Expires: Thu, 19 Nov 1981 08:52:00 GMT
Cache-Control: no-store, no-cache, must-revalidate, post-check=0, pre-check=0
Pragma: no-cache
Content-Length: 2
Connection: close
Content-Type: text/html; charset=utf-8

OK
```

## Cancelation of payment notification (API call: IPCPurchaseCancel)

### Purpose

This method is used by myPOS Virtual to notify the merchant that the customer has cancelled the payment. myPOS Virtual will redirect with this method when the customer chooses cancel payment. The call will be made on the previously supplied URL\_Cancel.

### Method properties

Property	Typical value	Type	Description
Amount	23.45	Double	Echo from IPCPurchase.
Currency	EUR	A(3)	Echo from IPCPurchase.
OrderID	201203319999999	string	Echo from IPCPurchase.
Signature	Byte[]	BASE64	SHA-256 HASH for all properties in the command. Signature is <b>ALWAYS THE LAST PARAMETER IN THE POST</b> , as it is not used to calculate the hash.

## Rollback of previous notification (API call: IPCPurchaseRollback)

### Purpose

This method is used by myPOS Virtual to notify that a reversal is passed for previous successful authorization. The merchant should mark the order as not paid (in case the merchant has received IPCPurchaseNotify method). This is used when myPOS Virtual does not receive and HTTP OK from the merchant as a response for IPCPurchaseNotify method. The call will be posted to URL\_Notify.

### Method properties

Property	Typical value	Type	Description
Amount	23.45	Double	Echo from IPCPurchase.
Currency	EUR	A(3)	Echo from IPCPurchase.
OrderID	201203319999999	string	Echo from IPCPurchase.
Signature	Byte[]	BASE64	SHA-256 HASH for all properties in the command. Signature is <b>ALWAYS THE LAST PARAMETER IN THE POST</b> , as it is not used to calculate the hash.

## Get transaction log for previously executed payment (API call: IPCGetTxnLog)

### Purpose

This method is used by the Merchant to get information about a previously executed payment. The myPOS Virtual API will return an xml with detailed information about a specific OrderID. This method is intended to be utilized by the Merchant in his website back-end. The Merchant could decide whether or not to use this method.

## Method properties

Property	Typical value	Type	Required	Description
OrderID	201203319999999	String	YES	Placeholder for the merchant. Used to put some data that will help the merchant to refer payments to orders. Up to 255 characters.
OutputFormat	xml	String	NO	Output format of data. The property can be "xml" or "json". If it is not specified in the request, the default value is "xml".

## Response properties

Property	Typical value	Type	Required	Description
OrderID	201203319999999	String	YES	Placeholder for the merchant. Used to put some data that will help the merchant to refer payments to orders. Up to 255 characters.
log				Returns detailed information about a specific OrderID.

## Example of the xml response

```
<ipc_response>
  <IPCmethod>IPCGetTxnLog</IPCmethod>
  <OrderID>201203319999999</OrderID>
  <log>
    <item>
      <time>11.03.2013 09:42:56</time>
      <action>Received new POST request</action>
      <result>
        <IPCmethod>IPCPurchase</IPCmethod>
        <IPCVersion>1.0</IPCVersion>
        <IPLanguage>en</IPLanguage>
        <SID>000000000000010</SID>
        <WalletNumber>61938166610</WalletNumber>
        <Amount>23.45</Amount>
        <Currency>EUR</Currency>
        <OrderID>201203319999999</OrderID>
        <URL_OK>http://site.ext/paymentOK</URL_OK>
        <URL_Cancel>http://site.ext/paymentNOK</URL_Cancel>
        <URL_Notify>https://site.ext/paymentNotify</URL_Notify>
        <KeyIndex>1</KeyIndex>
        <CustomerEmail>name@website.com</CustomerEmail>
        <CustomerPhone>+23568956958</CustomerPhone>
        <CustomerFirstNames>John Santamaria</CustomerFirstNames>
        <CustomerFamilyName>Smith</CustomerFamilyName>
        <CustomerCountry>DEU</CustomerCountry>
      </result>
    </item>
  </log>
</ipc_response>
```

```

        <CustomerCity>Hamburg</CustomerCity>
        <CustomerZIPCode>20095</CustomerZIPCode>
        <CustomerAddress>Kleine Bahnstr. 41</CustomerAddress>
        <Note>Some note here</Note>
        <CartItems>2</CartItems>
        <Article_1>HP ProBook 6360b sticker</Article_1>
        <Quantity_1>1</Quantity_1>
        <Price_1>20.00</Price_1>
        <Currency_1>EUR</Currency_1>
        <Amount_1>20.00</Amount_1>
        <Article_2>Delivery</Article_2>
        <Quantity_2>1</Quantity_2>
        <Price_2>3.45</Price_2>
        <Currency_2>EUR</Currency_2>
        <Amount_2>3.45</Amount_2>
        <Signature>WwSv6Hk3cH9VAnIfGxOkIY/Ph0psiWGywNuZl8dAYPQZa8qFrLDh5Ck1RDAE
        7h1oSH0CZw5fVW7jAlhSGcjaOnk+wybGVbIMxP59aUEKEG8wW0N2CH6oft92W/APQ60
        rZ6Qsdlge3uiAKchfvJEcy3buuWVb6C2D58isBvrMFik=</Signature>
    </result>
</item>
<item>
    <time>11.03.2013 09:42:56</time>
    <action>Checking method</action>
    <result>OK</result>
</item>
<item>
    <time>11.03.2013 09:42:56</time>
    <action>Checking signature</action>
    <result>OK</result>
</item>
<item>
    <time>11.03.2013 09:42:56</time>
    <action>Check if SID match to wallet number</action>
    <result>OK</result>
</item>
<item>
    <time>11.03.2013 09:42:56</time>
    <action>Checking is purchase paid on Start page</action>
    <result>OK</result>
</item>
<item>
    <time>11.03.2013 09:42:56</time>
    <action>Checking is amount > 0</action>
    <result>OK</result>
</item>
</item>
<item>

```

```
<time>11.03.2013 09:42:58</time>
<action>Checking for request marked as paid with this order_id</action>
<result>OK</result>
</item>
<item>
  <time>11.03.2013 09:43:02</time>
  <action>Checking is cart items > 0</action>
  <result>OK</result>
</item>
<item>
  <time>11.03.2013 09:43:06</time>
  <action>Checking the URLs (url_ok, url_cancel, url_notify)</action>
  <result>OK</result>
</item>
<item>
  <time>11.03.2013 09:43:26</time>
  <action>Checking cart</action>
  <result>OK</result>
</item>
<item>
  <time>11.03.2013 09:43:26</time>
  <action>Check post params</action>
  <result>OK</result>
</item>
<item>
  <time>11.03.2013 09:43:46</time>
  <action>Checking method</action>
  <result>OK</result>
</item>
<item>
  <time>11.03.2013 09:43:46</time>
  <action>Checking signature</action>
  <result>OK</result>
</item>
<item>
  <time>11.03.2013 09:43:46</time>
  <action>Check if SID match to wallet number</action>
  <result>OK</result>
</item>
<item>
  <time>11.03.2013 09:43:50</time>
  <action>Display page:</action>
  <result>Login or join</result>
</item>
<item>
  <time>11.03.2013 09:43:56</time>
  <action>Checking method</action>
```

```
<result>OK</result>
</item>
<item>
  <time>11.03.2013 09:43:56</time>
  <action>Checking signature</action>
  <result>OK</result>
</item>
<item>
  <time>11.03.2013 09:43:56</time>
  <action>Check if SID match to wallet number</action>
  <result>OK</result>
</item>
<item>
  <time>11.03.2013 09:43:56</time>
  <action>Validate form data</action>
  <result>Required params: OK</result>
</item>
<item>
  <time>11.03.2013 09:43:56</time>
  <action> Validate form data </action>
  <result>PAN: OK</result>
</item>
<item>
  <time>11.03.2013 09:43:56</time>
  <action> Validate form data </action>
  <result>Expire year: OK</result>
</item>
<item>
  <time>11.03.2013 09:43:56</time>
  <action> Validate form data </action>
  <result>Expire date: OK</result>
</item>
<item>
  <time>11.03.2013 09:43:56</time>
  <action> Validate form data </action>
  <result>CVC: OK</result>
</item>
<item>
  <time>11.03.2013 09:43:56</time>
  <action> Validate form data </action>
  <result>Phone code match with country: OK</result>
</item>
<item>
  <time>11.03.2013 09:43:56</time>
  <action> Validate form data </action>
  <result>Validate phone: OK</result>
</item>
```



```
<item>
  <time>11.03.2013 09:43:56</time>
  <action> Validate form data </action>
  <result>Validate email: OK</result>
</item>
<item>
  <time>11.03.2013 09:44:00</time>
  <action>Payment status</action>
  <result>0</result>
</item>
<item>
  <time>11.03.2013 09:44:00</time>
  <action>Sending POST request to Merchant with parameters:</action>
  <result>
    <IPCmethod>IPCPurchaseNotify</IPCmethod>
    <SID>000000000000010</SID>
    <Amount>23.45</Amount>
    <Currency>EUR</Currency>
    <OrderID>201203319999999</OrderID>
    <IPC_Trnref>370</IPC_Trnref>
    <RequestSTAN>000102</RequestSTAN>
    <RequestDateTime>2013-03-11 09:44:01</RequestDateTime>
    <Signature>mE0wMClbyhtkwS3NNccO8iJK+RT2tsDnj9XoIA6xA3bh+bPqglC4WBRD
    +IZFDRkIDFCh6tY4FvW8srbnmvlfwSOMmNqofe18mDnrDxAjy6Mj6yxCJReQslOK
    IHTryAzAnEk7NbBBw/WcypTGHX9uBg3lsM/Pwn5E+lpXMiHxA=</Signature>
  </result>
</item>
<item>
  <time>11.03.2013 09:44:00</time>
  <action>Connecting with URL_NOTIFY</action>
  <result>Connection with http://site.ext/paymentNotify is OK</result>
</item>
<item>
  <time>11.03.2013 09:44:00</time>
  <action>Merchant response</action>
  <result>OK</result>
</item>
<item>
  <time>11.03.2013 09:44:00</time>
  <action>View "Thank You - Success" page</action>
  <result>OK</result>
</item>
<item>
  <time>11.03.2013 09:44:00</time>
  <action>Order done with status</action>
  <result>1</result>
</item>
```

```
<item>
  <time>11.03.2013 09:44:02</time>
  <action>Display page:</action>
  <result>Thank you page (Success)</result>
</item>
</log>
<Status>0</Status>
<StatusMsg>Success</StatusMsg>
</ipc_response>
```

Detailed information about possible responses in the log is shown in the following table:

Response	Description
IPCmethod	Name of the method requested for execution from myPOS Virtual.
Checking is purchase paid on Start page	
Checking Merchant for 3DS for card scheme	
Merchant is not 3DS	
Merchant is 3DS	
Client is redirected to "Response 3DS page".	
Display page	
Checking is amount > 0	Checking for a valid amount.
Checking valid currency and if the post currency is the currency of MID	
Checking for request marked as paid with this order_id	Checking for an existing already paid request with the same order_id.
Checking is cart_items > 0	
Checking the URLs (url_ok, url_cancel, url_notify)	
Checking cart	
Received new POST request	
Checking version	Checking for version number of myPOS Virtual.
Checking method	Checking for valid method.
Checking signature	Checking if parameter 'Signature' is correct.
Payment status	Returns information about the status of the payment.
View "Thank You - Success" page	
View "Thank You - Error" page	
Adding request in TODO table (cron)	Used in case where the party's response is different from 'OK' or reversal is not passed.
Checking reversal	
Checking clearing	
Validate form data	Checking for required fields, PAN, Expire date, CVC.
Sending POST request to Merchant with parameters	Returns information about the request.
Connecting with URL_NOTIFY	
Merchant response	
Start checking card for 3DS	

Starting payment (Without 3DS)	
Redirecting to ACS	Redirecting to Access control server in order to check for 3DSecure.
Error while checking card for 3DS	
Redirecting to Home Page	
Redirection to Merchant Cancel URL	
Redirection to Merchant OK URL	
Starting payment (With 3DS)	
Card successfully validated	
Card check for 3DS	
Invalid card issuer	
Check referrer URL	
Check post params	
Check 3DS response	
User is blocked by issuer	
Trying to send reversal notify (cron)	
Update cron table (cron)	
Order done with status	Returns the order's status (0 – pending, 1 – paid, 2 – reversed, 3 – cancelled).
Check if SID match to Wallet number	
Display output data	
OTHER	Other additional data.

## Make a refund for previously executed payment (API call: IPCRefund)

### Purpose

This method is used by the Merchant to initiate a refund of a previously executed payment. The myPOS Virtual API will return an xml with the result. This method is intended to be utilized by the Merchant in his website back-end. The Merchant could decide whether or not to use this method.

### Method properties

Property	Typical value	Type	Required	Description
IPC_Trnref	12345678923	String	YES	Used to uniquely identify a transaction in IPC. Used as a parameter for subsequent refund or reversal if needed.
OrderID	201203319999999	String	NO	Placeholder for the merchant. Used to put some data that will help the merchant to recognize for which order is the payment. Up to 255 characters.
Amount	23.45	Double	YES	The amount of the payment requested.
Currency	EUR	A(3)	YES	ISO 3-character currency code. The currency for the payment should be registered and approved.

OutputFormat	xml	String	NO	Output format of data. The property can be “xml” or “json”. If it is not specified in the request, the default value is “xml”.
--------------	-----	--------	----	--

## Response properties

Property	Typical value	Type	Required	Description
IPC_Trnref	12345678923	String	YES	Used to uniquely identify a transaction in IPC. Used as a parameter for subsequent refund or reversal if needed.
Amount	23.45	Double	YES	The amount of the payment requested.
Currency	EUR	A(3)	YES	ISO 3-character currency code. The currency for the payment should be registered and approved.

## Example of the xml response

```
<IPC_response>
  <IPCmethod>IPCRefund</IPCmethod>
  <IPC_Trnref>12345678923</IPC_Trnref>
  <Amount>23.45</Amount>
  <Currency>EUR</Currency>
  <Status>0</Status>
  <StatusMsg>Success</StatusMsg>
</IPC_response>
```

## Get transaction status for previously executed payment (API call: IPCGetTxnStatus)

### Purpose

This method is used by Merchant to get the current status of a previously executed payment. The myPOS Virtual API will return an xml with detailed information about a specific OrderID. This method is intended to be utilized by the Merchant in his website back-end. The Merchant could decide whether or not to use this method.

### Method properties

Property	Typical value	Type	Required	Description
OrderID	201203319999999	String	YES	Placeholder for the merchant. Used to put some data that will help the merchant to refer payments to orders. Up to 255 characters.
OutputFormat	xml	String	NO	Output format of data. The property can be “xml” or “json”. If it is not specified in the request, the default value is “xml”.

### Response properties



Please refer to Response properties of every command in order to see the information which will be returned by myPOS Virtual API.

## Appendix I – Error messages

Code	Error	Description
0	Success	No errors
1	E_MISSING_REQ_PARAMS	Some of the required fields from the POST request are missing.
2	E_SIGNATURE_FAILED	The parameter 'Signature' is not correct.
3	E_IPAY_ERROR	Invalid or missing response from myPOS Virtual servers. Please contact myPOS engineers.
4	E_INVALID_SID	The parameter 'SID' is not correct.
5	E_INVALID_PARAMS	One or more of the other parameters from the POST request are not correct.
6	E_INVALID_REFERER	The POST request is received from an URL which is not approved.
8	E_TRANSACTION_AUTH_FAIL	The authorization of the transaction has been failed.
9	E_WRONG_AMOUNT	The parameter 'Amount' is not correct.
10	E_UNSUPPORTED_CALL	
11	E_INACTIVE_MANDATE_REFERENCE	The error will be returned both if the registration of the mandate reference is still not authorized, rejected or temporarily blocked.
12	E_INVALID_MANDATE_REFERENCE	
13	E_NOT_SUFFICIENT_FUNDS	
14	E_TRANSACTION_NOT_PERMITTED	
15	E_EXCEEDED_LIMIT	
16	E_MANDATE_ALREADY_REGISTERED	
17	E_INACTIVE_ACCOUNTIDENTIFIER	The error will be returned when the client's payment instrument is still not activated.
18	E_INVALID_ACCOUNTIDENTIFIER	
19	E_EXCEEDED_ACCOUNT_LIMITS	The error will be returned when the transaction cannot be processed because of reached Wallet Account limits.
99	E_UNDEFINED_ERROR	Other unspecified error.

### Note:

If the system returns an error on IPCPurchase call, the payment page will not be displayed. The cardholder (client) will see an error page with the following general message:

*[The online store name] has sent a shopping cart with errors in it.*

*We will contact the Merchant with a request to fix this problem. As this could be a temporary issue, you can go back to try checking out again.*

*<< Return to [Online store name]*

## Appendix II – Testing data

Field name	Test value
myPOS Virtual payment link	<a href="https://www.mypos.eu/vmp/checkout-test">https://www.mypos.eu/vmp/checkout-test</a>
SID	000000000000010
myPOS Account Number	61938166610

### Test private key

-----BEGIN RSA PRIVATE KEY-----

```
MIICXAIBAAKBgQCf0TdcTuph7X+Zwekt1XKEWZDczSGecfo6vQfqvraf5VPzcnJ
2Mc5J72HBm0u98EJHan+nle2WOZMVGItTa/2k1FRWwbt7iQ5dzDh5PEeZASg2UWe
hoR8L8MpNBqH6h7ZITwVTfRS4LSBvIEft7Pzhm5YJKfM+CdzDM+L9WVEGwiDAQAB
AoGAYfKxwUtEbq8uVrD3nnWhF+hk1k6KejdUq0dLYN29w8WjbCMKb9laokmqWiQ
5iZGERYxh7G4BDP8AW/+M9HXM4oqm5SEkaxhbTlgks+E1s9dTpdFQvL76TvodqSy
l2E2BghVgLLgkdhrN9buaFzYta95JKfgyKGonNxsQA39PwECQQDKbG0Kp6KEkNgB
srCq3Cx2od5OfiPDG8g3RYZKx/O9dMy5CM160DwusVJpuywbpRhcWr3gkz0QgRMd
IRVwyxNbAkeAyh3sipmcgN7SD8xBG/MtBYPqWP1vxhSVYPfJzuPU3gS5MRJzQHBz
sVCLhTBY7hHSOqilqWYasi81JzBEwEuQQJBaKw9qGcZjyMH8JU5TDSGllr3jybx
FFMPj8TgJs346AB8ozqLL/ThvWPpxHttJbH8QAdNuyWdg6dIfVAa95h7Y+MCQEZg
jRDl1Bz7eWGO2c0Fq9OTz3IVLWpnmGwfw+HyaxizxFhV+FOj1GUVir9hylV7V0DU
Qjlajyv/oeDWhFQ9wQECQCydHJ6NaNQOCZh+6QTrH3TC5MeBA1Yeipoe7+BhsLNr
cFG8s9sTxRnltcZl1dXaBSemvpNvBizn0Kzi8G3ZAgc=
```

-----END RSA PRIVATE KEY-----

### myPOS test public certificate

-----BEGIN CERTIFICATE-----

```
MIIBkDCB+qADAgECAGAwDQYJKoZIhvcNAQEFBQAwDzENMAAsGA1UEChMEaVBheTAe
Fw0xMzAzMTMxMTI1MTFaFw0yMzAzMTExMTI1MTFaMA8xDTALBgNVBAoTBGQYXkw
gZ8wDQYJKoZIhvcNAQEBBQADgY0AMIGJAoGBAML+VTmiY4yChoOTMZTXAIG/mk+x
f/9mjwHxWzxtBJbNncNK0OLi0VXYKW2GgVklGHHQjvew1hTFkEGjnCJ7f5CDnbgx
evtyASDGst92a6xcAedEadP0nFXhUz+cYYIglcgfDcX3ZWeNEF5kscqy52kpD2O7
nFNCV+85vS4duJBNAgMBAAEwDQYJKoZIhvcNAQEFBQADgYEAfSfqJHH9Vp9Y4osJ
sLg1Um5LOoTgn6u4JepHMFoSiwYE0n/N3D3JlgqAjdVJ+1rZV95Vaf/+TKzWzvP
V8L01LJ8aRFkUaPGenVsGvBT2mtsbu34QUOIPgzCi3huidwk0yIMX7zo8uxu1cXv
/bg5jBGe5SjvJP8Tq257QcAGgkA=
```

-----END CERTIFICATE-----

## Appendix III – Example for successful payment notification

### Result POST data

```
Array (  
  [IPCmethod] => IPCPurchaseNotify  
  [SID] => 0000000000000010  
  [Amount] => 50  
  [Currency] => EUR  
  [OrderID] => 1440146333  
  [IPC_Trnref] => 813705  
  [RequestSTAN] => 000006  
  [RequestDateTime] => 2015-08-21 10:39:37  
  [Signature] =>  
aPn96fYghN/hdLgP0oGbCglBACwJlG6TSIf7UJP9Qe9UMzBP9+B5r7enaLZNAWAHaws7iZH9INgQrHbnn/laIKFYzjWL  
HqCQm2xYPF4IAbln5MC9UjfUXyMxZ42ENVx/Poeq0ZH1Zd34j2qbFJ8tViawUjey5BZRleJaEmQr3ww=  
);
```