INTERNEL
API DOCUMENTATION

GENERAL INFORMATION AND DESCRIPTION OF THE INTERNEL WEB SERVICE
Version 1.4

DOCUMENT CONTENTS

INTRODUCTION

This document contains the full description of Internel's API and is intended as a reference documentation for a customer's own programmers or an external IT service provider.

Data exchange basis

- The data exchange with Internel's web service is based on SOAP protocols
- Exchanged data is collected in predefined XML structures
- The exchanged data can contain only Windows 1250-character sets

Access for testing purposes

We are happy to provide you access to our test and later to our production server. In case you are a new client, please contact your Internel sales representative, who will send you the necessary access details for the test server. In case you are already amongst our happy customers, please contact your designated customer care team for system access.

System authorisation

Authorisation for customers are made automated and with every system request – except as for the test connection – by sending credentials as parameters. Once you work on the live system, you shall receive your user name and password from your designated customer care team or from our IT department directly.

Once the start-up credentials were received, please change to a new password immediately by considering the following password restrictions:

- Passwords must contain at least one of each of the following: small letter, capital letter, number and special character
- Passwords cannot contain the same characters next to each other, no matter if small or capital letter, number or special character
- Passwords must have a length of 12 to 26 characters
- Passwords are only valid 30 days and shall be changed before the end of this period
- New passwords cannot be the same as a passwords used for the last 6 months
- Passwords cannot be changed more than once a day

In case of a successful authorisation, every web service function will return the password expiration time as an element of the XML response.
In case of problems during the authorisation process, instead of the password expiration time, an error element will be returned with an error message.

Data safety and security

The data exchange with our production server uses a secure internet connection, whereas data exchange with the test server is not secured. Therefore, please do not send us sensitive or real consumer data whilst testing.
Due to security reasons, we must refuse connections with servers from unknown IP addresses. For access to our test or production servers, you will need to provide us you're your static public IP address of your software to ensure white listing on our side.

Working with different API versions

The current API version is advised as a parameter in every request and its response. Even we strive to ensure that input and output structures remain the same in the various versions, please ensure you work with the latest of our API versions – a version summary is available on the last page of this document.

Error handling

Most errors are reported in a standard XML response from the web service, covered in the section "Errors" in each paragraph. In case of problems with specific parameters of the request, for example an incorrect XML data input, a SOAP fault element will be returned, instead of a standard XML response.

Available functions

Internel's web service contains a great variety of functions, allowing you to fully integrate into our system. Available functions are:

- Test connection
- Test credentials
- Change password
- Transfer outgoing orders
- Retrieve outgoing order status
- Transfer incoming deliveries
- Retrieve incoming delivery order status
- Remove incoming deliveries
- Transfer vendor data
- Retrieve vendor data
- Update vendor data
- Transfer product data
- Retrieve product data
- Update product data
- Retrieve stock data

General tips and quick start information

After receiving the access credentials, please ensure to immediately change the password as outlined earlier on in this document.

We provide functions for a test connection and credentials, to ensure that basic connection is possible.

For new accounts, the first step shall be to add product vendors, as they are required during the product definition. After that, you may add your product list to the system.

Please make sure, that product and vendor data are correct, before adding them to the system – not all of data sets can be changed via API at a later stage.

Once vendors and products were defined, the exchange of order and incoming delivery data may begin. Please note, that if you send an order/incoming delivery with an undefined product to the system, the whole data section will be rejected.

TEST CONNECTION

Request name: 'testConnection'. This function allows testing of the connection with Internel.

Data input

The input 'testConnection' uses the string 'name' as an optional parameter outlined below:

Parameter	Description	Type	Required	Example
Name	Username	String	No	LeBron

Data output

As output, a simple XML data string is returned.

Exameple – testConnection request

```
<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelope
 xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
 xmlns:ns1="http://soapServer/"
 xmlns:xsd="http://www.w3.org/2001/XMLSchema"
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xmlns:SOAP-ENC="http://schemas.xmlsoap.org/soap/encoding/"
 SOAP-ENV:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
  <SOAP-ENV:Body>
    <ns1:testConnection>
      <name xsi:type="xsd:string">LeBron</name>
    </ns1:testConnection>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Example – testConnection response

```
<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelope
 xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
 xmlns:ns1="http://soapServer/"
 xmlns:xsd="http://www.w3.org/2001/XMLSchema"
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xmlns:SOAP-ENC="http://schemas.xmlsoap.org/soap/encoding/"
 SOAP-ENV:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
  <SOAP-ENV:Body>
    <ns1:testConnectionResponse>
      <return xsi:type="xsd:string">Hello LeBron! You have successfully connected to Internel Web Service</return>
    </ns1:testConnectionResponse>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

TEST CREDENTIALS

Request name: 'testCredentials'. This is a function to test Internel's system credentials.

Data input

The input 'testCredentials' requires a testCredentialsDataType object, parameters as follows:

Parameter	Description	Type	Required	Example
authLogin	Username	String	Yes	Testuser
authPassword	Password	String	Yes	Testpassword
version	API Version	String(X.Y)	Yes	1.4

Data output

As output a string with XML data in a predefined structure (XML response) is returned, named TestCredentialsResponse and containing the following elements:

Parameter	Description	Type	Compulsory	Example
PasswordExpirationTime	Password expiration period (days)	Num(2)	Yes	17
Success	Success message	String	Yes	Hello testuser...

Errors

In case of an error, the XML response contains error elements such as:

Parameter	Description	Type	Compulsory	Example
Error			Yes	
> ErrorCode	Error code	String	Yes	21.114
> ErrorMsg	Error message	String	Yes	Login error...

Possible error messages:

- Login error - Could not find user credentials
- Login error - Password expired

Example – testCredentials request

```xml
<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelope
 xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
 xmlns:ns1="http://soapServer/"
 xmlns:xsd="http://www.w3.org/2001/XMLSchema"
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xmlns:SOAP-ENC="http://schemas.xmlsoap.org/soap/encoding/"
 SOAP-ENV:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
  <SOAP-ENV:Body>
    <ns1:testCredentials>
```

```
    <obj xsi:type="ns1:testCredentialsDataType">
      <authLogin xsi:type="xsd:string">testuser</authLogin>
      <authPassword xsi:type="xsd:string">testpassword</authPassword>
      <version xsi:type="xsd:string">1.4</version>
    </obj>
   </ns1:testCredentials>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Example – testCredentials response (successful)

```
<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelope
 xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
 xmlns:ns1="http://soapServer/"
 xmlns:xsd="http://www.w3.org/2001/XMLSchema"
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xmlns:SOAP-ENC="http://schemas.xmlsoap.org/soap/encoding/"
 SOAP-ENV:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
  <SOAP-ENV:Body>
   <ns1:testCredentialsResponse>
    <return xsi:type="xsd:string">
     &lt;?xml version="1.0" encoding="UTF-8" standalone="yes"?&gt;
     &lt;TestCredentialsResponse&gt;
      &lt;PasswordExpirationTime&gt;17&lt;/PasswordExpirationTime&gt;
      &lt;Success&gt;Hello testuser! You have successfully connected to the web service.&lt;/Success&gt;
     &lt;/TestCredentialsResponse&gt;
    </return>
   </ns1:testCredentialsResponse>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Example – testCredentials response (error)

```
...
   <ns1:testCredentialsResponse>
    <return xsi:type="xsd:string">
     &lt;?xml version="1.0" encoding="UTF-8" standalone="yes"?&gt;
     &lt;TestCredentialsResponse&gt;
      &lt;Error&gt;
       &lt;ErrorCode&gt;21.114&lt;/ErrorCode&gt;
       &lt;ErrorMsg&gt;Login error - Could not find user credentials&lt;/ErrorMsg&gt;
      &lt;/Error&gt;
     &lt;/TestCredentialsResponse&gt;
    </return>
   </ns1:testCredentialsResponse>
...
```

CHANGE PASSWORD

Request name: ‘changePassword’. This function is used for changing a password.

Data input

The input ‘changePassword’ requires a changePasswordDataType object, same as the testCredentialDataType object, but with an additional 4th parameter:

Parameter	Description	Type	Required	Example
authLogin	Username	String	Yes	Testuser
authPassword	Password	String	Yes	Testpassword
version	API Version	String(X.Y)	Yes	1.4
newPassword	New password	String	Yes	Qq!1Ww@2Ee#3

Data output

As output, a string with XML data (XML response) is returned, with its main element ChangePasswordResponse and the password expiration time, re-set to 30 days:

Parameter	Description	Type	Compulsory	Example
PasswordExpirationTime	Password expiration period (days)	Num(2)	Yes	30

Errors

In case of an error, XML response error messages are as follows:

- New password must be different than the current password
- Password must contain at least one small letter, capital letter, number and special character
- Password cannot contain the same characters next to each other
- Password must have at least 12 characters, but no more than 26
- New password must be different then the passwords from the last 6 months
- Password cannot be changed more than one time a day

Example – changePassword request

```
...
  <ns1:changePassword>
    <obj xsi:type="ns1:changePasswordDataType">
      <authLogin xsi:type="xsd:string">testuser</authLogin>
      <authPassword xsi:type="xsd:string">testpassword</authPassword>
      <version xsi:type="xsd:string">1.4</version>
      <newPassword xsi:type="xsd:string">newpassword</newPassword>
    </obj>
  </ns1:changePassword>
...
```

Example – XML response (successful)


```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ChangePasswordResponse>
  <PasswordExpirationTime>30</PasswordExpirationTime>
</ChangePasswordResponse>
```


Example – XML response (error)


```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ChangePasswordResponse>
  <PasswordExpirationTime>17</PasswordExpirationTime>
  <Error>
    <ErrorCode>12.109</ErrorCode>
    <ErrorMsg>Password cannot be changed more than one time a day</ErrorMsg>
  </Error>
</ChangePasswordResponse>
```

TRANSFER OUTGOING ORDERS

Request name: 'newOrder'. Each order shall include a unique reference number as well as all customer order data. Once B2C orders are transferred to Internel's system, the order is generated and packing lists, invoices and/or return forms printed. Per transfer request, the maximum number of orders is 50.

Data input

The input 'newOrder' requires a generalDataType object with the following parameters:

Parameter	Description	Type	Required	Example
authLogin	Username	String	Yes	testuser
authPassword	Password	String	Yes	testpassword
version	API Version	String(X.Y)	Yes	1.4
xmlData	Data of orders	String	Yes	<?xml...

The function xmlData is a string with data in XML format, its main element must be named NewOrder and contain the following parameters:

Element	Description	Type	Required	Example
Order	Max 50 per request		Yes	
> OrderID	Order reference number	Str(20)	Yes	698429
> ShipmentMethod		Enum	Yes	REGULAR
> PaymentMethod	Transfer – only confirmed payments	Enum	Yes	CASH
> OrderDateTime		Date(YYYY-MM-DD HH:MM:SS)	No	2017-07-26 14:31:25
> Client			Yes	
>> CustomerID		Str(20)	No	35649826
>> ClientFirstName		Str(20)	Yes	Adam
>> ClientLastName		Str(20)	Yes	Kowalski
>> CompanyName		Str(20)	No	Big Company
>> ShippingAddress			Yes	
>>> Country	ISO 3166-1 alpha-2 codes	Str(2)	Yes	PL
>>> City		Str(30)	Yes	Warszawa
>>> ZipCode		Str(10)	Yes	02-315
>>> Street		Str(30)	Yes	Main Street

>>> HouseNumber		Str(5)	Yes	59A
>>> ApartmentNumber		Str(8)	No	7
>> Email		Str(40)	Yes	email@email.pl
>> PhoneNumber		Str(15)	Yes	+48123123132
> Product			Yes	
>> ProductID		Str(25)	Yes	69654985
>> ProductName		Str(100)	No	Czajnik
>> Quantity		Numeric(10)	Yes	2
>> DetailedPrice			Yes	
>>> ValueNet	Per single unit	Numeric(10.2)	No	100.00
>>> ValueVAT	Per single unit	Numeric(10.2)	No	23.00
>>> ValueGross	Per single unit	Numeric(10.2)	Yes	123.00
>> VAT		Numeric(2)	Yes	23
> PayableTotal	Total cost, including shipping (COD)	Numeric(10.2)	Yes	256.00
> ValueCashTotal	Gross value of all items	Numeric(10.2)	Yes	246.00
> ValueNetTotal	Net value of all items	Numeric(10.2)	No	200.00
> ValueVATTotal	VAT total of all items	Numeric(10.2)	No	46.00
> ValueShippingTotal	Cost of shipping (may be 0)	Numeric(10.2)	Yes	10.00

ShipmentMethod possible values are:
- REGULAR – regular delivery

PaymentMethod possible values are:
- CASH – cash on delivery service
- TRANSFER – payment in advance in any mean, i.e. credit card, bank transfer, PayPal

All products in XML format must have a ValueGross element, with the price of a single product. whereas ValueNet and ValueVAT are only for verification purposes. If used, their sum must be equal to ValueGross. The PayableTotal is the total cost of an order and must be equal to the sum of ValueCashTotal and ValueShippingTotal. ValueNetTotal and ValueVATTotal are for verification only, if used, their sum must be equal to ValueCashTotal.

Data output

As output, the string with XML data (XML response) is named NewOrderResponse and contains the following elements:

Parameter	Description	Type	Compulsory	Example
PasswordExpirationTime	Password expiration period (days)	Num(2)	Yes	17
Order			Yes	
> OrderID	Order reference number	Str(20)	Yes	698429
> Status	Status of order	Enum	Yes	ADDED

Possible status values are:

- ADDED – the order was successfully added to the Internel system
- OMITTED – an error occurred; the order was omitted/ignored

Errors (XML response)

In case of an error, the XML response will contain an error message such as:

- Missing customer first and the last name. Import of order omitted.
- Missing product {ProductID} in database. Import of order omitted.
- Undefined ShipmentMethod /PaymentMethod. Import of order omitted
- Order already exists. Import for order omitted

Errors (SOAP fault)

In case of incorrect xmlData, a SOAP fault will be returned before processing an order. The SOAP fault element contains one or more fault codes/fault messages such as:

- XML can contain only characters from Windows 1250-character set
- Invalid XML. Name of the main XML element should be {X} instead of {Y}
- Invalid XML. Number of orders in one XML should be between 1-50
- Invalid XML. Value of {X} element cannot be {Y}. Allowed values are: {A B C}
- Invalid XML. {X} cannot be empty
- Invalid XML. Value of {X} element cannot be {Y}. Only integer, max {A} digits
- Invalid XML. Value of {X} element cannot be {Y}. Only numbers in {A.B} format

Example – xmlData

```xml
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<NewOrder>
 <Order>
  <OrderID>698429</OrderID>
  <ShipmentMethod>BY_NOON</ShipmentMethod>
  <PaymentMethod>CASH</PaymentMethod>
  <OrderDateTime>2018-06-25 12:50:53</OrderDateTime>
  <Client>
   <ClientFirstName>Adam</ClientFirstName>
   <ClientLastName>Kowalski</ClientLastName>
   <ShippingAddress>
    <Country>PL</Country>
    <City>Warszawa</City>
    <ZipCode>02-315</ZipCode>
    <Street>Main Street</Street>
    <HouseNumber>59A</HouseNumber>
   </ShippingAddress>
   <Email>email@email.com</Email>
```

```xml
      <PhoneNumber>+48123123123</PhoneNumber>
    </Client>
    <Product>
      <ProductID>69654985</ProductID>
      <ProductName>Czajnik</ProductName>
      <Quantity>2</Quantity>
      <DetailedPrice>
        <ValueGross>123.00</ValueGross>
      </DetailedPrice>
      <VAT>23</VAT>
    </Product>
    <PayableTotal>256.00</PayableTotal>
    <ValueCashTotal>246.00</ValueCashTotal>
    <ValueShippingTotal>10.00</ValueShippingTotal>
  </Order>
  <Order>
    <OrderID>698430</OrderID>
    <ShipmentMethod>REGULAR</ShipmentMethod>
    <PaymentMethod>TRANSFER</PaymentMethod>
    <OrderDateTime>2018-06-25 12:58:51</OrderDateTime>
    <Client>
      <ClientFirstName>Jaromir</ClientFirstName>
      <ClientLastName>Nazwisko</ClientLastName>
      <ShippingAddress>
        <Country>PL</Country>
        <City>Kraków</City>
        <ZipCode>32-080</ZipCode>
        <Street>Sesame Street</Street>
        <HouseNumber>11</HouseNumber>
      </ShippingAddress>
      <Email>mail@mail.com</Email>
      <PhoneNumber>+48321321321</PhoneNumber>
    </Client>
    <Product>
      <ProductID>4321</ProductID>
      <ProductName>Basketball Jersey</ProductName>
      <Quantity>1</Quantity>
      <DetailedPrice>
        <ValueGross>109.99</ValueGross>
      </DetailedPrice>
      <VAT>23</VAT>
    </Product>
    <Product>
      <ProductID>4356</ProductID>
      <ProductName>Football Jersey</ProductName>
      <Quantity>1</Quantity>
      <DetailedPrice>
        <ValueGross>109.99</ValueGross>
      </DetailedPrice>
      <VAT>23</VAT>
    </Product>
    <PayableTotal>219.98</PayableTotal>
    <ValueCashTotal>219.98</ValueCashTotal>
    <ValueShippingTotal>0</ValueShippingTotal>
  </Order>
</NewOrder>
```


Example – XML response (successful)


```xml
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<NewOrderResponse>
  <PasswordExpirationTime>25</PasswordExpirationTime>
  <Order>
    <OrderID>698429</OrderID>
    <Status>ADDED</Status>
  </Order>
  <Order>
    <OrderID>698430</OrderID>
    <Status>ADDED</Status>
  </Order>
</NewOrderResponse>
```

Example – XML response (error)


```xml
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<NewOrderResponse>
  <PasswordExpirationTime>25</PasswordExpirationTime>
  <Order>
    <OrderID>698431</OrderID>
    <Error>
      <ErrorCode>18.120</ErrorCode>
      <ErrorMsg>Order already exists. OrderID: 698431. Import for order omitted</ErrorMsg>
    </Error>
    <Status>OMITTED</Status>
  </Order>
  <Order>
    <OrderID>698431</OrderID>
    <Status>ADDED</Status>
  </Order>
</NewOrderResponse>
```


Example – newOrder response (SOAP fault)


```xml
<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
  <SOAP-ENV:Body>
    <SOAP-ENV:Fault>
      <faultcode>22.161</faultcode>
      <faultstring>Invalid XML. Element OrderID is required and must be single on its level.</faultstring>
    </SOAP-ENV:Fault>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

RETRIEVE OUTGOING ORDER STATUS

Request name: 'orderStatus'. This function allows the retrieving of order status, status dates and tracking information. Each request must contain a unique order reference number (OrderID). The maximum number of orders per request is 50.

There are two ways to retrieve the order status, depending on the StatusMode option:

- SHOW LATEST – returns only the latest statuses, changed since the last orderStatus request. This option is set as default and no new statuses will be reported for previously reported orders with no status change, such orders will be omitted in the response.
- SHOW ALL – returns all order statuses with dates.

Possible status of orders:

- ORDER_TRANSFERED_TO_FULFILLER – order successfully sent
- ORDER_CANCELLED – order cancelled
- ORDER_PROCESSED – order processed
- ORDER_TRANSFERRED_TO_CARRIER – order collected by carrier/courier company
- ORDER_SHIPPED – order dispatched to be delivered
- ORDER_COMPLETED – order delivered
- ORDER_RETURNED – order returned by customer

Data input

The input 'orderStatus' requires a generalDataType object with parameters such as:

Parameter	Description	Type	Required	Example
authLogin	Username	String	Yes	Testuser
authPassword	Password	String	Yes	testpassword
version	API Version	String(X.Y)	Yes	1.4
xmlData	Data of orders	String	Yes	<?xml...

The function xmlData is a string with data in XML format, its main element must be named OrderStatus with the following parameters:

Element	Description	Type	Required	Example
StatusMode	Status mode	Enum	No	SHOW ALL
Order			Yes	
> OrderID	Order reference number	Str(20)	Yes	698429

StatusMode allowed values are:

- SHOW LATEST
- SHOW ALL

Data output

As output, a string with XML data (XML response) is returned, with its main XML response element named OrderStatusResponse, containing the following elements:

Parameter	Description	Type	Compulsory	Example
PasswordExpirationTime	Password expiration period (days)	Num(2)	Yes	17
Order			Yes	
> OrderID	Order reference number	Str(20)	Yes	698429
> TrackingNumber		String	No	0000000329744Q
> ShipmentProvider		String	No	DPD
> OrderStatus			No	
>> Status	Status of order	String	Yes	NEW_ORDER
>> Date	Date of status change	Date(YYYY-MM-DD HH:MM:SS)	Yes	2017-07-26 14:31:25
>> ReturnedPositions	List of returned products		No	
>>> Product			Yes	
>>>> ProductID		Str(25)	Yes	69654985
>>>> Quantity		Numeric(10)	Yes	4

Errors

In case of an error, the XML response contains an error message, in case of incorrect xmlData, a SOAP fault will be returned. Example XML response error message:

- Order not found in database. OrderID: {orderID}

Example – xmlData

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<OrderStatus>
 <Order>
  <OrderID>698428</OrderID>
 </Order>
 <Order>
  <OrderID>698429</OrderID>
 </Order>
 <Order>
  <OrderID>698430</OrderID>
 </Order>
</OrderStatus>
```

Example – XML response (error)

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<OrderStatusResponse>
 <PasswordExpirationTime>24</PasswordExpirationTime>
 <Order>
  <OrderID>698428</OrderID>
  <Error>
   <ErrorCode>23.132</ErrorCode>
   <ErrorMsg>Order not found in database. OrderID: 698428</ErrorMsg>
  </Error>
 </Order>
 <Order>
  <OrderID>698429</OrderID>
  <TrackingNumber>0000028461905Q </TrackingNumber>
```

```xml
    <ShipmentProvider>DPD</ShipmentProvider>
    <OrderStatus>
      <Status>ORDER_TRANSFERED_TO_FULFILLER</Status>
      <Date>2018-01-08 14:35:55</Date>
    </OrderStatus>
    <OrderStatus>
      <Status>ORDER_PROCESSED</Status>
      <Date>2018-01-09 07:56:51</Date>
    </OrderStatus>
    <OrderStatus>
      <Status>ORDER_TRANSFERRED_TO_CARRIER</Status>
      <Date>2018-01-09 14:12:00</Date>
    </OrderStatus>
    <OrderStatus>
      <Status>ORDER_SHIPPED</Status>
      <Date>2018-01-10 08:02:17</Date>
    </OrderStatus>
    <OrderStatus>
      <Status>ORDER_COMPLETED</Status>
      <Date>2018-01-10 13:33:07</Date>
    </OrderStatus>
    <OrderStatus>
      <Status>ORDER_RETURNED</Status>
      <Date>2018-01-14 10:24:52</Date>
      <ReturnedPositions>
        <Product>
          <ProductID>6452718</ProductID>
          <Quantity>5</Quantity>
        </Product>
        <Product>
          <ProductID>6375676</ProductID>
          <Quantity>1</Quantity>
        </Product>
      </ReturnedPositions>
    </OrderStatus>
  </Order>
</OrderStatusResponse>
```

TRANSFER INCOMING DELIVERIES

Request name: 'newDelivery'. This function allows the advising incoming deliveries to Internel. The maximum number of deliveries per request is 50. Each delivery shall contain full delivery data with a product list of products already defined in the system before. If a product does not exist, it will be removed whilst importing delivery data; the delivery will be processed without it.

Data input

The input 'newDelivery' requires a generalDataType object with parameters such as:

Parameter	Description	Type	Required	Example
authLogin	Username	String	Yes	testuser
authPassword	Password	String	Yes	testpassword
version	API Version	String(X.Y)	Yes	1.4
xmlData	Data of deliveries	String	Yes	<?xml...

The function xmlData is a string with data in XML format; its main element must be named NewDelivery, with the following parameters:

Element	Description	Type	Required	Example
Delivery	Max 50 per request		Yes	
> DeliveryID	Delivery reference number	Str(20)	Yes	12341234
> DeliveryDate		Date(YYYY-MM-DD)	Yes	2017-07-26
> Vendor	Only for verification purposes		No	
>> VendorID		Str(20)	Yes	02-1234
> Product			Yes	
>> ProductID		Str(25)	Yes	64589317
>> Quantity		Numeric(10)	Yes	50
>> ValueNet	Per single unit	Numeric(10.2)	Yes	81.29

A supplier (VendorID) is assigned to each product and provided in a 'newProduct' request, among the rest of the product specification. It is understood, that products in each delivery must have the same supplier. A delivery containing products from different suppliers will not be processed and an error message will be generated. In order to verify the supplier for a delivery, the vendor element may be included in the xmlData. With a defined VendorID, all products in one delivery must have the same VendorID assigned.

Data output

As output, a string with XML data (XML response) is returned, its main element is named NewDeliveryResponse and contains the following elements:

Parameter	Description	Type	Compulsory	Example
PasswordExpirationTime	Password expiration period (days)	Num(2)	Yes	17
Delivery			Yes	
> DeliveryID	Order reference number	Str(20)	Yes	698429
> Status	Status of delivery	Enum	Yes	ADDED

Possible statuses of a delivery are:

- ADDED – the delivery was successfully added to the system
- OMITTED – the delivery was omitted

Errors

In case of an error, the XML response includes an error message. In case of incorrect xmlData, a SOAP fault will be returned. If a product is not defined in the system, it will be removed from the delivery, but such delivery will be processed by the system (unless a different error occurs).

Possible error messages are:

- Missing product {ProductID} in database, for delivery {DeliveryID}. Product removed from delivery.
- VendorID for delivery {DeliveryID} doesn't match with product vendor (ProductID: {ProductID}). Delivery omitted.
- Products in delivery {DeliveryID} have different vendors. Delivery omitted.
- Vendor {VendorID} not found in database, for delivery {DeliveryID}. Delivery omitted.
- Delivery {DeliveryID} is empty. Delivery omitted.
- Delivery {DeliveryID} already exists. Delivery omitted.

Example – xmlData

```xml
<?xml version='1.0' encoding='UTF-8' standalone='yes'?>
<NewDelivery>
 <Delivery>
  <DeliveryID>111111</DeliveryID>
  <DeliveryDate>2018-07-22</DeliveryDate>
  <Vendor>
   <VendorID>2122</VendorID>
  </Vendor>
  <Product>
   <ProductID>1234</ProductID>
   <Quantity>50</Quantity>
   <ValueNet>123.12</ValueNet>
  </Product>
  <Product>
   <ProductID>4321</ProductID>
   <Quantity>100</Quantity>
   <ValueNet>60.56</ValueNet>
  </Product>
 </Delivery>
 <Delivery>
  <DeliveryID>111112</DeliveryID>
  <DeliveryDate>2018-07-25</DeliveryDate>
  <Product>
```

```xml
      <ProductID>777</ProductID>
      <Quantity>20</Quantity>
      <ValueNet>22.13</ValueNet>
    </Product>
    <Product>
      <ProductID>888</ProductID>
      <Quantity>30</Quantity>
      <ValueNet>9.68</ValueNet>
    </Product>
    <Product>
      <ProductID>999</ProductID>
      <Quantity>40</Quantity>
      <ValueNet>14.20</ValueNet>
    </Product>
  </Delivery>
  <Delivery>
    <DeliveryID>111113</DeliveryID>
    <DeliveryDate>2018-07-27</DeliveryDate>
    <Vendor>
      <VendorID>3133</VendorID>
    </Vendor>
    <Product>
      <ProductID>4567</ProductID>
      <Quantity>20</Quantity>
      <ValueNet>60</ValueNet>
    </Product>
    <Product>
      <ProductID>5678</ProductID>
      <Quantity>20</Quantity>
      <ValueNet>40</ValueNet>
    </Product>
  </Delivery>
</NewDelivery>
```


Example – XML response (error)


```xml
<?xml version='1.0' encoding='UTF-8' standalone='yes'?>
<NewDeliveryResponse>
  <PasswordExpirationTime>13</PasswordExpirationTime>
  <Delivery>
    <DeliveryID>111111</DeliveryID>
    <Error>
      <ErrorCode>24.112</ErrorCode>
      <ErrorMsg>VendorID for delivery 111111 doesn't match with product vendor (ProductID: 1234). Delivery omitted.</ErrorMsg>
    </Error>
    <Status>OMITTED</Status>
  </Delivery>
  <Delivery>
    <DeliveryID>111112</DeliveryID>
    <Status>ADDED</Status>
  </Delivery>
  <Delivery>
    <DeliveryID>111113</DeliveryID>
    <Error>
      <ErrorCode>24.112</ErrorCode>
      <ErrorMsg>Missing product 4567 in database, for delivery 111113. Product removed from delivery.</ErrorMsg>
    </Error>
    <Status>ADDED</Status>
  </Delivery>
</NewDeliveryResponse>
```

RETRIEVE INCOMING DELIVERY ORDER STATUS

Request name: 'deliveryStatus'. This function allows the retrieving of status of incoming deliveries. Each request must contain the StatusMode or IDs of the requested deliveries and the maximum number of deliveries per request is 50.

Depending on the presence and value of the StatusMode element, there are few different ways to retrieve the status of deliveries:

- LAST – retrieves the last 10 deliveries
- LAST XX – retrieves the last XX deliveries, where XX is a number in (1-99) range
- AWAIT – retrieves all deliveries with AWAIT status from the last month
- PROCESSING – retrieves all deliveries with PROCESSING status from the last month
- PARTIAL - retrieves all deliveries with PARTIAL status from the last month
- COMPLETED - retrieves all deliveries with COMPLETED status from the last month

Possible delivery status:

- NOT_FOUND – delivery not found in the database.
- AWAIT – delivery is in the Internel's system and is waiting for processing.
- PROCESSING – delivery is being processed. That may happen, when a delivery being processed by Internel's team. A delivery with PROCESSING status cannot be removed.
- PARTIAL – delivery is partially completed. That may happen, if a delivery arrives with either excess items or missing units compared to the original advised quantities or packing list. A delivery with PARTIAL status cannot be removed.
- COMPLETED – delivery is completed. Delivery with COMPLETED status cannot be removed.

Regardless of the presence of the StatusMode element, DeliveryIDs can be provided in the separate delivery elements for the status of specific deliveries.

Data input

The input 'deliveryStatus' requires a generalDataType object with the following parameters:

Parameter	Description	Type	Required	Example
authLogin	Username	String	Yes	testuser
authPassword	Password	String	Yes	testpassword
version	API Version	String(X.Y)	Yes	1.4
xmlData	Data of deliveries	String	Yes	<?xml...

The function xmlData is a string with data in XML format, its main element must be named DeliveryStatus with the following parameters:

Element	Description	Type	Required	Example
StatusMode	Status mode	Enum	No*	LAST_20
Delivery			No*	
> DeliveryID	Delivery reference number	Str(20)	Yes	d-101

* For StatusMode and Delivery at least one of them must be included within xmlData.

StatusMode allowed values are:

- LAST
- LAST_XX
- AWAIT
- PROCESSING
- PARTIAL
- COMPLETED

Data output

As output, a string with XML data (XML response) is returned; its main response element is named DeliveryStatusResponse and contains the following elements:

Parameter	Description	Type	Compulsory	Example
PasswordExpirationTime	Password expiration period (days)	Num(2)	Yes	17

Example – XML response


```xml
<DeliveryStatusResponse>
 <PasswordExpirationTime>17</PasswordExpirationTime>
 <Delivery>
  <DeliveryID>155201</DeliveryID>
  <Status>AWAIT</Status>
  <Date>2018-07-24</Date>
 </Delivery>
 <Delivery>
  <DeliveryID>155202</DeliveryID>
  <Status>AWAIT</Status>
  <Date>2018-07-13</Date>
 </Delivery>
 <Delivery>
  <DeliveryID>d-101</DeliveryID>
  <Status>NOT_FOUND</Status>
 </Delivery>
 <Delivery>
  <DeliveryID>d-102</DeliveryID>
  <Status>COMPLETED</Status>
  <Date>2018-01-07</Date>
 </Delivery>
 <Delivery>
  <DeliveryID>d-103</DeliveryID>
  <Status>PARTIAL</Status>
  <Date>2018-07-25</Date>
 </Delivery>
</DeliveryStatusResponse>
```

REMOVE INCOMING DELIVERIES

Request name: 'removeDelivery'. This is a function for removing incoming deliveries. Removing a delivery is only possible if the status of the delivery is AWAIT. Completed, partially completed or processed deliveries cannot be removed. Each request must contain the IDs of the requested deliveries. The maximum number of deliveries per request is 50.

Data input

The input 'removeDelivery' requires a generalDataType object and parameters as follows:

Parameter	Description	Type	Required	Example
authLogin	Username	String	Yes	Testuser
authPassword	Password	String	Yes	Testpassword
version	API Version	String(X.Y)	Yes	1.4
xmlData	Data of deliveries	String	Yes	<?xml...

The function xmlData is a string with data in XML format, its main element must be named RemoveDelivery and shall contain the following delivery data:

Element	Description	Type	Required	Example
Delivery			Yes	
> DeliveryID	Delivery reference number	Str(20)	Yes	d-101

Data output

As output, a string with XML data (XML response) is returned, its main element is named RemoveDeliveryResponse and contains the following elements:

Parameter	Description	Type	Compulsory	Example
PasswordExpirationTime	Password expiration period (days)	Num(2)	Yes	17
Delivery			Yes	
> DeliveryID	Delivery reference number	Str(20)	Yes	d-101
> Status	Status of operation	Enum	Yes	REMOVED

Possible statuses are:

- REMOVED – delivery was successfully removed
- OMITTED – delivery removal was omitted

Errors

In case of an error, no removal will be done, the status will be OMITTED; the response contains an error message. In case of incorrect xmlData, a SOAP fault will be returned.

Example error message:

- Delivery {deliveryID} status is COMPLETED. Delivery cannot be removed.
- Delivery {deliveryID} status is PARTIAL. Delivery cannot be removed.
- Delivery {deliveryID} status is PROCESSING. Delivery cannot be removed.
- Delivery {deliveryID} not found in database. Delivery cannot be removed.
- Invalid XML. Maximum number of deliveries in one XML is 50.

Example – xmlData


```xml
<?xml version='1.0' encoding='UTF-8' standalone='yes'?>
<RemoveDelivery>
 <Delivery>
  <DeliveryID>07_101</DeliveryID>
 </Delivery>
 <Delivery>
  <DeliveryID>07_102</DeliveryID>
 </Delivery>
 <Delivery>
  <DeliveryID>07_103</DeliveryID>
 </Delivery>
 <Delivery>
  <DeliveryID>07_104</DeliveryID>
 </Delivery>
</RemoveDelivery>
```


Example – XML response (error)


```xml
<RemoveDeliveryResponse>
 <PasswordExpirationTime>17</PasswordExpirationTime>
 <Delivery>
  <DeliveryID>07_101</DeliveryID>
  <Status>REMOVED</Status>
 </Delivery>
 <Delivery>
  <DeliveryID>07_102</DeliveryID>
  <Status>OMITTED</Status>
  <Error>
   <ErrorCode>28.87</ErrorCode>
   <ErrorMsg>Delivery 07_102 status is PARTIAL. Delivery cannot be removed.</ErrorMsg>
  </Error>
 </Delivery>
 <Delivery>
  <DeliveryID>07_103</DeliveryID>
  <Status>OMITTED</Status>
  <Error>
   <ErrorCode>28.87</ErrorCode>
   <ErrorMsg>Delivery 07_103 status is COMPLETED. Delivery cannot be removed.</ErrorMsg>
  </Error>
 </Delivery>
 <Delivery>
  <DeliveryID>07_104</DeliveryID>
  <Status>OMITTED</Status>
  <Error>
   <ErrorCode>28.108</ErrorCode>
   <ErrorMsg>Delivery 07_104 not found in database. Delivery cannot be removed.</ErrorMsg>
  </Error>
 </Delivery>
</RemoveDeliveryResponse>
```

TRANSFER VENDOR DATA

Request name: 'newVendor'. This functions allows the transfer of new supplier data to Internel. The maximum number of vendors per request is 50.

Data input

The input 'newVendor' requires a generalDataType object with the following parameters:

Parameter	Description	Type	Required	Example
authLogin	Username	String	Yes	testuser
authPassword	Password	String	Yes	testpassword
version	API Version	String(X.Y)	Yes	1.4
xmlData	Data of vendors	String	Yes	<?xml...

The function xmlData is a string with data in XML format, its main element must be named NewVendor, with the following parameters:

Element	Description	Type	Required	Example
Vendor	Max 50 per request		Yes	
> VendorID	Vendor reference number	Str(20)	Yes	V1001
> VendorName	Name of supplier company	Str(150)	Yes	Big Company
> Country	ISO 3166-1 alpha-2 codes	Str(2)	Yes	PL
> City		Str(30)	Yes	Wrocław
> ZipCode		Str(10)	Yes	80-337
> Street		Str(40)	Yes	Kolejowa
> HouseNumber		Str(10)	Yes	12
> ApartmentNumber		Str(10)	No	14
> ContactPerson	Name of vendor representative	Str(60)	No	Jan Kowalski
> PhoneNumber		Str(20)	No	+48123123123
> Email		Str(50)	No	jan@mail.pl

Data output

The output returns a string with XML data (XML response), the main response element is named NewVendorResponse and contains the following elements:

Parameter	Description	Type	Compulsory	Example
PasswordExpirationTime	Password expiration period (days)	Num(2)	Yes	20
Vendor			Yes	
> VendorID	Vendor reference number	Str(20)	Yes	V1001
> Status	Status of vendor	Enum	Yes	ADDED

Possible vendor statuses are:

- ADDED – vendor was successfully added to the system
- OMITTED – vendor was omitted

Errors

In case of an error, the XML response contains an error message. In case of incorrect xmlData, a SOAP fault will be returned.

Example error messages:

- Invalid XML. Number of vendors in one XML should be between 1-50.
- Invalid XML. Vendor {VendorID} found twice in XML.
- Vendor {VendorID} already exists. Vendor omitted.
- Internal error. Vendor {VendorID}. Vendor omitted.

Example – xmlData


```xml
<?xml version='1.0' encoding='UTF-8' standalone='yes'?>
<NewVendor>
 <Vendor>
  <VendorID>V1001</VendorID>
  <VendorName>Big Company</VendorName>
  <Country>PL</Country>
  <City>Wrocław</City>
  <ZipCode>80-337</ZipCode>
  <Street>Kolejowa</Street>
  <HouseNumber>12</HouseNumber>
  <ApartmentNumber>14</ApartmentNumber>
  <ContactPerson>Jan Kowalski</ContactPerson>
  <PhoneNumber>+48123123123</PhoneNumber>
  <Email>jan@mail.pl</Email>
 </Vendor>
 <Vendor>
  <VendorID>V1002</VendorID>
  <VendorName>Small Company</VendorName>
  <Country>PL</Country>
  <City>Warszawa</City>
  <ZipCode>01-234</ZipCode>
  <Street>Niedługa 3/4</Street>
  <ContactPerson>Władysław Łącznik</ContactPerson>
  <PhoneNumber>+48123123123</PhoneNumber>
 </Vendor>
</NewVendor>
```

INTERNEL e-commerce logistics

Example – XML response (error)


```
<newVendorResponse>
 <PasswordExpirationTime>29</PasswordExpirationTime>
 <Vendor>
  <VendorID>V1001</VendorID>
  <Status>ADDED</Status>
 </Vendor>
 <Vendor>
  <VendorID>V1002</VendorID>
  <Status>OMITTED</Status>
  <Error>
   <ErrorCode>30.75</ErrorCode>
   <ErrorMsg>Vendor V1002 already exists. Vendor omitted.</ErrorMsg>
  </Error>
 </Vendor>
</newVendorResponse>
```


RETRIEVE VENDOR DATA

Request name: 'getVendor'. This function allows the retrieving of supplier data from Internel's system. The maximum number of vendors per request is 50.

Data input

The input 'getVendor' requires a generalDataType object with the following parameters:

Parameter	Description	Type	Required	Example
authLogin	Username	String	Yes	testuser
authPassword	Password	String	Yes	testpassword
version	API Version	String(X.Y)	Yes	1.4
xmlData	Data of vendors	String	Yes	<?xml...

The function xmlData is a string with data in XML format, its main element must be named GetVendor with the following parameters:

Element	Description	Type	Required	Example
Vendor	Max 50 per request		Yes	
> VendorID	Vendor reference number	Str(20)	Yes	V1001

Data output

As output, a string with XML data (XML response) is returned, with its main response element named GetVendorResponse and containing the following elements:

Element	Description	Type	Compulsory	Example
PasswordExpirationTime	Password expiration period (days)	Num(2)	Yes	20
Vendor			Yes	

> VendorID	Vendor reference number	Str(20)	Yes	V1002
> VendorData			Yes	
>> VendorName	Name of supplier company	Str(150)	Yes	Other Company
>> Country	ISO 3166-1 alpha-2 codes	Str(2)	Yes	PL
>> City		Str(30)	Yes	Warszawa
>> ZipCode		Str(10)	Yes	01-234
>> Street		Str(40)	Yes	Niedługa
>> HouseNumber	May be empty	Str(10)	Yes	12/14
>> ApartmentNumber	May be empty	Str(10)	Yes	46a
>> ContactPerson	May be empty	Str(60)	Yes	Henryk Przykład
>> PhoneNumber	May be empty	Str(20)	Yes	48123123123
>> Email	May be empty	Str(50)	Yes	henryk@mail.pl

Errors

In case of an error, the XML response contains an error message instead of a VendorData element. In case of incorrect xmlData, a SOAP fault will be returned.

Example error messages:

- Invalid XML. Number of vendors in one XML should be between 1-50.
- Invalid XML. Vendor {VendorID} found twice in XML.
- Vendor {VendorID} not found in database.

Example – xmlData

```
<?xml version='1.0' encoding='UTF-8' standalone='yes'?>
<GetVendor>
  <Vendor>
    <VendorID>V1001</VendorID>
  </Vendor>
  <Vendor>
    <VendorID>V1002</VendorID>
  </Vendor>
</GetVendor>
```

Example – XML response (error)

```
<GetVendorResponse>
  <PasswordExpirationTime>29</PasswordExpirationTime>
  <Vendor>
    <VendorID>V1001</VendorID>
    <VendorData>
      <VendorName>Big Company</VendorName>
      <Country>DE</Country>
      <City>München</City>
      <ZipCode>80337</ZipCode>
      <Street>Lothstraße</Street>
      <HouseNumber>12</HouseNumber>
      <ApartmentNumber>14</ApartmentNumber>e
      <ContactPerson>Hans Beispiel</ContactPerson>
```

```
    <Email>hans@mail.de</Email>
    <PhoneNumber>49123123123</PhoneNumber>
   </VendorData>
  </Vendor>
  <Vendor>
   <VendorID>V1002</VendorID>
   <VendorData>
    <VendorName>Small Company</VendorName>
    <Country>PL</Country>
    <City>Warszawa</City>
    <ZipCode>01-234</ZipCode>
    <Street>Niedługa 3/4</Street>
    <HouseNumber/>
    <ApartmentNumber/>
    <ContactPerson>Władysław Łącznik</ContactPerson>
    <Email/>
    <PhoneNumber>48123123123</PhoneNumber>
   </VendorData>
  </Vendor>
  <Vendor>
   <VendorID>V1003</VendorID>
   <Error>
    <ErrorCode>30.202</ErrorCode>
    <ErrorMsg>Vendor V1003 not found in database.</ErrorMsg>
   </Error>
  </Vendor>
</GetVendorResponse>
```

UPDATE VENDOR DATA

Request name: 'updateVendor'. This function allows the update of existing supplier data. The maximum number of vendors per request is 50. Optional vendor properties (for example vendor e-mail) can be removed by submitting an empty value.

Data input

The input 'updateVendor' requires a generalDataType object with parameters as below:

Parameter	Description	Type	Required		Example
authLogin	Username	String	Yes		Testuser
authPassword	Password	String	Yes		Testpassword
version	API Version	String(X.Y)	Yes		1.4
xmlData	Data of vendors	String	Yes		<?xml...

The function xmlData is a string with data in XML format; its main element must be named UpdateVendor with the following parameters:

Element	Description	Type	Required	Example
Vendor	Max 50 per request		Yes	
> VendorID	Vendor reference number	Str(20)	Yes	V1001
> VendorName	Name of supplier company	Str(150)	No	Big Company
> Country	ISO 3166-1 alpha-2 codes	Str(2)	No	PL
> City		Str(30)	No	Wrocław
> ZipCode		Str(10)	No	80337
> Street		Str(40)	No	Dworcowa
> HouseNumber	May be empty (in order to remove)	Str(10)	No	12
> ApartmentNumber	May be empty (in order to remove)	Str(10)	No	14
> ContactPerson	May be empty (in order to remove)	Str(60)	No	Jan Kowalski
> PhoneNumber	May be empty (in order to remove)	Str(20)	No	+48123123123
> Email	May be empty (in order to remove)	Str(50)	No	jan@mail.pl

To update vendor data, a VendorID is required. All other parameters are optional.

Data output

As string with XML data is returned (XML response); the XML response main element is named UpdateVendorResponse and contains the following elements:

Parameter	Description	Type	Compulsory	Example
PasswordExpirationTime	Password expiration period (days)	Num(2)	Yes	20
Vendor			Yes	
> VendorID	Vendor reference number	Str(20)	Yes	V1001
> Status	Status of vendor update	Enum	Yes	UPDATED

Possible vendor statuses are:

- UPDATED – vendor data was successfully updated
- OMITTED – vendor update was omitted

Errors

In case of an error, the XML response contains an error message. In case of incorrect xmlData, a SOAP fault will be returned.

Example error messages:

- Invalid XML. Number of vendors in one XML should be between 1-50.
- Invalid XML. Vendor {VendorID} found twice in XML.
- Internal error. Vendor {VendorID}. Vendor omitted.
- Vendor {VendorID} not found in database.

Example – xmlData

V1001 – **update** VendorName, Street, PhoneNumber; **remove** HouseNumber, ApartmentNumber.
V1002 – **update** PhoneNumber, Email; **remove** ContactPerson.
V1003 – **update** Street.

```
<?xml version='1.0' encoding='UTF-8' standalone='yes'?>
<UpdateVendor>
  <Vendor>
    <VendorID>V1001</VendorID>
    <VendorName>Bigger Company</VendorName>
    <Street>Dworcowa 156/13</Street>
    <HouseNumber/>
    <ApartmentNumber/>
    <PhoneNumber>+48321321321</PhoneNumber>
  </Vendor>
  <Vendor>
    <VendorID>V1002</VendorID>
    <ContactPerson/>
    <PhoneNumber>+48321321321</PhoneNumber>
    <Email>contact@smallcompany.pl</Email>
  </Vendor>
  <Vendor>
    <VendorID>V1003</VendorID>
    <Street>Sample Street 12/34</Street>
  </Vendor>
</UpdateVendor>
```

Example – XML response (error)


```xml
<UpdateVendorResponse>
  <PasswordExpirationTime>29</PasswordExpirationTime>
  <Vendor>
    <VendorID>V1001</VendorID>
    <Status>UPDATED</Status>
  </Vendor>
  <Vendor>
    <VendorID>V1002</VendorID>
    <Status>UPDATED</Status>
  </Vendor>
  <Vendor>
    <VendorID>V1003</VendorID>
    <Status>OMITTED</Status>
    <Error>
      <ErrorCode>30.139</ErrorCode>
      <ErrorMsg>Vendor V1003 not found in database.</ErrorMsg>
    </Error>
  </Vendor>
</UpdateVendorResponse>
```

TRANSFER PRODUCT DATA

Request name: 'newProduct'. This functions allows the transfer of new products to Internel. The maximum number of products per request is 50.

Data input

The input 'newProduct' requires a generalDataType object with the following parameters:

Parameter	Description	Type	Required	Example
authLogin	Username	String	Yes	testuser
authPassword	Password	String	Yes	testpassword
version	API Version	String(X.Y)	Yes	1.4
xmlData	Data of products	String	Yes	<?xml...

The function xmlData is a string with data in XML format; its main element must be named NewProduct with the following parameters:

Element	Description	Type	Required	Example
Product	Max 50 per request		Yes	
> ProductID	Product reference number	Str(20)	Yes	P0001
> ProductName	Name of product	Str(100)	Yes	Office calculator Citizen SDC-812BN
> ProductNumber	Number of product	Str(25)	No	SDC-812BN
> VendorID	Vendor reference number	Str(20)	Yes	V0001
> NetWeight	Netto Weight (in grams)	Num(8)	No	150
> GrossWeight	Gross Weight (in grams)	Num(8)	Yes	200
> ProductDimensions			No	
>> Height	In millimetres	Num(4)	Yes	30
>> Width	In millimetres	Num(4)	Yes	150
>> Length	In millimetres	Num(4)	Yes	120
> EAN	European Article Number	Str(30)	Yes	4562195133322
> OriginCountry	ISO 3166-1 alpha-2 codes	Str(2)	Yes*	CN
> HSCode	HS code	Str(20)	Yes*	847010

* Origin Country and HSCode are required only, in case of exports outside the EU.
ProductNumber is a number visible on the outer packaging or on the product itself, used for identification of the item in our warehouses. This parameter is optional.

Every product must have an assigned supplier (VendorID).
ProductDimensions are the dimensions of a product or an outer product box.

Data output
As output, a string with XML data is returned (XML response); with its main element named NewProductResponse and containing the following elements:

Parameter	Description	Type	Compulsory	Example
PasswordExpirationTime	Password expiration period (days)	Num(2)	Yes	20
Product			Yes	
> ProductID	Product reference number	Str(20)	Yes	P0001
> Status	Status of product	Enum	Yes	ADDED

Possible product statues are:
- ADDED – product was successfully added to the system
- OMITTED – product was omitted

Errors
In case of an error, the XML response includes an error message. In case of incorrect xmlData, a SOAP fault will be returned.

Example error messages:
- Invalid XML. Number of products in one XML should be between 1-50.
- Invalid XML. Product {ProductID} found twice in XML.
- Product {ProductID} already exists. Product omitted.
- Internal error. Product {ProductID}. Product omitted.
- Product {ProductID} not found in database. Product omitted.

Example – xmlData

```xml
<?xml version='1.0' encoding='UTF-8' standalone='yes'?>
<NewProduct>
 <Product>
  <ProductID>P0050</ProductID>
  <VendorID>V1001</VendorID>
  <ProductName>Office calculator Citizen SDC-812BN</ProductName>
  <ProductNumber>SDC-812BN</ProductNumber>
  <NetWeight>150</NetWeight>
  <GrossWeight>200</GrossWeight>
  <ProductDimensions>
   <Height>30</Height>
   <Width>150</Width>
   <Length>120</Length>
  </ProductDimensions>
  <EAN>4562195133322</EAN>
  <OriginCountry>CN</OriginCountry>
  <HSCode>847010</HSCode>
 </Product>
 <Product>
  <ProductID>P0051</ProductID>
  <VendorID>V1002</VendorID>
  <ProductName>Smartwatch Samsung Gear S3 Classic</ProductName>
  <GrossWeight>360</GrossWeight>
  <ProductDimensions>
   <Height>80</Height>
```

```xml
      <Width>100</Width>
      <Length>100</Length>
    </ProductDimensions>
    <EAN>887276184326</EAN>
    <OriginCountry>KR</OriginCountry>
    <HSCode>95030030</HSCode>
  </Product>
  <Product>
    <ProductID>P0031</ProductID>
    <VendorID>V1003</VendorID>
    <ProductName>Smartwatch S124</ProductName>
    <GrossWeight>360</GrossWeight>
    <ProductDimensions>
      <Height>80</Height>
      <Width>100</Width>
      <Length>100</Length>
    </ProductDimensions>
    <EAN>112233445566</EAN>
    <OriginCountry>CN</OriginCountry>
    <HSCode>95030030</HSCode>
  </Product>
</NewProduct>
```

Example – XML response (error)

```xml
<NewProductResponse>
  <PasswordExpirationTime>21</PasswordExpirationTime>
  <Product>
    <ProductID>P0050</ProductID>
    <Status>ADDED</Status>
  </Product>
  <Product>
    <ProductID>P0051</ProductID>
    <Status>ADDED</Status>
  </Product>
  <Product>
    <ProductID>P0031</ProductID>
    <Status>OMITTED</Status>
    <Error>
      <ErrorCode>32.94</ErrorCode>
      <ErrorMsg>Product P0031 already exists. Product omitted.</ErrorMsg>
    </Error>
  </Product>
</NewProductResponse>
```

RETRIEVE PRODUCT DATA

Request name: 'getProduct'. This function retrieves product data from Internel's system. The maximum number of products per request is 50.

Data input

The input 'getProduct' requires a generalDataType object with the following parameters:

Parameter	Description	Type	Required	Example
authLogin	Username	String	Yes	testuser
authPassword	Password	String	Yes	testpassword
version	API Version	String(X.Y)	Yes	1.4
xmlData	Data of products	String	Yes	<?xml...

The function xmlData is a string with data in XML format; its main element must be named GetProduct. If GetProduct is empty, a list of all products with ID and ProductName will be returned. For more specific data, the ProductID is required. The maximum number of retrievable products per request is 50.

Element	Description	Type	Required	Example
Product	Max 50 per request		No	
> ProductID	Product reference number	Str(20)	Yes	P0001

Data output

As output, a string with XML data is returned (XML response), whose main element is named GetProductResponse. XML response with a simple product list:

Element	Description	Type	Compulsory	Example
PasswordExpirationTime	Password expiration period (days)	Num(2)	Yes	20
Product			Yes	
> ProductID	Product reference number	Str(20)	Yes	P0001
> ProductName		Str(100)	Yes	Office calculator Citizen SDC-812BN

XML response with detailed product list:

Element	Description	Type	Compulsory	Example
PasswordExpirationTime	Password expiration period (days)	Num(2)	Yes	20

Product			Yes	
> ProductID	Product reference number	Str(20)	Yes	P0001
> ProductData			Yes	
>> ProductName		Str(100)	Yes	Office calculator Citizen SDC-812BN
>> ProductNumber	May be empty	Str(25)	Yes	SDC-812BN
>> VendorID		Str(20)	Yes	V1001
>> EAN		Str(30)	Yes	123412341234
>> NetWeight	May be empty	Num(8)	Yes	50
>> GrossWeight		Num(8)	Yes	60
>> HSCode	May be empty	Str(20)	Yes	871203
>> OriginCountry	May be empty	Str(2)	Yes	CN
>> ProductDimensions	May be empty		Yes	
>>> Height		Num(4)	No	50
>>> Width		Num(4)	No	60
>>> Length		Num(4)	No	70

Errors

In case of an error, the XML response contains an error instead of the ProductData element. In case of incorrect xmlData, a SOAP fault will be returned.

Example error messages:

- Invalid XML. Number of products in one XML should be between 1-50.
- Invalid XML. Product {ProductID} found twice in XML.
- Product {ProductID} not found in database.

Example – xmlData (listing all products)

```
<?xml version='1.0' encoding='UTF-8' standalone='yes'?>
<GetProduct>
</GetProduct>
```

Example – xmlData (specific product data)

```
<?xml version='1.0' encoding='UTF-8' standalone='yes'?>
<GetProduct>
 <Product>
  <ProductID>P0001</ProductID>
 </Product>
 <Product>
  <ProductID>P0002</ProductID>
 </Product>
</GetProduct>
```

Example – XML response (listing all products)

```
<GetProductResponse>
 <PasswordExpirationTime>20</PasswordExpirationTime>
 <Product>
  <ProductID>P0051</ProductID>
  <ProductName>Smartwatch Samsung Gear S3 Classic</ProductName>
```

```
 </Product>
 <Product>
  <ProductID>P0050</ProductID>
  <ProductName>Office calculator Citizen SDC-812BN</ProductName>
 </Product>
</ GetProductResponse>
```


Example – XML response (specific product data, error)


```
<GetProductResponse>
 <PasswordExpirationTime>20</PasswordExpirationTime>
 <Product>
  <ProductID>P0050</ProductID>
  <ProductData>
   <ProductName>Office calculator Citizen SDC-812BN</ProductName>
   <ProductNumber>SDC-812BN</ProductNumber>
   <VendorID>V1001</VendorID>
   <EAN>4562195133322</EAN>
   <NetWeight>150</NetWeight>
   <GrossWeight>200</GrossWeight>
   <HSCode>847010</HSCode>
   <OriginCountry>CN</OriginCountry>
   <ProductDimensions>
    <Height>30</Height>
    <Width>150</Width>
    <Length>120</Length>
   </ProductDimensions>
  </ProductData>
 </Product>
 <Product>
  <ProductID>P0051</ProductID>
  <ProductData>
   <ProductName>Smartwatch Samsung Gear S3 Classic</ProductName>
   <ProductNumber/>
   <VendorID>V1002</VendorID>
   <EAN>887276184326</EAN>
   <NetWeight/>
   <GrossWeight>360</GrossWeight>
   <HSCode>95030030</HSCode>
   <OriginCountry>KR</OriginCountry>
   <ProductDimensions>
    <Height>80</Height>
    <Width>100</Width>
    <Length>100</Length>
   </ProductDimensions>
  </ProductData>
 </Product>
 <Product>
  <ProductID>P0052</ProductID>
  <Error>
   <ErrorCode>32.204</ErrorCode>
   <ErrorMsg>Product P0052 not found in database.</ErrorMsg>
  </Error>
 </Product>
</GetProductResponse>
```

UPDATE PRODUCT DATA

Request name: 'updateProduct'. This functions allows the update of products. The maximum number of products per request is 50. ProductNumber, EAN and HSCode cannot be changed via API.

Data input

The input 'updateProduct' requires a generalDataType object with parameters as follows:

Parameter	Description	Type	Required	Example
authLogin	Username	String	Yes	Testuser
authPassword	Password	String	Yes	Testpassword
version	API Version	String(X.Y)	Yes	1.4
xmlData	Data of products	String	Yes	<?xml...

The function xmlData is a string with data in XML format; its main element must be named updateProduct with the following parameters:

Element	Description	Type	Required	Example
Product	Max 50 per request		Yes	
> ProductID	Product reference number	Str(20)	Yes	P0050
> ProductName	Name of product	Str(100)	No	Calculator Citizen SDC-812BN
> VendorID	Vendor reference number	Str(20)	No	V0002
> NetWeight	Netto Weight (in grams)	Num(8)	No	140
> GrossWeight	Gross Weight (in grams)	Num(8)	No	190
> ProductDimensions			No	
>> Height	In millimetres	Num(4)	Yes	28
>> Width	In millimetres	Num(4)	Yes	148
>> Length	In millimetres	Num(4)	Yes	118
> OriginCountry	May be empty*, ISO 3166-1	Str(2)	No	LT

* OriginCountry; a value may be left away/empty in case of national or EU sales, whereas for exports outside of the EU this is a mandatory input.

Data output

As output, a string with XML data (XML response) is returned, its main element is named UpdateProductResponse and contains the following elements:

Parameter	Description	Type	Compulsory	Example
PasswordExpirationTime	Password expiration period (days)	Num(2)	Yes	20
Product			Yes	
> ProductID	Product reference number	Str(20)	Yes	P0001
> Status	Status of product update	Enum	Yes	UPDATED

Possible product statuses are:

- UPDATED – product data was successfully updated
- OMITTED – product update was omitted

Errors

In case of an error, the XML response contains an error message. In case of incorrect xmlData, a SOAP fault will be returned.

Example error messages:

- Invalid XML. Number of products in one XML should be between 1-50.
- Invalid XML. Product {ProductID} found twice in XML.
- Internal error. Product {ProductID}. Product omitted.
- Product {ProductID} not found in database. Product omitted.

Example – xmlData


```xml
<?xml version='1.0' encoding='UTF-8' standalone='yes'?>
<UpdateProduct>
  <Product>
    <ProductID>P0050</ProductID>
    <VendorID>V1002</VendorID>
    <ProductName>Calculator Citizen SDC-812BN</ProductName>
    <NetWeight>140</NetWeight>
    <GrossWeight>190</GrossWeight>
    <ProductDimensions>
      <Height>28</Height>
      <Width>148</Width>
      <Length>118</Length>
    </ProductDimensions>
    <OriginCountry>LT</OriginCountry>
  </Product>
  <Product>
    <ProductID>P0051</ProductID>
    <VendorID>V1042</VendorID>
  </Product>
  <Product>
    <ProductID>P0052</ProductID>
    <VendorID>V1042</VendorID>
  </Product>
</UpdateProduct>
```


Example – XML response (error)


```xml
<UpdateProductResponse>
  <PasswordExpirationTime>20</PasswordExpirationTime>
  <Product>
    <ProductID>P0050</ProductID>
    <Status>UPDATED</Status>
  </Product>
  <Product>
    <ProductID>P0051</ProductID>
    <Status>OMITTED</Status>
```

```
    <Error>
      <ErrorCode>32.264</ErrorCode>
      <ErrorMsg>Vendor V1042 not found in database. Product omitted.</ErrorMsg>
    </Error>
  </Product>
  <Product>
    <ProductID>P0052</ProductID>
    <Status>OMITTED</Status>
    <Error>
      <ErrorCode>32.247</ErrorCode>
      <ErrorMsg>Product P0052 not found in database.</ErrorMsg>
    </Error>
  </Product>
</UpdateProductResponse>
```

RETRIEVE STOCK DATA

Request name: ‘getStockReport’. This function retrieves stock data from Internel’s system. The maximum number of products per request is 50.

Data input

The input ‘getStockReport’ requires a generalDataType object with parameters as follows:

Parameter	Description	Type	Required	Example
authLogin	Username	String	Yes	Testuser
authPassword	Password	String	Yes	Testpassword
version	API Version	String(X.Y)	Yes	1.4
xmlData	Data of products	String	Yes	<?xml...

The function xmlData is a string with data in XML format; its main element must be named GetStockReport. If GetStockReport is empty, a list of all products with stock data will be returned. The maximum number of retrievable products per request is 50.

Element	Description	Type	Required	Example
Product	Max 50 per request		No	
> ProductID	Product reference number	Str(20)	Yes	P0050

Data output

As output, a string with XML data (XML response) is returned, whose main element is named GetStockReportResponse and contains the following elements:

Parameter	Description	Type	Compulsory	Example
PasswordExpirationTime	Password expiration period (days)	Num(2)	Yes	20
Product			Yes	
> ProductID	Product reference number	Str(20)	Yes	P0001
> ProductQuantity			Yes	
>> AvailableFree	Quantity of available items	Numeric(10)	Yes	150
>> Damaged	Quantity of damaged items	Numeric(10)	Yes	0
>> Reserved	Quantity of reserved items	Numeric(10)	Yes	17
>> Returned	Quantity of returned items	Numeric(10)	Yes	2

Errors

In case of an error, the XML response contains an error message. In case of incorrect xmlData, a SOAP fault will be returned.

Example error messages:

- Invalid XML. Maximum number of products in one XML is 50.
- Invalid XML. Product {ProductID} found twice in XML.
- Product {ProductID} not found in database. Product omitted.

Example – xmlData

```xml
<?xml version='1.0' encoding='UTF-8' standalone='yes'?>
<GetStockReport>
 <Product>
  <ProductID>P0050</ProductID>
 </Product>
 <Product>
  <ProductID>P0051</ProductID>
 </Product>
 <Product>
  <ProductID>P0052</ProductID>
 </Product>
</GetStockReport >
```

Example – XML response (error)

```xml
<GetStockReportResponse>
 <PasswordExpirationTime>20</PasswordExpirationTime>
 <Product>
  <ProductID>P0050</ProductID>
  <ProductQuantity>
   <AvailableFree>150</AvailableFree>
   <Damaged>0</Damaged>
   <Reserved>17</Reserved>
   <Returned>2</Returned>
  </ProductQuantity>
 </Product>
 <Product>
  <ProductID>P0051</ProductID>
  <ProductQuantity>
   <AvailableFree>34</AvailableFree>
   <Damaged>2</Damaged>
   <Reserved>3</Reserved>
   <Returned>0</Returned>
  </ProductQuantity>
 </Product>
 <Product>
  <ProductID>P0052</ProductID>
  <Status>OMITTED</Status>
  <Error>
   <ErrorCode>1.2:39.139</ErrorCode>
   <ErrorMsg>Product P0052 not found in database.</ErrorMsg>
  </Error>
 </Product>
</GetStockReportResponse>
```

DOCUMENT CHANGE SUMMARY

Version	Changes	Release date
1.4	- Changed NewOrder input data requirements	02.01.2020
1.2	- Added function: getStockReport - Changed function: orderStatus – added tracking information and ORDER_RETURNED status - Changed function: newOrder – data input structure changed - Changed function: newVendor – data input structure changed	17.10.2018
1.1	- Changed function: newOrder – data input structure changed - Changed function: updateProduct – data input structure changed	15.08.2018
1.0	Base version	09.04.2017

Code sample (PHP)

To use below code samples in PHP, please download the PHP SOAP Client (InternelAPISoapClient.php file) from provided API server url.

Test connection

```php
<?php
require_once 'InternelAPISoapClient.php';

//create the client from included library
$cli = new InternelAPISoapClient();

//send request and show the result
var_dump($cli->testConnection('Mark'));
```

Test credentials

```php
<?php
require_once 'InternelAPISoapClient.php';

//create the client from included library
$cli = new InternelAPISoapClient();

//Login/version data
$login = 'yourlogin';
$password = 'yourpassword';
$version = '1.4';

//function testCredentials requires testCredentialsDataType (described in the InternelAPISoapClient)
$credentials = new testCredentialsDataType();
$credentials->authLogin = $login;
$credentials->authPassword = $password;
$credentials->version = $version;

//send testCredentials request
$resp = $cli->testCredentials($credentials);

//create XML object from the response
$xml = new SimpleXMLElement($resp);

//process the response (with error handling)
if (isset($xml->Error)) {
   //handle error
   echo "Error code: {$xml->Error->ErrorCode}\n";
   echo "Error message: {$xml->Error->ErrorMsg}\n";
} else {
   echo "Response message: {$xml->Success}\n";
   echo "Your password will expire in: {$xml->PasswordExpirationTime}\n";
}
```

Create new vendor

```php
<?php
require_once 'InternelAPISoapClient.php';
```

```
//create the client from included library
$cli = new InternelAPISoapClient();

//Login/version data
$login = 'yourlogin';
$password = 'yourpassword';
$version = '1.4';

//function newVendor requires generalDataType
$generalDataType = new generalDataType();
$generalDataType->authLogin = $login;
$generalDataType->authPassword = $password;
$generalDataType->version = $version;

//creating xmlData with new vendor(s) data:
$newVendor = new SimpleXMLElement('<NewVendor/>');
//creating Vendor 1
$vendor = $newVendor->addChild('Vendor');
$vendor->addChild('VendorID', 'V01');
$vendor->addChild('VendorName', 'New Test Vendor 1');
$vendor->addChild('Country', 'IT');
$vendor->addChild('City', 'Rome');
$vendor->addChild('ZipCode', '12345');
$vendor->addChild('Street', 'Test Street');
$vendor->addChild('HouseNumber', '12');
//creating Vendor 2
$vendor2 = $newVendor->addChild('Vendor');
$vendor2->addChild('VendorID', 'V02');
$vendor2->addChild('VendorName', 'New Test Vendor 2');
$vendor2->addChild('Country', 'IT');
$vendor2->addChild('City', 'Rome');
$vendor2->addChild('ZipCode', '54321');
$vendor2->addChild('Street', 'Test Street');
$vendor2->addChild('HouseNumber', '52');
//xmlData is a string, containing XML data
$generalDataType->xmlData = $newVendor->asXML();

//send newVendor request
$resp = $cli->newVendor($generalDataType);

//processing response with error handling
$xml = new SimpleXMLElement($resp);
if (isset($xml->Error)) {
    echo "Error code: {$xml->Error->ErrorCode}\n";
    echo "Error message: {$xml->Error->ErrorMsg}\n";
} else {
    echo "Your password will expire in: {$xml->PasswordExpirationTime}\n";
    foreach ($xml->Vendor as $vendor) {
        echo "VendorID: {$vendor->VendorID}\n";
        echo "Status: {$vendor->Status}\n";
        if (isset($vendor->Error)) {
            echo "Error code: {$vendor->Error->ErrorCode}\n";
            echo "Error message: {$vendor->Error->ErrorMsg}\n";
        }
    }
}
```