# ALERTS MANAGEMENT API

Documentation

**ver. 2.31**


Národní organizace pro ověřování pravosti léčiv, z.s., IBC Pobřežní 620/3, Karlín, 186 00 Praha 8
IČ: 05851742, www.czmvo.cz

# Content

# 1. Introduction

API CZMVO.CZ is simple REST API, which allows integration of MAH's or end users' alert management system with NOOL alert management system. Benefit of the integration is automation of processes and tasks related to alerts investigation for the both parties MAHs and end users.

User without own alert management system or users unable to connect via API due to company policies can use web interface of alert management system. Web interface has the same functionalities as API. Description of alert management system web interface is in the separated user manuals different for MAHs and end users. There are also more detail description of alert system logic and support for communication between MAH and end user during investigation.

## 1.1. Access to the API and web interface

List of access points is in the table bellow.

| Environment | Web interface | API |
|---|---|---|
| Production | https://portal.czmvo.cz/ | https://api.czmvo.cz/ |
| Test | https://sandbox.czmvo.cz/ | https://api.czmvo.cz/t/ |

Data from the production dataset are copied to the development database nightly. Tests can be than performed with the real user's data. Changes made in test environment has no effect to the data in production system, so they can be changed as needed. Changes performed in the test environment are overwritten each night, this fact needs to be considered during testing.

Access credentials type:

a) Login name and password which can be generated upon request at email address registrace@czmvo.cz
b) End users (e.g. pharmacies or distributors) can get status of single alert via Alert ID used in login and Location ID as password.
c) Development account for end user IT software suppliers. The account allows access to development environment only (to the both API and web portal). They can select end user for which are developing software once logged in into system (if end user is not selected, any alert will not be visible). Selection of end-user can be performed any time in web portal via select box beside language selection.

**Access credentials allows access to the both production and test systems.**

## 1.2. Email notifications

System sends email notification when new message/document is inserted. Sending notifications is configurable in settings.
Notification format is plain text. Each notification contains **footer** with data for easier machine processing. Format is following:

**EVENT:NEWMESSAGE*ID:20*AUDIT:FALSE**

Individual fields may be subject of change in the future but format of the message will keep the structure:
**VARIABLE1:VALUE1*VARIABLE2:VALUE2* … *VARIABLEX:VALUEX**
Possible variables and theirs's values are following:

**EVENT** – value: currently only NEWMESSAGE
**ID** – value: message ID for the new message (EVENT: NEWMESSAGE)
**AUDIT** – Boolean type (TRUE if audit trail record from end user system is requested, otherwise FALSE)

# 2. API description

## 2.1. General

It is equivalent in terms of function to a web interface. It is a REST API with basic authorization. The data exchange utilizes the JSON file format (if not specified otherwise).

A simple sample code of a request in PHP and CURL: (will be similar for other languages).

```
$ch = curl_init();
curl_setopt($ch, CURLOPT_URL, URL));
curl_setopt($ch, CURLOPT_RETURNTRANSFER, true);
curl_setopt($ch, CURLOPT_USERPWD, „LOGIN:PASSWORD");
curl_setopt($ch, CURLOPT_HTTPAUTH, CURLAUTH_BASIC);
curl_setopt($ch, CURLOPT_CUSTOMREQUEST, http_METHOD);
curl_setopt($ch, CURLOPT_POSTFIELDS, „REQUEST" );
curl_setopt($ch, CURLOPT_HTTPHEADER, array(,Content-
Type:application/json'));
$result = curl_exec($ch);
```

Where:

- URL is the address of a function
- LOGIN, PASSWORD are login credentials
  Login credentials are either generated by the system upon a request, or for access to a specific alert (for end users only) it is possible to use an „ID alert"– UPRC as the login and the location ID as the password.
- http_METHOD is one of the methods „GET", „POST", „DELETE", „PUT"

- REQUEST – JSON formatted request. Can also be generated in the web interface.

The output is JSON, the format differs in used function or method. The basic structure is as follows:

```
{„status“:“ok“,“code“:0,“message“:“OK“,“result“:{ … result …}}
```

If *code* is different from zero, then *message* contains the description of the error. The field *status* is either „ok“ – request was processed or „error“ – request was not processed (that usually means– if there is no internal API error, that *code* – number of the error – is not zero).

All methods have their equivalents in the web portal. Everything valid in the web portal is also valid in the direct use of API. We therefore recommend obtaining additional information from the “Web portal” section.

### 2.1.1 Setting-up the language of the answer

Some values of answers can be localized – e.g. API error messages, alert states in the code list. Http header Accept-Language can be used to set up the desirable language.
Supported languages are currently Czech (cs) and English (en).

Examples:

```
Accept-Language: en
```

or

```
Accept-Language: cs-CZ
```

## 2.2. Parameter passing for GET and DELETE methods

Some systems and libraries don’t allow passing data in the body of request (POST data) for the GET and DELETE methods. In such case parameters need to be inserted directly into url. Parameters in url must correspond with parameters from the requested JSON query.

Example:
JSON request “Get all messages since 6 Aug 2019 12:00:00”
{"resultAs":"json","list":"messages,"changedFrom":"2019-08-06 12:00:00"}

Passing parameters via url (test environment) looks like:
https://portal.czmvo.cz/alerts/?resultAs=json&list=messages&changedFrom=2019-08-06+12%3A00%3A00

## 2.3. Type of return values

Request parameter “resultAs” can be replaced equivalent http header Accept.
For "resultAs": "json":
Accept: application/json

For "resultAs": "csv":
Accept: text/csv

## 2.4. Overview of error states –„code“ field

| code (error code) | http response | Description |
|---|---|---|
| **0** | 200 | Request was correctly processed. |
| **1** | 404 | Unknown function.<br>*Most likely an incorrect URL of a request.* |
| **2** | 401 | Unauthorized access.<br>*User authentication was not successfully completed, the user cannot be logged in.* |
| **3** | 401 | Function not allowed.<br>*Most likely an incorrect URL of a request, or the user does not have authorization to access the API.* |
| **4** | 405 | Forbidden calling method.<br>*Unknown http method (allowed methods are GET, POST, PUT, DELETE).* |
| **5** | 400 | Forbidden value of a parameter.<br>*Also returns the specific parameter.* |
| **11** | 400 | Parameter value not filled in.<br>*Most likely the mandatory parameter is not filled in.* |
| **12** | 404 | Alert not found. |
| **13** | 405 | No authorization to write into and alert. |
| **14** | 400 | The file could not be decoded.<br>Incorrectly coded file in JSON request. |
| **15** | 400 | File size exceeded NMB<br>*Returns a current maximum file size (16MB).* |
| **16** | 500 | Message could not be saved.<br>*Internal API error.* |
| **17** | 401 | Authorization to edit the message not granted. |
| **18** | 401 | Message cannot be answered.<br>*Either it no longer exists, or it is closed.* |
| **19** | 401 | Message cannot be deleted.<br>*Message has an attached response and cannot be deleted.* |
| **20** | 400 | At least one parameter has to be entered UPRC, ID |
| **21** | 404 | File ID %s not found |
| **22** | 401 | No authorization to read this file. |
| **23** | 415 | File type not supported. |

| | | |
|---|---|---|
| | | Supported file types are *txt,pdf,csv,jpg,png,tiff*. |

## 2.5. Function Alerts

Function url: https://api.czmvo.cz/alerts/
Test environment function url: https://api.czmvo.cz/t/alerts/

Generation of specific queries and display of API answers in JSON format is also available in the web interface of the API system.

### 2.5.1 GET method

Is the equivalent to the operation „Insert data“ in the web portal.

#### Get alert states

Example of the request (detection of alert states):

```
{"resultAs":"json","list":"state","uprc":"CZ-0VG-ZZW-5BU-
LZ0","latest":true,"audit":true,"createdFrom":"2019-08-06
00:00:00","createdTo":"2019-08-13 00:00:00","changedFrom":"2019-08-04
00:00:00","state":1,"page":1}
```

Example of the answer

```
{"status":"ok","code":0,"message":"OK","result":{"alerts":[{"uprc":"CZ-0VR-
Y94-KK5-6FJ","created":"2019-07-16
07:50:04","productcode":"08595116521485","stateid":"1","state":"Nov\u00fd",
"lastmessageid":"20","audit":false}]}}
```

Example of the request (detection of number of pages in a subset):

```
{"resultAs":"json","list":"state","page":-1}
```

Example of the answer:

```
{"status":"ok","code":0,"message":"OK","result":{"pages":3,"currentPage":0}
}
```

*Notes:*
If a parameter is not mandatory, it does not have to be a part of the request
**resultAs** can have values „*json*“ or „*csv*“. If set to csv, then the output file is a csv table.
**uprc** – unique alert identifier
**list** – always „*state*“.
**latest** – *true/false* – sorted by the newest alert.
**createdFrom** – displays alerts newer than specified time only. All time values have to follow format „YYYY-MM-DD HH:MM:SS“.
**createdTo** – displays alerts older than specified time only.
**changedFrom** – displays only alerts that have been changed after the specified time.
**state** – displays alerts with desired state (according to the alert state code list),
**page** – number – determines what page (subset) is displayed in the answer. Maximum number of alerts (which is one page) in one subset is 500 (depending on specific authorization settings,

the number of alerts in any subset can be lower). If the *page* parameter is set to a negative number (e.g. -1), then the total number of subset pages will be returned in the answer.
In JSON answers the values **pages** and **currentPage** are always returned.

In the answer (apart from obvious fields):
**alerts** is always a field – even if the result is only one entry
**stateid** – alert state ID
**state** – state ID in plain text
**lastmessageid** – last entered message.
**audit** – *true/false* – indicator, the request for audit trail is registered with alerts – if there is a request (value is true), then to resolve the request:

1) First to locate the message ID, which is the request (see examples in section „Insert message“),
2) Then create a new message with identifier „audit“ as an answer to this request and attach a file containing the audit trail record.

**pages** – number of pages of entries (subsets)
**currentPage** – current returned page of results in the alerts field (if the **page** number is positive, then this value is returned)

### Get messages

Example of a request:
Detect all messages since 6.8.2019 12:00:00

```
{"resultAs":"json","list":"messages,"changedFrom":"2019-08-06 12:00:00"}
```

Detect all messages regarding alert CZ-0VG-ZZW-5BU-LZP

```
{"resultAs":"json","list":"messages","uprc":"CZ-0VG-ZZW-5BU-LZP"}
```

Detect/Load message with ID 20

```
{"resultAs":"json","list":"messages","id":"20"}
```

Detect a message with audit trail record request

```
{"resultAs":"json","list":"messages","uprc":"CZ-0VG-ZZW-5BU-
LZP","audit":true}
```

Example answer:

```
{"status":"ok","code":0,"message":"OK","result":{"messages":[{"id":"19","pa
rent":"0","uprc":"CZ-0VR-Y94-KK5-6FJ","created":"2019-08-06
10:45:59","changed":"2019-08-06  10:45:59","subject":"info","message":"Uplne
ok","isfile":false,"audit":false,"public":false,"fromme":true},{"id":"20","
parent":"19","uprc":"CZ-0VR-Y94-KK5-6FJ","created":"2019-08-06
10:59:06","changed":"2019-08-06                    10:59:06","subject":"Re:
```

info","message":"Fajn","isfile":false,"audit":false,"public":true,"fromme":false}]}}

*Notes:*
**uprc** – unique alert identifier, it is a mandatory parameter in case that parameter **id** or **changedFrom** is not used
**list** – always „*messages*“
**id** – message ID
**changedFrom** – time in format „YYYY-MM-DD HH:MM:SS“ – returns messages newer than specified time. If neither **uprc** nor **id** is entered, then the **changedFrom** value should not be older than 1 month.
**audit** – true/false – returns messages that have audit requirement.

In the answer: (apart from obvious fields)
**messages** – is always a field – even if the result is only one entry
**id** – message ID.
**parent** – message ID, that is being responded to (requirement/request ID).
**uprc** – unique alert identifier.
**created** – message creation time.
**changed** – time of the last change.
**subject** – message subject.
**message** – message body.
**isfile** – *true/false* – message contains a file
**audit** – *true/false* – message is an audit requirement (for step by step solution see section Reading the alert states).
**fromme** – true/false – identifier, that signifies if the message was created by the current user

## Get file

Example request:

```
{"resultAs":"json","list":"file","id":"21"}
```

or (direct file export)

```
{"resultAs":"csv","list":"file","id":"21"}
```

Example answer:

{"status":"ok","code":0,"message":"OK","result":{"filename":"xxx.pdf","filedata":"JVBERi0xLjcKCjQgMCBvYmoKKElkZW50aXR5KQplbmRvYmoKNSAwIG9iagooQWRvYmUpCmVuZG9iago4IDAgb2JqCjw8Ci9GaWx0ZXIgL0ZsYXRlRGVjb2RlCi9MZW5ndGggODk1NjAKL1R5cGUgL1N0cmVhbQo+PgpzdHJlYW0KeJzsfQlgVMX9/em80mu5vNRUIOkpADAtlwiVIkAmKiRsOlYD2CcqkV8ATBA6sith7xQsWDJfEIiBWPHmqtWK2irTWtt5Wf2FJqlez+vzO7yW7kaNJ\/aPnl9z678515M\/OdmTfzne+b77y3+4AAQBoSAdZPmHH8cYYfjn4MhDMaAbx\/OG7CxElb6zruAe63XQB8xXEN02acdCDjb8C97QCSPu+4GSeP+\/zOs\/4E\/ …

*Notes:*

**resultAs** – *json* – returns a file in a format that is indicated in the example above
**resultAs** – *csv* – directly exports a file (with the corresponding mime type).
**list** – always „file“.

### Get alert state code list

Example of a request:

```
{"resultAs":"csv","list":"enumState"}
```

or (direct file sending)

```
{"resultAs":"csv","list":"file","id":"21"}
```

Example of an answer:

```
{"status":"ok","code":0,"message":"OK","result":{"states":[{"id":"1","name"
:"Nov\u00fd","externalcode":"01","finalstate":false},{"id":"5","name":"V
\u0159e\u0161en\u00ed","externalcode":"#","finalstate":false},{"id":"3","na
me":"Uzav\u0159en\u00fd","externalcode":"06a,06b,06c","finalstate":true},{"
id":"6","name":"Odlo\u017een\u00fd","externalcode":"","finalstate":false},{
"id":"7","name":"Chyba                    import                    na
callcentrum","externalcode":"CALLFAIL","finalstate":false}]}}
```

*Notes:*
**resultAs** - may take values *„json“* or *„csv“* – If value csv is set, the output will be table in csv format.
**states** – always a field.
**name** – name of the state.
**externalcode** – code according alert state code list.
**finalstate** – *true/false* – indicates whether the state is finite (i.e. whether the alert is resolved) or not.
**settingallowed** – *true/false* – indicates whether the state of the alert can be set through an API (only for MAH’s) or standard way.

### Get group of alerts

Only for MAH’s.
Allows loading of a list of alerts that are in the same group as the default alert. The system group alerts with respect to their similarity.

Example of a request:

```
{"resultAs":"json","list":"group","uprc":"CZ-0VR-YE5-C1N-KLM"}
```

Example of an answer:

```
{"status":"ok","code":0,"message":"OK","result":{"uprc":["CZ-0VR-YE5-C1N-
KLM","CZ-0VR-YE5-VS7-BXP"]}}
```


Národní organizace pro ověřování pravosti léčiv, z.s., IBC Pobřežní 620/3, Karlín, 186 00 Praha 8
IČ: 05851742, www.czmvo.cz

*Notes:*
**uprc** – in query – identifier of the default (one) alert

In answer:
**uprc** - list of all alerts that belong to the same group. If an alert does not belong to any group, the field in the answer is empty.

### 2.5.2 POST method

It is equivalent to „Insert data“ operation from the web portal. Meaning of the fields is described there as well.

Request (simple insert):

```
{"uprc":"CZ-0VR-Y94-KK5-
6FJ","public":true,"subject":"test","message":"test"}
```

Request (insert as an answer to another message):

```
{"uprc":"CZ-0VR-Y94-KK5-6FJ","public":true,"id_parent":20,"subject":"Re:
Re: info","message":"test"}
```

Request (insertion of a file and an audit trail – if it is just a simple file, parameter audit takes value *false* or is not specified):

```
{"uprc":"CZ-0VR-Y94-KK5-6FJ","public":false,"audit":true,"subject":"Re: Re:
info","message":"test","file":"iVBORw0KGgoAAAANSUhEUgAAABAAAAAQCAIAAACQkWg2
AAAACXBIWXMAAAsTAAALEwEAmpwYAAAAIGNIUk0AAHolAACAgwAA+f8AAIDpAAB1MAAA6mAAADq
YAAAXb5JfxUYAAADjSURBVHjaYnwaJcSABBhZ2RlQwf\/fP5G5THiUwsWRpZiQVePSgyzFhF8dJ
mBB43PahAmkTEAWeRYtDLf\/\/++fjE+jhCB88cmXmXiFcRn8PEEK6iRGVnZOmzDJBc9eZKihGf
w8QQquTnLBM4Qfvh9Z9WFOgdTSt8+iETZILX0LCVBkQUQo4dGDFiQIT38\/sgqi6FmkgNTyD3A9
EMaHOQVYQgmqZ\/mHZ9HCcKUQ1RApBgYGFohD4fYi7IHpQVb9\/\/dPFswQhOtBVooz4pD1oKmG
ugU5teJJI\/A0y4QnJcMFkcUBAwCuU3b1BVKmxQAAAABJRU5ErkJggg==","filename":"test
.png"}
```

Answer:

```
{"status":"ok","code":0,"message":"OK","result":{"id":22}}
```

*Notes:*
**uprc** – alert identifier. Mandatory if the *id_parent* field is not used.
**public** – *true/false* – indicates that the answer is public in a sense that both MAH and end-user has access to it (if value is set to false, only author and system administrator have access to the message)
**id_parent** – message ID which is being replied to.
**subject** – mandatory – message subject
**message** – message text. Mandatory if file is not attached.
**file** – base64 encoded binary file (of an unrestricted type).
**filename** – name of the file (mandatory if file is attached).

**audit** – *true/false* - indicates whether the request resolves an audit trail request. ID provided in the answer is equal to message ID assigned to message being created.

### 2.5.3 PUT method

It is equivalent to „Edit data“ operation from the web portal.

#### *2.5.3.1 Edit existing message*

Allows editing of *public, subject* and *message* fields in an already existing message.

Note: Author only is allowed edit data. Messages for which already exist answers can't be edited.

Example of a request:

```
{"id":22,"public":true,"subject":"TEST"}
```

Example of an answer:

```
{"status":"ok","code":0,"message":"OK","result":{"id":22,"changed":"2019-
08-06 15:36:35"}}
```

*Notes:*
**id** – message ID
only *public*, *subject* and *message* fields can be edited.

#### *2.5.3.2 Edit alert status*

Allows state change of single alert or group of alerts.

Request: (change state of single alert)
```
{"uprc":"CZ-0VR-YE5-VS7-BXP","state":5,"group":false}
```

Request: (change state of group of alerts)
```
{"uprc":"CZ-0VR-YE5-VS7-BXP","state":5,"group":true}
```

Request: (alternative change state of group of alerts)
```
{"uprc":["CZ-0VR-YE5-VS7-BXP","CZ-0VR-YE5-C1N-KLM"],"state":5}
```

Request: (set alert for standard processing)
```
{"uprc":"CZ-0VR-YE5-VS7-BXP","state":-1,"group":false}
```

Answer:
```
{"status":"ok","code":0,"message":"OK","result":{"uprc":["CZ-0VR-YE5-
C1N-KLM","CZ-0VR-YE5-VS7-BXP"]}}
```

*Note:*
**uprc** – unique alert identifier - mandatory

**state** – new alert state ID. For the list of all possible states see section **Loading of dial states**. States that can be configured though API have identifier **settingallowed**. If the parameter **state** is entered as a negative number (resp. **-1**), the alert state itself is not changed, but the alert is marked in a way that enables standard processing, which happens with alerts where corresponding parties do not have access to this API.

**group** – true/false – If true, then the state is set for all alerts from the same group (see GET method, getting alerts from a group) as an input alert (parameter **uprc**). If set to false, then the change applies to the state of selected alert only.

In answer:
List of affected (changed) alerts is in the **uprc**.

### 2.5.4 DELETE method

It is equivalent to „Delete data“ operation from the web portal. Enables user to delete a message.

Request:

```
{"id":22}
```

Answer:

```
{"status":"ok","code":0,"message":"OK","result":{"id":22}}
```

Note:
Message ID is the only parameter possible. The conditions for deleting a message are identical to conditions for its editing.

## 2.6. Function Exceptions

Exceptions function is equivalent to the functionality in the web interface. It is REST API with basic authentication. Data exchange is via JSON format (if not stated otherwise).

Simple example code in PHP and CURL:

```
$ch = curl_init();
curl_setopt($ch, CURLOPT_URL, URL));
curl_setopt($ch, CURLOPT_RETURNTRANSFER, true);
curl_setopt($ch, CURLOPT_USERPWD, "LOGIN:PASSWORD");
curl_setopt($ch, CURLOPT_HTTPAUTH, CURLAUTH_BASIC);
curl_setopt($ch, CURLOPT_CUSTOMREQUEST, HTTP_METHOD);
curl_setopt($ch, CURLOPT_POSTFIELDS, "REQUEST" );
curl_setopt($ch, CURLOPT_HTTPHEADER, array('Content-Type:application/json'));
$result = curl_exec($ch);
```

Where:

- URL is function address
- LOGIN, PASSWORD are access credentials
- HTTP_METHOD is one of the methods “GET”, “POST”, “DELETE”
- REQUEST – request in JSON format. It can be generated in the web interface if needed.

Output is JSON. Format depends on the used method or function. Basic structure is following:
{"status":"ok","code":0,"message":"OK","result":{ … *výsledek* …}}

If code is different to “0” than message contains error description.

Function url https://api.czmvo.cz/filter/
Function url for the test environment https://api.czmvo.cz/t/filter/

### 2.6.1 GET method

Method is equivalent to the function “Read data” in the web interface.

**Get the state code list**
Request:
{"resultAs":"json","list":"enumState"}

Answer:
{"status":"ok","code":0,"message":"OK","result":{"states":[{"code":"NO","name":"Uzav\u0159eno - MAH - nelze opravit"},{"code":"OP","name":"Uzav\u0159eno - MAH - opraveno"}]}}

Note:
resultAs can take values “json” or “csv” – it the value is set to “csv” result is table in json format.

**Get the list of existing exceptions**
Request:
{"resultAs":"json","list":"product","productCode":"kod","batch":"","serialNumber":"","id":[1,2,3]}

Answer:
{"status":"ok","code":0,"message":"OK","result":{"products":[],"count":0}}

Note:
“id” is array id, which are assigned to filter when inserted, parameters for filtering “id”.
Parameters „productCode“, „batch“, „serialNumber“ are optional

**Verify pack for the exception**
Request:
{"resultAs":"json","list":"verify","productCode":"08594158891136","batch":"1","serialNumber":"00000045287"}

Answer:
{"status":"ok","code":0,"message":"OK","result":{"isException":true,"info":{"id":41,"productCode":"0859
4158891136","batch":null,"serialNumber":"00000045287","stateId":1,"state":"Nov\u00fd"}}}

Note:
At least one of the arrays „productCode“, „batch“, „serialNumber“ has to be filled.
In answer:
isException: true/false – product or pack falls under the exception
info – if isException is true, than the array contains exception description

### 2.6.2 POST method

Method is equivalent to the operation “Insert data” from the web interface.

Request (single item):
{"validity":"2019-04-
30","state":"OP","productCode":"0123456789","batch":"123456","serialNumber":"0123"}

Request (batch insert via csv file):
{"validity":null,"state":"NO","csv":"zCIsIsSMbMOhbmVrIDU3IGtvZGV4dSIsIklEIGRyxb5pdGVsZSByZWdpc3
RyYWNlIChNQUggSUQpIiRyxb5pdGVsZSByZWdpc3RyYWNlIChNQUgpIiwiQWRyZXNhIGRyxb5pdGVsZSByZ
Wdpc3RyYWNlIiwiTc3RvIGRyxb5pdGVsZSByZWdpc3RyYWNlIiwiUFPEjCBkcsW+aXRlbGU0cmFjZSIsIlplbcSbI
GRyxb5gaahjfkleuHGAFlgheghv4fdb2xxYVNS5fAS5gdrgpdGVsZSByZWdpc3RyYWNlIiwiw5pkYWplIG8gZGlz
dHJpYnV0b3JvdmkiDQo="}

Answer:
{"status":"ok","code":0,"message":"OK","result":{"products":[{"lineNo":1,"productCode":"0123456789","
batch":"123456","serialNumber":"0123","validity":null,"state":1,"ID":1,"errorCode":0,"errorText":""}],"co
unt":0}}

Note:
Parameter “validity” is date in format [YYYY]-[MM]-[DD]. If inserted format doesn’t match, it is considered as not entered at all.
Parameter “state” is default value for the set alert state (if state is not defined in the csv file).
Answer can contain more rows for the batch insert, where ID or reason is described for the each not inserted row.
Parameter “csv” is csv file encoded in base64 format. Structure is described in the web interface section.

### 2.6.3 DELETE method

Method is equivalent to the operation “Remove” from the web interface.

Request:
{"productCode":"0123456789","batch":"123456","serialNumber":"sn","id":[1,2]}

Answer:
{"status":"ok","code":0,"message":"OK","result":{"affected":1,"deleted":[{"id":"2","productCode":"0123456789","batch":"123456","serialNumber":"sn"}]}}

Note:
Parameters productCode, batch, serialNumber are optional, but at least one parameter has to be used (it is not possible to remove all items in single request)