API v1

REFERENCE GUIDE

Incassoservicedesk
TRIVION | EIKENDREEF 4, 2350 VOSSELAAR

Change Log

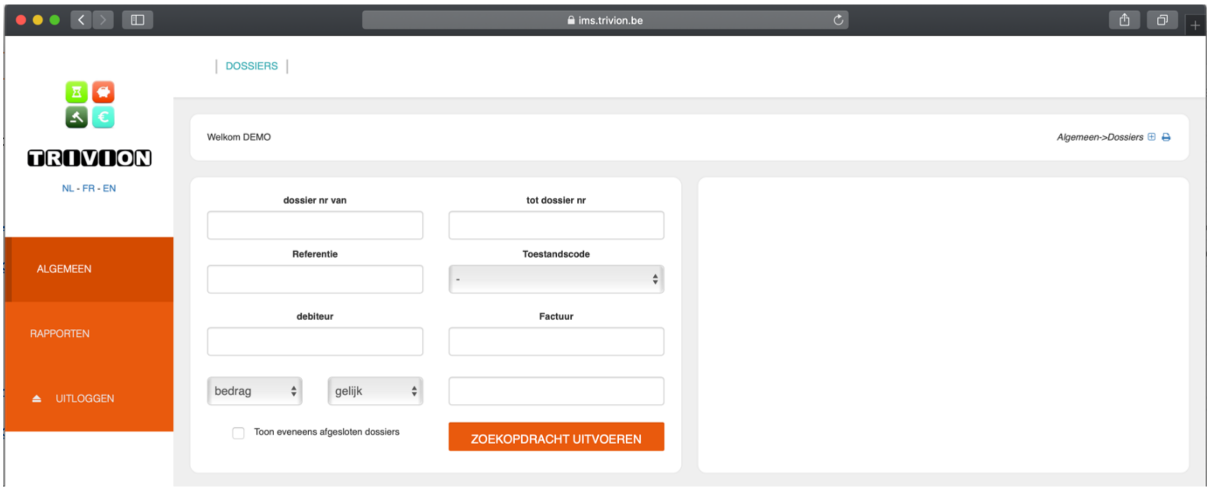
2020-01-27	Initial version
2020-01-29	Get-Token Header changed
2020-02-10	Submit-Payment added
2020-03-15	Extend Submit-Invoice with payments Extend response with status and debtor info Get-Feedback added
2020-03-16	Add password in Request body as mandatory
2020-03-24	Get-Statuscodes added Response with HTTP response code
2020-03-27	Update Get-Statuscodes
2021-10-07	Submit-Remark Update Submit-Invoice with Remarks
2023-09-01	Update Submit-Invoice: custtype and filetype
2024-02-17	Add type to Payments in Submit-Invoice and Submit-Payment
2025-03-29	Add payoutref to Submit-Invoice

Table of Contents

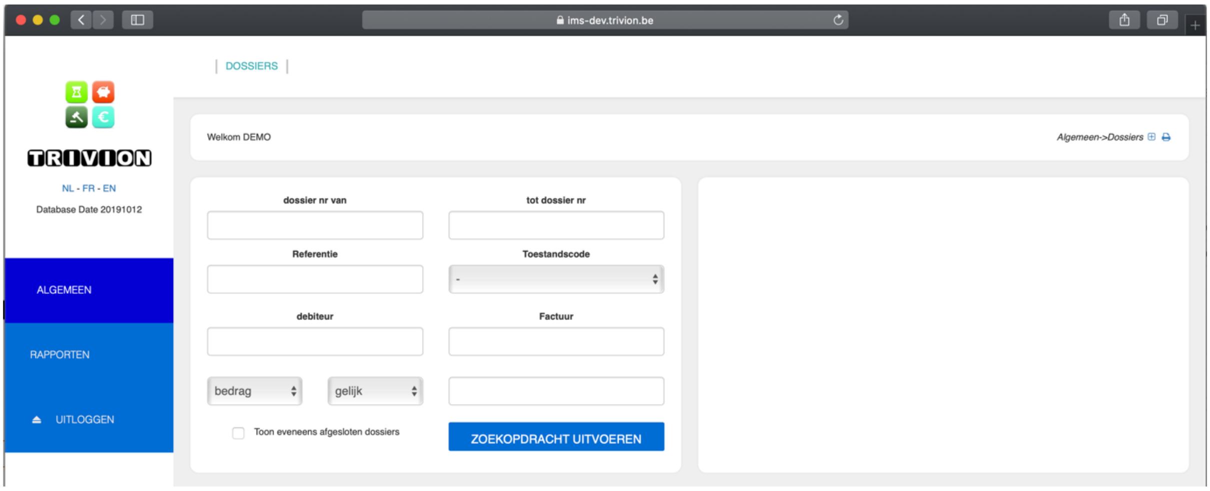
1. Environments

There are 2 environments available for the api and the application.

LIVE: https://ims.trivion.be



TEST: https://ims-dev.trivion.be



In this document you will find the word {ENVIRONMENT}. This placeholder needs to be replaced with the right environment at the time of using. So, when you are testing your interface with our application you need to use TEST. And when everything is working fine and you've got a GO from our side, you need to use the LIVE.

2. get-token

A token is necessary to get access to the api of ims.trivion.be. To get a token you first need to request a client_id and a client_secret by sending a mail to it@incassoservicedesk.be .

These client_id and client_secret are strict confident and can only be used by 1 application. So do not share it.

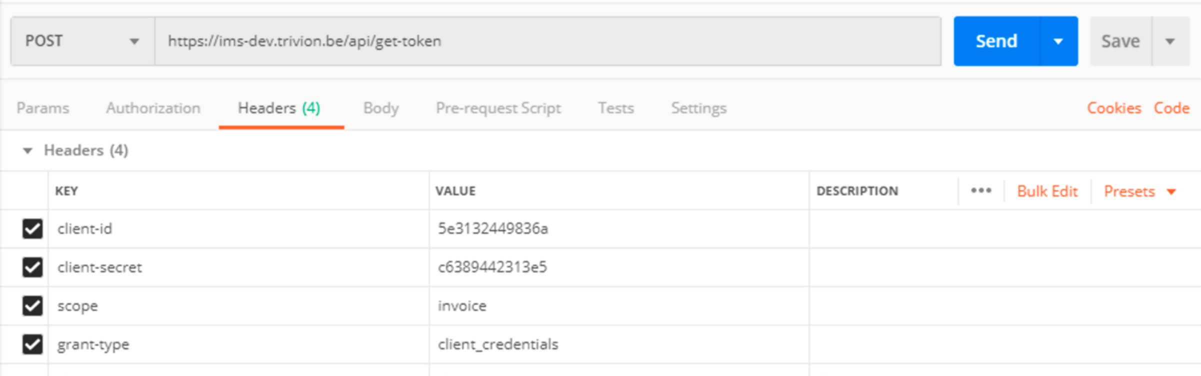
To get a token you need to make a POST request.

Method	POST
Url	{ENVIRONMENT}/api/get-token

With the following fields in the header.

client-id	{client-id}
client-secret	{client-secret}
scope	One or more scopes separated by comma. Depending by the next request(s).
grant-type	client_credentials

Postman Sample



The success result will be a JSON response like this.


```
{
    "status": 200,
    "code": "OK",
    "data": {
        "token_type": "Bearer",
        "expires_in": 3600,
        "access_token": "fltMuasrmnqaZFD7YIIOZn1SNWszN3WwpH4jwMAKSSQlHUnktt0kCaItzbJsbrv9tgEE2Hf2nXE97
5OToCEfaRaKFU4a9geMgzlfJt9BDU8gFtLaeIax3ukgWO1pPJHlbKDwhSsoCotFKK7kGw0TSsWYOHEVlO4c24UFS8prdKoNFC6LdkY
JUSEO3CDEzUCcg3SuwsnmyBpoEu4zkQtDnUZ344HwdVOflNHv6KGcDvynji2goVE6Md6Eb3PH057pK5IqYAsYqmFgvdDlh39Y58TnA
W2ixE8R2qA2UMUQRshMryHTMJTOodMZcf65rj6KBZK5NvsVbQpNfIiZZKVXSuorG8iQ7HadZR89bsWHpzMbeAmduumMBV9DKQclOfU
eE6R9Tcgc1lSAV7SpO0bIhukiDCyQeuGPD0TF4vOo4SU6nI12T10kw6q8NNLTmzWJOILaQdAPvJrd4QTN6TpVfM0Fo4QspaNsD8o6m
UxB3jVs7PVxfRjJ5JD4iGubky2LxyFWPyQX3gHA7FU1xrVXURz8dDnIUXGvtxxpyi431c3UMr5Ia6NEkQUrZu0DBiEM77RtxOjwsyn
myJmgX9vyiNrU3ZOqnb8AKsZx00wkjWMidLukIVw35tqBrAfKgf6XBxCpe7Ydfc6Jq3EpdY7ClJfUWZ7N6mkyP2Vyl4fnx4qL76piq
QTQwnoVUshEZaYDZlHnl86qaRqfd7szFVW1HlRMkFqxWFvbulJNzfgjJD8zriauryEPCF4oWwQrXnr4xRt2vUDqrCxPBDeWx3iQgyi
wymK0qNtjh1ZSsCYVFfSfiQOi24nItYSfrX6mXOh00adPctQASZajct38myJJ7Zv2WrsgEvBcSV0AQnr2cpS0QvoHjlukAwYKbrm9i
IzPnj7qGZzCoIaaqT0kZQXdEq6gYOOoH9c72gyG6nBMbIDMBNtxSLIKsyiIeKJ5peoOJHA9vOPYR09pDlSCBi91c9NXFI03blXlHau
cPJmj"
    }
}
```

In the data part you will get the following values

token_type	The type of the token
expires_in	Every token is only valid for 3600 seconds
access_token	This is unique text token of 1000 characters and needs to be used for all the other request.

A failure response will look like this.


```
{
    "status": 400,
    "code": "NOK",
    "error": "Json Request: Client Id and Client Secret not valid."
}
```

3. submit-invoice

To get invoices in our system, you can send them through the "submit-invoice" method of our api. This is done invoice by invoice. But within our system we can decide whether we create a file per invoice or per debtor.

This method will only create a file once, and if you sent it multiple times it will only check and update the payments.

You will always get the same response.

Instead of passing a username and password you need to add a token as an Authorization key in the header of your request. The token must be generated with 'invoice' included in the scope. Otherwise you cannot access this method.

To send an invoice to our api you need to make a POST request.

Method	POST
Url	{ENVIRONMENT}/api/submit-invoice

With the following fields in the header.

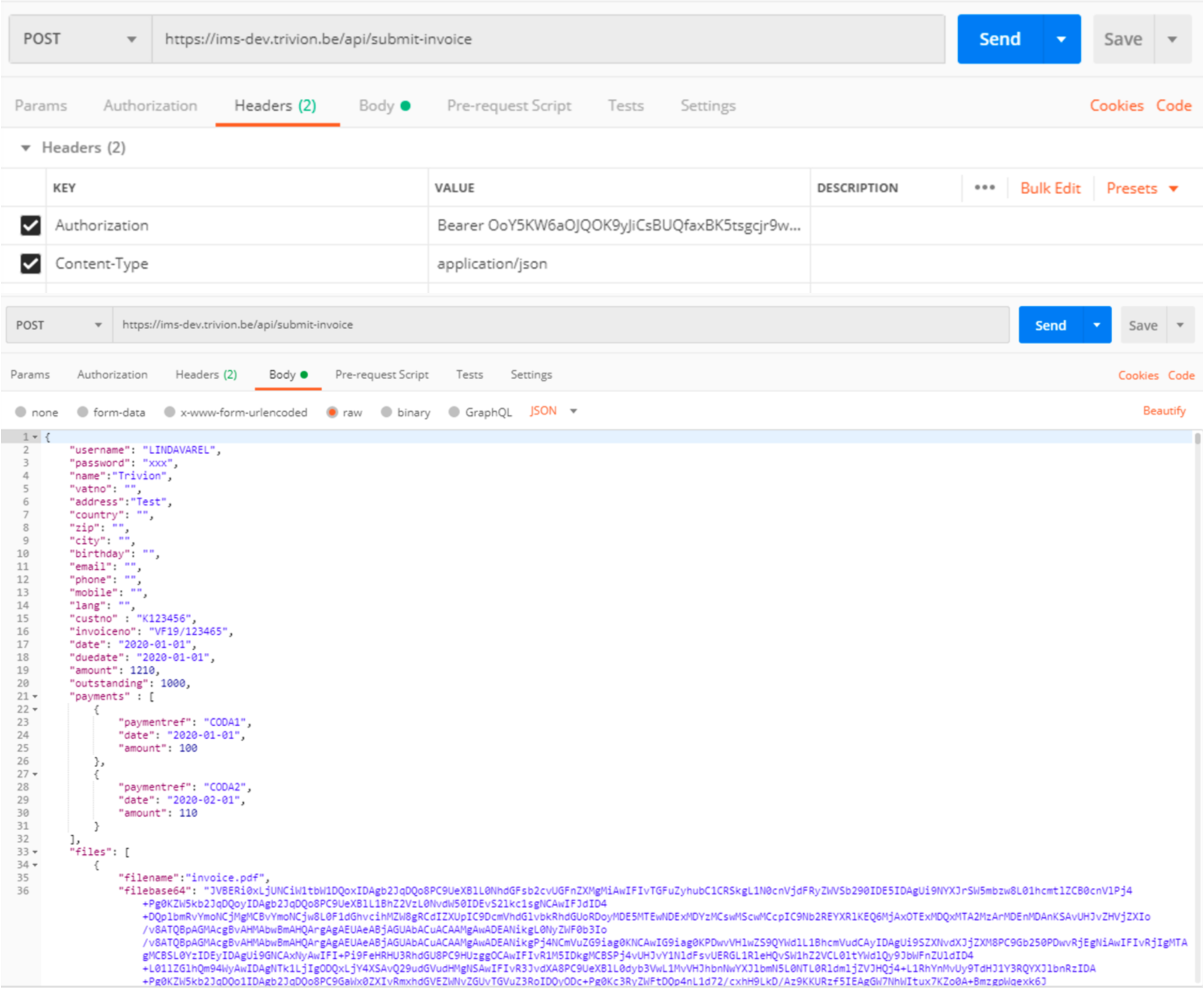
Authorization	{token_type} {access_token}
Content-Type	application/json

And in the body you need to add all your data.

username	This will be sent to your customer. They can use it to login on our website. (mandatory)
password	This will be sent to your customer. They can use it to login on our website. (mandatory)
name	The name of your customer (mandatory)
vatno	VAT Registration No
address	Address of you customer (mandatory)
country	ISO country code
zip	Zip code
city	City name
birthday	Day of birth (Optional)
email	
phone	
mobile	
lang	Language code
remarks	
custno	Your internal no of your customer. (mandatory)
custtype	Classification of your customer. Values: B2C, B2B or B2G (Optional) When not provided the system will use the vatno as an indicator. When vatno is empty = B2C otherwise B2B.

payoutref	A reference can be added. The will be used when your account is setup to payout “Per file” instead of “Per period” (Optional)
invoiceno	Your invoice no (mandatory). This can only be sent once.
date	Invoice Date (mandatory)
duedate	Due date of your invoice (mandatory)
amount	Initial Amount including VAT (mandatory)
outstanding	Outstanding Amount (mandatory). This will be used for the file.
payments	An Array of payments can be added. Full payments or partial. Every payment will be checked everytime and updated or created.
type	Optional: Indication of the type of payment. Values: payment, creditnote. Default: payment when not provided
paymentref	Internal reference for payment. This must be unique per invoiceno
date	Date of your payment
amount	Total amount paid
files	An Array of files can be added in Base64 string. For example, your invoice and reminder document. One file must contain a filename and the filebase64 string
filename	The name of your document
filetype	Indication of the type of your document. Values: invoice, reminder, other. Default: invoice
filebase64	Base64 value of your document

Postman Sample



The success result will be a JSON response like this.


```json
{
    "status": 200,
    "code": "OK",
    "data": {
        "invoiceno": "K123456",
        "filenr": "102000134",
        "filedate": "2020-03-14",
        "statuscode": "NEW",
        "statusdescription": "Nieuw dossier - Nouveau dossier",
        "statusdate": "2020-03-14",
        "debtor": {
            "name": "Trivion",
            "vatnr": "",
            "address": "",
            "zip_code": "",
            "city": "",
            "country_code": "BE",
            "telephone": "",
            "gsm": "",
            "e_mail": ""
        }
    }
}
```

A failure response will look like this.


```
{
    "status": 401,
    "code": "NOK",
    "error": "outstanding is missing."
}
```

4. submit-payment

Payments can also be sent to our system. This can be done by using the “submit-payment” method of our api. The payment will be added to your file.

Instead of passing a username and password you need to add a token as an Authorization key in the header of your request. The token must be generated with ‘payment’ included in the scope. Otherwise you cannot access this method.

To send an invoice to our api you need to make a POST request.

Method	POST
Url	{ENVIRONMENT}/api/submit-payment

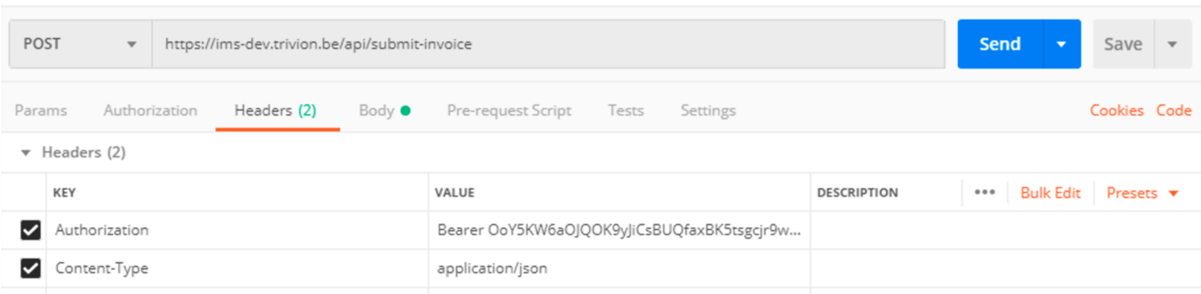
With the following fields in the header.

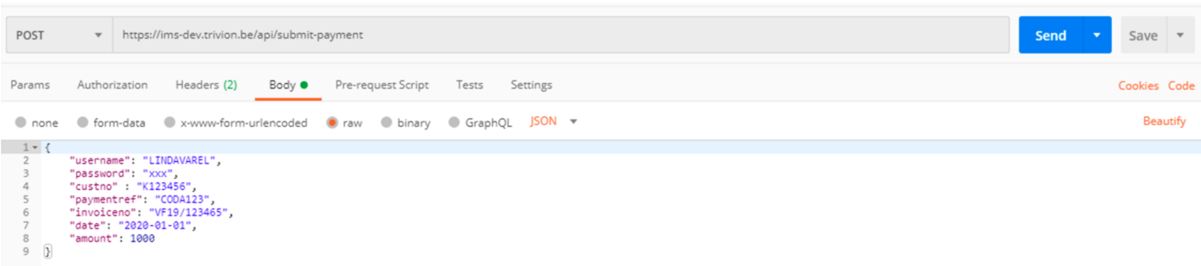
Authorization	{token_type} {access_token}
Content-Type	application/json

And in the body you need to add all your data.

username	This will be sent to your customer. They can use it to login on our website. (mandatory)
password	This will be sent to your customer. They can use it to login on our website. (mandatory)
custno	Your internal no of your customer. (mandatory)
type	Optional: Indication of the type of payment. Values: payment, creditnote. Default: payment when not provided
paymentref	Your payment reference (mandatory).
invoiceno	Your invoice no (mandatory).
date	Invoice Date (mandatory)
amount	Total payment amount (mandatory)

Postman Sample





The success result will be a JSON response like this.

```
{
    "status": 200,
    "code": "OK",
}
```

A failure response will look like this.

```
{
    "status": 401,
    "code": "NOK",
    "error": "FILE DOES NOT EXISTS FOR INVOICE VF19/123466"
}
```

5. get-feedback

If you would like to get only feedback of a certain invoice, you can use the “get-feedback” method of our api.

Instead of passing a username and password you need to add a token as an Authorization key in the header of your request. The token must be generated with ‘feedback’ included in the scope. Otherwise you cannot access this method.

To send an invoice to our api you need to make a POST request.

Method	POST
Url	{ENVIRONMENT}/api/get-feedback

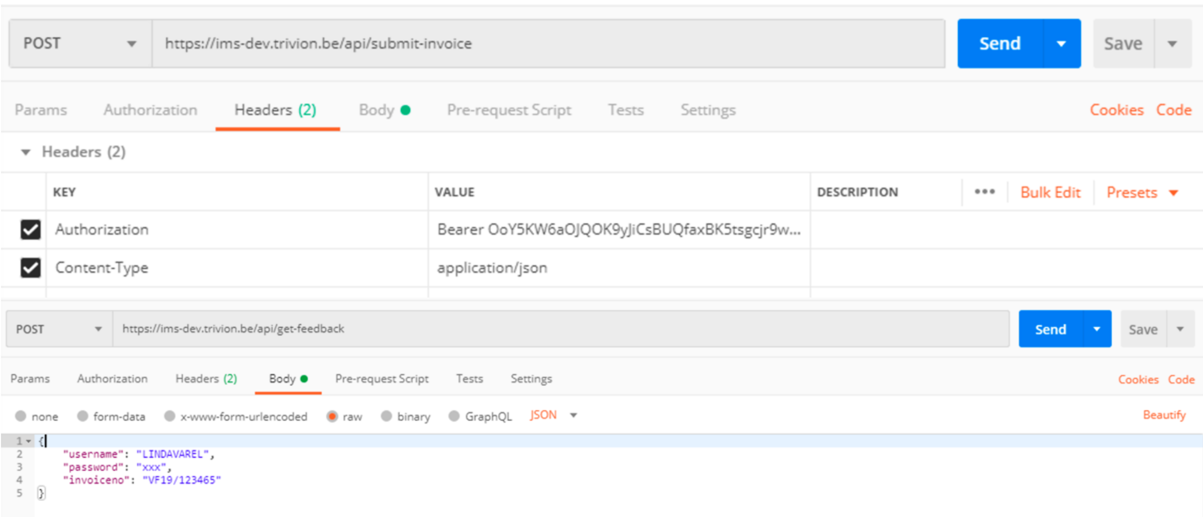
With the following fields in the header.

Authorization	{token_type} {access_token}
Content-Type	application/json

And in the body you need to add all your data.

username	This will be sent to your customer. They can use it to login on our website. (mandatory)
password	This will be sent to your customer. They can use it to login on our website. (mandatory)
invoiceno	Your invoice no (mandatory).

Postman Sample



The success result will be a JSON response like this.


```
{
    "status": 200,
    "code": "OK",
    "data": {
        "invoiceno": "K123456",
        "filenr": "102000133",
        "filedate": "2020-02-12",
        "statuscode": "CLOSED",
        "statusdescription": "Gesloten dossier",
        "statusdate": "2020-02-12",
        "debtor": {
            "name": "Trivion",
            "vatnr": "",
            "address": "",
            "zip_code": "",
            "city": "",
            "country_code": "BE",
            "telephone": "",
            "gsm": "",
            "e_mail": ""
        }
    }
}
```


A failure response will look like this.


```
{
    "status": 401,
    "code": "NOK",
    "error": "FILE DOES NOT EXISTS FOR INVOICE VF19/123466"
}
```

6. get-statuscodes

In the submit-invoice and get-feedback the system returns a "`statuscode`". To get the full list of all the active statuscodes you can call the "get-statuscodes" method of our api.

Instead of passing a username and password you need to add a token as an Authorization key in the header of your request. The token can be generated with all of the available scopes, but at least one. Otherwise you cannot access this method.

To send an invoice to our api you need to make a POST request.

Method	POST
Url	{ENVIRONMENT}/api/get-statuscodes

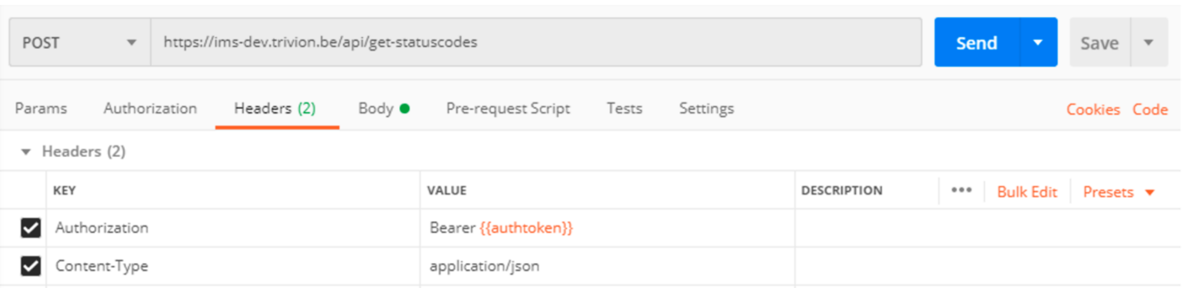
With the following fields in the header.

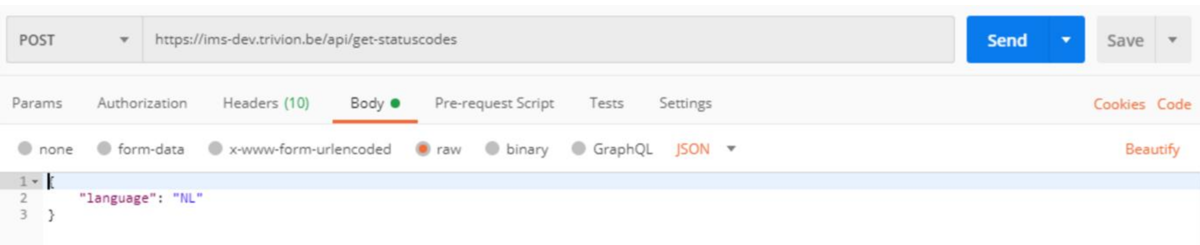
Authorization	{token_type} {access_token}
Content-Type	application/json

And in the body you need to add all your data.

language	Language code. NL = Dutch, FR = French, EN = English (Optional)

Postman Sample





The success result will be a JSON response like this.


```
{
    "status": 200,
    "code": "OK",
    "data": [
        {
            "code": "AANG_SCHULDVORDERING",
            "description": "Aangifte schuldvordering"
        },
        {
            "code": "ADRES",
            "description": "Adres aanvraag"
        },
        {
            "code": "ADRES_HER",
            "description": "Adresaanvraag herinnering"
        },
        {
            "code": "ADV1",
            "description": "Dossier bij advocaat"
        },
        {
            "code": "AFBETALINGSOVEREENKO",
            "description": "AFBETALINGSOVEREENKOMST"
        },
        {
            "code": "AGENDA",
            "description": "Dossier te bekijken"
        },
        {
            "code": "AGENT",
            "description": "via agent"
        },
        {
            "code": "BETAALBELOFTE",
            "description": "Betaalbelofte"
        },
        {
            "code": "BETAALPLAN",
            "description": "Betaalplan overeengekomen"
        },
        {
            "code": "BEV_DAGV",
            "description": "Bevestig dagvaarding"
        },
        {
            "code": "BEZOEK",
            "description": "bezoek"
        },
        {
            "code": "BEZOEKVERSLAG",
            "description": "Bezoekverslag"
        },
        {
            "code": "BEZOEKVERSLAG_AANM",
            "description": "Bezoekersverslag aanmaken"
        },
        {
            "code": "BRF1",
            "description": "Eerste aanmaning"
        },
        {
            "code": "BRF1_ZONDER_KOSTEN",
            "description": "Brief1 zonder kosten"
        },
        {
            "code": "BRF2",
            "description": "Tweede aanmaning"
        },
        {
            "code": "BRF3",
            "description": "Derde aanmaning"
        },
        {
            "code": "BRIEF",
            "description": "BRIEF AAN DEBITEUR"
        },
        {
            "code": "BUITENLAND",
            "description": "Buitenland"
        },
        {
            "code": "BZKVERSLAG_GEKOPPELD",
            "description": "Bezoekverslag gekoppeld"
        },
        {
            "code": "CLOSED",
            "description": "Gesloten dossier"
        },
        {
            "code": "DAGV_VOORSTEL",
            "description": "Voorstel tot dagvaarding"
        },
        {
            "code": "DISPUUT",
            "description": "Dispuut"
        },
        {
            "code": "DW",
            "description": "Doorgeven aan deurwaarder"
        },
        {
            "code": "DW2",
            "description": "Bij deurwaarder"
        },
        {
            "code": "E-MAIL",
            "description": "e-mail"
        },
        {
            "code": "EXPLOIT",
            "description": "Exploit toegevoegd"
        },
        {
            "code": "HUISBEZOEK",
            "description": "huisbezoek"
        },
        {
            "code": "HUISBEZOEK_UITGV",
            "description": "Huisbezoek uitgevoerd"
        },
```

```
        {
            "code": "KLACHT_CONSUMENTENB",
            "description": "Klacht consumentenbond"
        },
        {
            "code": "KLACHT_ECC",
            "description": "Klacht ECC"
        },
        {
            "code": "KLACHT_TESTAANKOOP",
            "description": "Klacht Testaankoop"
        },
        {
            "code": "LAATSTE AANMANING",
            "description": "ALLERLAATSTE INGEBREKESTELLING AANGET./ONTVANGKAART"
        },
        {
            "code": "NEW",
            "description": "Nieuw dossier - Nouveau dossier"
        },
        {
            "code": "NEW_TOCHECK",
            "description": "The credit info that was found needs to be looked at."
        },
        {
            "code": "PAUZE",
            "description": "Afwachtend op antwoord"
        },
        {
            "code": "REOPEN",
            "description": "Factuur opnieuw betaalbaar"
        },
        {
            "code": "RINGRING_SMS_LIST",
            "description": "Ringring sms lijst"
        },
        {
            "code": "RINGRING_TEL_LIST",
            "description": "Ringring telefoon lijst"
        },
        {
            "code": "SCHULDBEKENTENIS",
            "description": "Schuldbekentenis"
        },
        {
            "code": "TEL.CONTACT",
            "description": "Telefonisch contact"
        },
        {
            "code": "TEL1",
            "description": "Telefonische actie"
        },
        {
            "code": "TEL1_DONE",
            "description": "A phone call was logged."
        },
        {
            "code": "TEL2",
            "description": "Telefonische actie 2"
        },
        {
            "code": "TEL_TTS",
            "description": "Telephone text to speech"
        },
        {
            "code": "VONNIS",
            "description": "Vonnis bekomen"
        }
    ]
}
```


A failure response will look like this.


```
{
    "status": 400,
    "code": "NOK",
    "error": "INVALID TOKEN"
}
```

7. submit-remark

Remarks can also be sent to our system. This can be done by using the “submit-remark” method of our api. The remark will be added to your file.

Instead of passing a username and password you need to add a token as an Authorization key in the header of your request. The token must be generated with ‘remark included in the scope. Otherwise you cannot access this method.

To send an invoice to our api you need to make a POST request.

Method	POST
Url	{ENVIRONMENT}/api/submit-remark

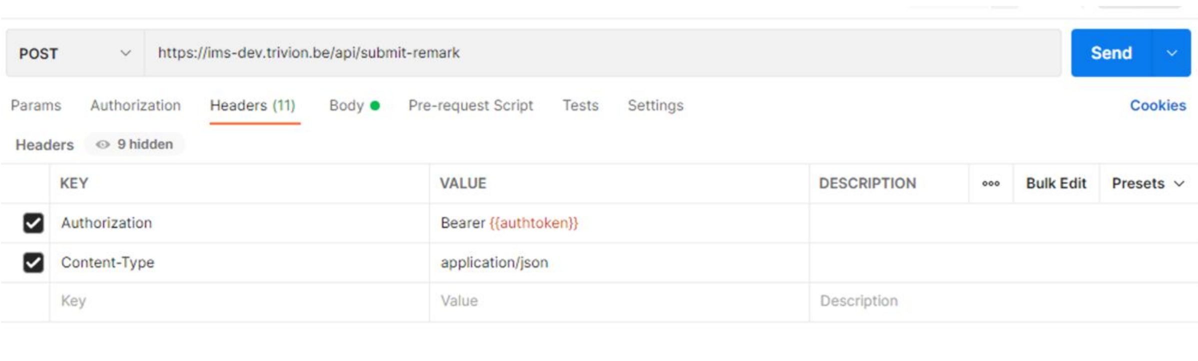
With the following fields in the header.

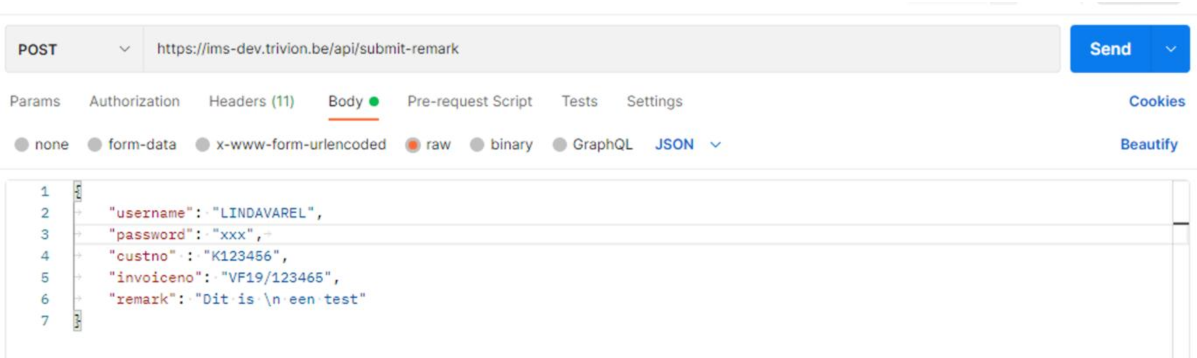
Authorization	{token_type} {access_token}
Content-Type	application/json

And in the body you need to add all your data.

username	This will be sent to your customer. They can use it to login on our website. (mandatory)
password	This will be sent to your customer. They can use it to login on our website. (mandatory)
custno	Your internal no of your customer. (mandatory)
invoiceno	Your invoice no (mandatory). This can only be sent once.
date	Remark Date (mandatory).
remark	This can be a multiline text with \n as line delimiter. (mandatory)

Postman Sample





The success result will be a JSON response like this.

```
{
    "status": 200,
    "code": "OK",
}
```

A failure response will look like this.

```
{
    "status": 401,
    "code": "NOK",
    "error": "FILE DOES NOT EXISTS FOR INVOICE VF19/123466"
}
```