AVNS GNR - Get Netsuite Records

Setup Guide

Table of Contents

GET NETSUITE RECORDS SETUP GUIDE

Overview

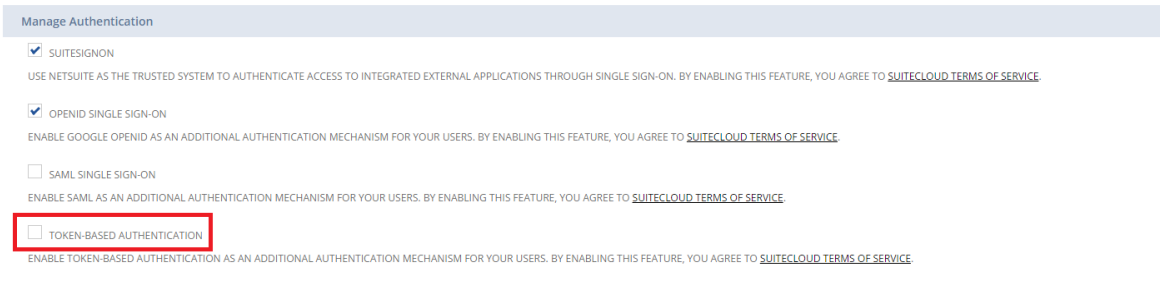
This document describes how the GNR bundle is installed and the steps that must be followed in order to configure it according to the customer needs. In this document we will go through installation of the bundle, creation of tokens, enabling the feature token-based authentication, creation of an employee which will be associated to the bundled custom role and creation of a new integration using the created tokens. Configuring hidden fields that come with the bundle is also part of this guide. Moving on we will need to customize a saved transaction search which is a very important part of this bundle. The structure of the data returned from the search is based on the standard 'Sveafaktura 1.0' created by SFTI. Certain customizations are made in order to adapt it to NetSuite. More on this in the search section in the document. Lastly we will be testing the integration.

Installation

Enable feature Token-based Authentication.

Before we install the bundle we need to enable this feature, otherwise a setting in the bundled role will not be imported.
The feature token-based authentication will be the only sign-in method used for our RESTlets in this documentation. To enable the feature navigate to:

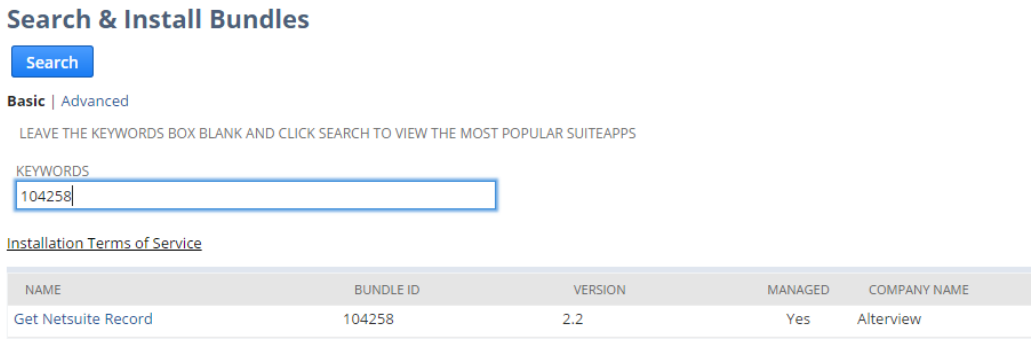
1. Setup -> Company -> Enable Features under Setup Tasks
2. In the subtab SuiteCloud, locate TOKEN-BASED AUTHENTICATION and mark the checkbox:



3. Read the Terms of Service and click on I agree at the end of the page.
4. Save the changes.

Installation of Bundles

Install bundle ID 104258, name: Get Netsuite Record.
We need to also install the bundle with ID 104259 named Outbound Invoice.
Go to Customization -> SuiteBundler -> Search & Install Bundles. In the Keywords field enter bundle ID and click on search:



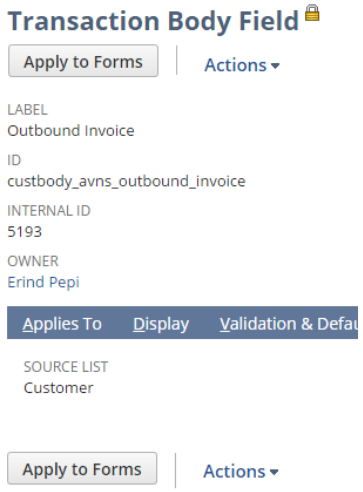
Click on the Bundle Name and then on Install. Repeat these steps for the bundle Outbound Invoice and wait for the installation to complete.

Editing the Components of the bundle

The two bundles add to the environment custom fields, records, one role, a saved search, a list and two different scripts for the two different RESTlets – the first to fetch records and the second to acknowledge the fetched records.
All custom fields that come with the bundle are hidden as default to not interfere and create confusion in the customer's environment. However, some are needed to be shown and be populated with values in order for the records to be fetched.

Applying custom fields to forms

First we need to apply the bundled custom fields to the forms we need before we can continue. Navigate to the field themselves and click on the 'Apply to Forms' button. It is recommended we apply the fields on Invoice forms but also on Sales Orders in the case the customer creates invoices from sales orders. Example:



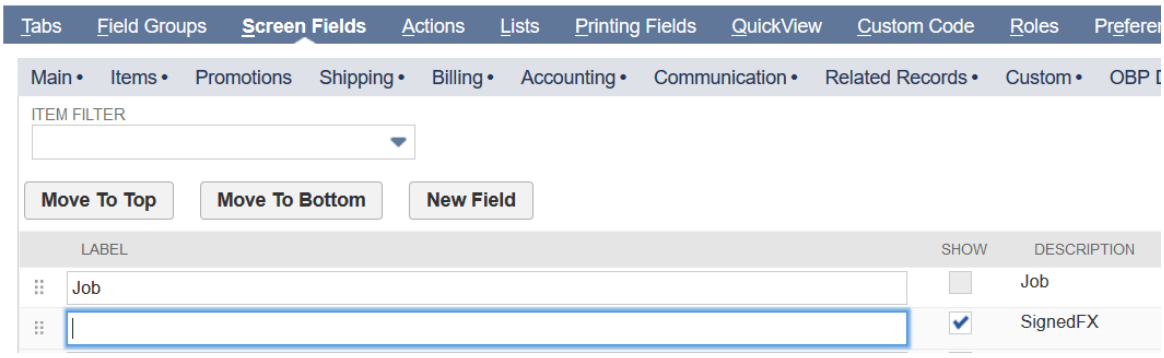
List of custom fields to apply to forms:

Type of field	Name	Optional/Mandatory
Entity – Move to settings	Outbound Invoice	Mandatory
Entity	EAN	Optional
Entity	Registered Office BrE (SWE: Säte)	Optional
Entity	GLN	Optional
Entity	Invoice Delivery Method	Mandatory
Entity	VAN Operator	Optional
Transaction Body	Do not send to Customer	Mandatory, used to avoid sending a specific invoice.
Transaction Body	Send to Scancloud	Used for Credit Memos
Transaction Body	Invoice Delivery Method	Mandatory – set to eInvoice

Transaction Body	Outbound Invoice	Optional, sources from Entity (Customer) field with same name
Transaction Body	Acknowledged Log	Mandatory
Transaction Body	Acknowledged Timestamp	Mandatory
Transaction Body	Trx Attachment PDF	Optional
Transaction Body	Transaction PDF	Optional
Item	Show Components GNR	Optional – Only for group items
Transaction Column	Signedfx	*Mandatory but the label from the form must be emptied. See below.

Editing the fields on the records

1. **Outbound Invoice (entity & transaction body)** - We need to mark the invoices we want to send electronically by marking the checkbox Outbound Invoice on the entity record. From the moment we do that, all invoices created afterwards will inherit the setting on the entity. The transaction body field Outbound Invoice will source from the entity record from the field with the same name.
2. **Do not send to customer (transaction body)** - In case we want to exclude a certain transaction from being fetched via the RESTlet then we can mark 'Do not send to customer' on the transaction.
3. **Invoice Delivery Method (entity & transaction body)** - Just as in step one we set this on the entity record and transactions created afterwards will inherit the setting. For the purpose of this bundle we set it to 'eInvoice'.
4. **Acknowledged Timestamp & Log (transaction body)** - used to set the timestamp and eventual error messages once the Acknowledged RESTlet tries to mark the transaction as acknowledged.
5. **EAN** (Entity) – Customer ID used in Denmark.
6. **Registered Office BrE and GLN (Entity)** - shall be filled only if required by the customer. Customizing the bundled saved search.
7. ***Signedfx** – Must be applied on the transaction form but the label must be emptied. In the form navigate to Screen Fields – Columns. Example:



NOTE: If we do not remove the label the column will show and it will display an error message which is normal.

Editing the Saved Search

As next step we need to customize the bundled saved transaction search named GNR ScanCloud Invoice Template. This saved search returns invoices and credit memos in a ready-format for the external party to fetch via the RESTlet. The structure of the fetched data will be in JSON. We will be changing the results of the search but also one criteria for environments with only one language activated. More info on this in section 'Environments with Single Language'. A base understanding of the saved search is needed before we make certain changes. The different columns on the search are divided into 4 categories. Identification of these categories is done via the prefix in the custom label. The categories are:

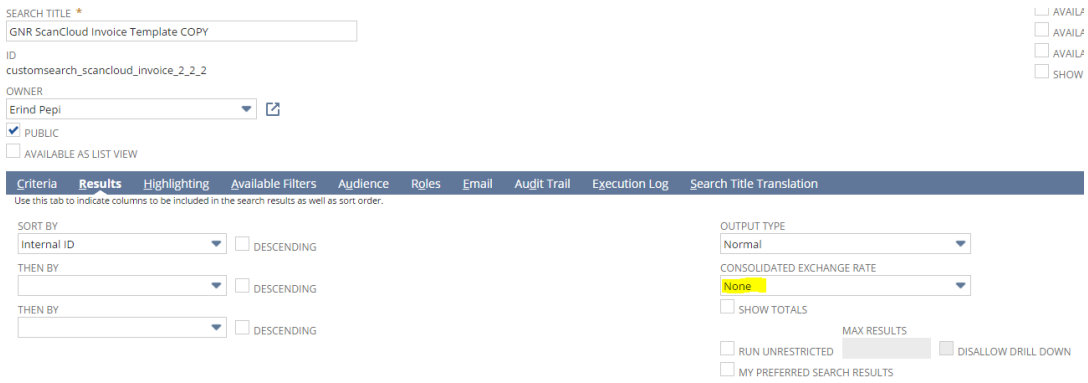
1. DOC
2. INV
3. INVLINE
4. TAX
5. REF

Short explanation of the categories:

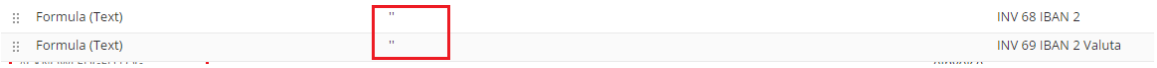
DOC – Used for development purposes, should NOT be changed or considered for editing.
INV – Columns that return info found in the transaction header known as the 'Main Line' in the searches.
INVLINE – Columns that return info found on the item rows of transactions.
TAX – Columns that return info found on the tax line. There will be one row per tax rate in the search results.
REF – Not used at the moment. Needed to return referring documents to the transactions.

Certain columns in the bundled saved search are customer specific so we will need to edit them. Unable to do so as the bundled search is locked, we will make a copy of it and work in a separate search which will be used by the RESTlet request. We have locked the original search as it might be needed at later times for comparison purposes or most likely to fix any errors we might introduce while making the customizations.
NOTE: It is very important to not change the custom label of the columns in the result. They serve as ID for returning the correct JSON structure for the RESTlet request.

Before editing the columns we need to make sure that CONSOLIDATED EXCHANGE RATE is set to 'None' (only available in OneWorld environments with more than one subsidiary):



When editing the columns, we will only edit within the 'Formula' column in the search. In the example below we can write text between the single quotation marks.



List of columns we might need to edit:

1. INV 15 Orgnummer – Organisation number. Many customers already have a custom field for this one, so maybe we should remap it in the search.
2. INV 25 Skattebefrielse anledning – Tax exemption reason.
3. INV 38 Betalsätt – Payment method. Plusgiro or Bankgiro. We suggest adding a new field on the subsidiary if the environment is OneWorld (more than one subsidiary) and saving the payment method per subsidiary, the string 'Plusgiro' or 'Bankgiro'. We then point to that field in the search by inserting the id of the field as a Formula (Text). We can also hardcode it in the search if the payment method will always be the same.
4. INV 40 Kontonummer – Account number. Again we recommend to have a custom field on the subsidiary and point to that field in the search by inserting the id of the field as a Formula (Text).
5. INV 41 BIC – BIC or SWIFT code. Same suggestion as the previous column.
6. INV 66 IBAN 1 – Same suggestion as above.
7. INV 67 IBAN 1 Valuta – Currency of the account number for IBAN 1. Same suggestion as above. We have included support for up to 5 different IBANs and belonging currencies.
8. INV 42 OCR – OCR number
9. INV 59 Reference 1 – To be mapped if customer has a specific reference field
10. INV 60 Reference 2 – Same as INV 59 with the possibility for another reference field. Some customers send more than 1 reference.
11. INV 76 Invoice Attachment – Attachment to invoice such as sales order, cotract etc. Make sure to return the ID of the field such as {field.ID}, same as in INV 77.

12. INV 77 Invoice PDF – Make sure to return the ID of the file, not the filename. I.E. {file.id}
13. INV 79 – For customers not able to receive eInvoices, Scancloud has the possibility to send the invoices via e-mail. If the value in this field is 5 then the invoice will be e-mailed to the address in INV 78. If the value here is other than 5 then the invoice will be sent in an envelope via postal mail.

A comprehensive description of most of columns can be found in Sveafaktura 1.0 – SFTI's document with the term list that the standard includes (in Swedish):
http://www.sfti.se/download/18.aa4f88114a4d3fe64b8484/1418745668729/SFTI-Basic+Invoice-1.0+%282%29.xls

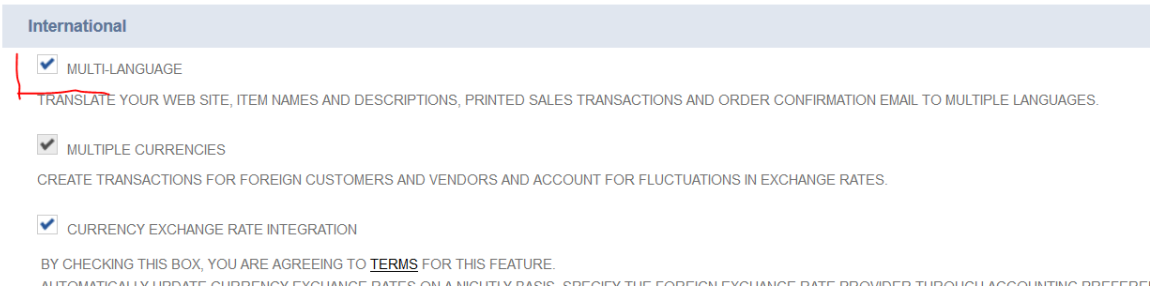
Environments with no Subsidiaries

For Companies/Environments that are not One World (not having subsidiaries) we will need to change all address related fields on the Sales party side. In the bundle we populate these search columns with fields on the subsidiaries which are missing in an environment that is not One World. For this reason we will have to hard code in the search the following columns:

INV 19 Säljandepart – Legal name of the selling party
INV 20 Säljandepart gata – Street name and number
INV 21 Säljandepart stad – City
INV 22 Säljandepart postnummer – Zip code
INV 23 Sälj.part land – Country
INV 24 Sälj.part Postbox
INV 25 Skattebefrielse anledning – Tax exemption. For Sweden the standard text is: 'Godkänd för F-skatt'
INV 26 S.stad – City of the address registered at tax registration
INV 27 Landkod – Two lettered country code according to ISO Alpha-2
INV29 Säljandepart VATnummer – VAT number
INV 30 Säljare GLN – GLN number
INV 31 Säljandepart Orgnummer – Organisation number for the seller
INV 32 Säljandepart Telefon – Phone number
INV 33 Säljandepart epost – E-mail address

Environments with Single Language

A criteria in the saved search is written for supporting environments with many languages activated and translation of item names. For single language environments we will have to remove a criteria from the saved search. If you are unsure whether the environment is single or multiple-language, that can be checked at Setup -> Company -> Enable Features. Under subtab 'Company' check section 'International' for Multi-language:

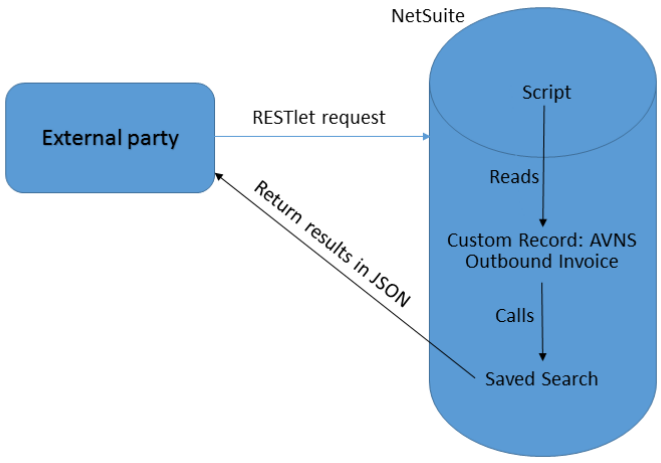


If the checkbox in the screenshot above is not checked then we will have to remove the following formula-criteria from the saved search:

CASE WHEN NVL({item.id},'0')='0' then 1
ELSE
CASE WHEN {taxline.id}='T' THEN 1
ELSE
CASE WHEN (NVL({customer.language}, 'null') = NVL({item.language}, 'null')) OR ({item.language} is null) THEN 1
ELSE 0 END END END

Creating a post in the custom record

We have to point to the newly created search in the integration. A visual and simplified overview of the process flow can be seen in the picture below.



As we understand from the picture above we have created the Saved Search but we need to point to that search from the Custom Record called AVNS Outbound Invoice. In order to do that we need to create a post in that Custom Record. Navigate to Customization -> List, Records & Fields -> Record Types and locate AVNS Outbound Invoice. We create a new post and give it a name. The following need to be set in the record post

- Name: Invoice – This will be the id in the RESTlet request
- Description: Free of choice
- Transaction Type: Must be set to 'Invoice'
- Search Name: The search used for fetching the invoices

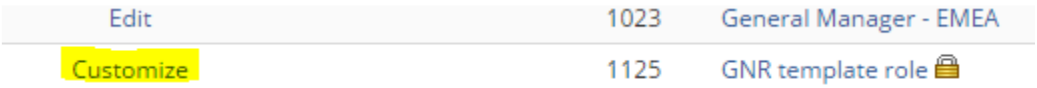
Token-based Authentication

Token-based authentication will be standard in this bundle. No login via user credentials such as username or password shall be used. Below steps show how to enable the feature in NS, how to create an access token and how to test the RESTlet via an add-on in Firefox.

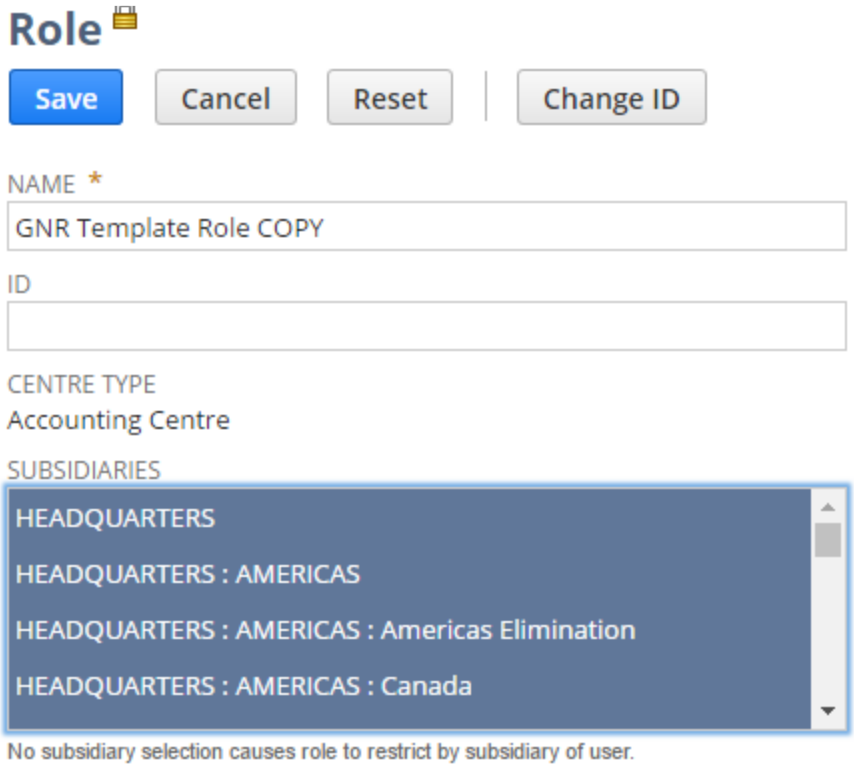
Edit the bundled role

NOTE: This section is applicable only for One World environments with different subsidiaries.

A new role is added in the environment with the installation of the bundle Get Netsuite Record. It includes the minimum amount of rights to make the RESTlet requests. Unfortunately it is impossible bundling the subsidiaries in this role as they differ from one customer to another. That being said we need to edit/create a new role with the subsidiary selection.
The bundled role is called GNR template role and can be found in the list of roles via Setup –> User/Roles -> Manage Roles. As the role is locked we only have the customize button to click on to the left of the role in the role list:



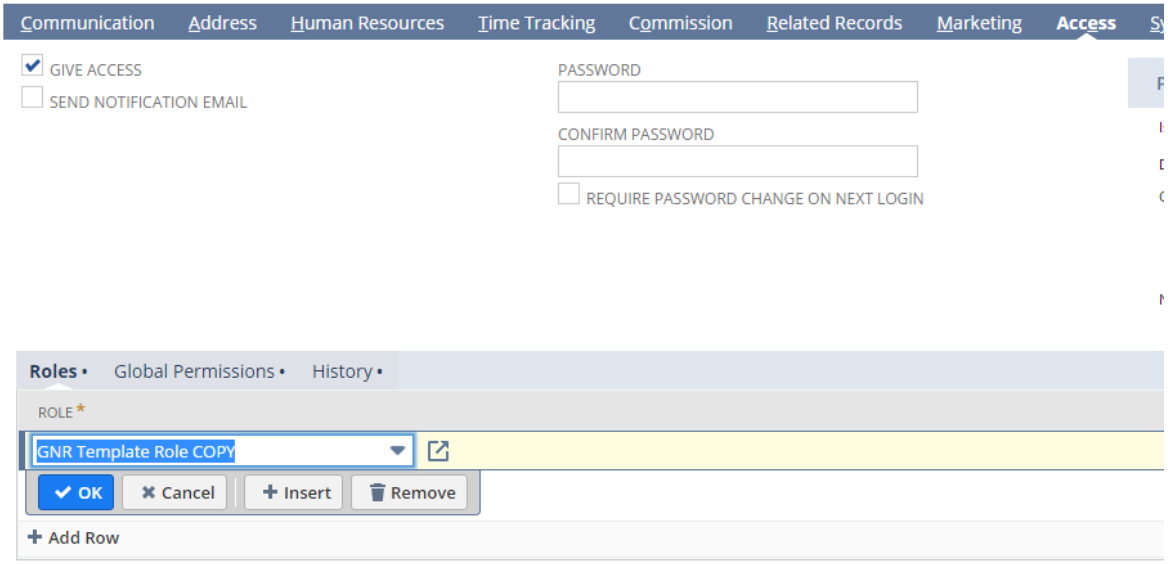
Depending on which subsidiary the transactions will be, we choose only the relevant subsidiaries. If we will be creating transactions in all subsidiaries then we can choose all of them (recommended):



Finally give the new role a name and save it.

Create a new employee

We need to create a new employee to which we will assign the new role that we created in the step above. As mentioned earlier this new role will limit the NetSuite access only to the minimum amount needed for external parties making the RESTlet requests. Netsuite requires we setup a password although we have set on the role to 'Log in using Access Tokens'. Create a password and assign the GNR Role to the employee/user.

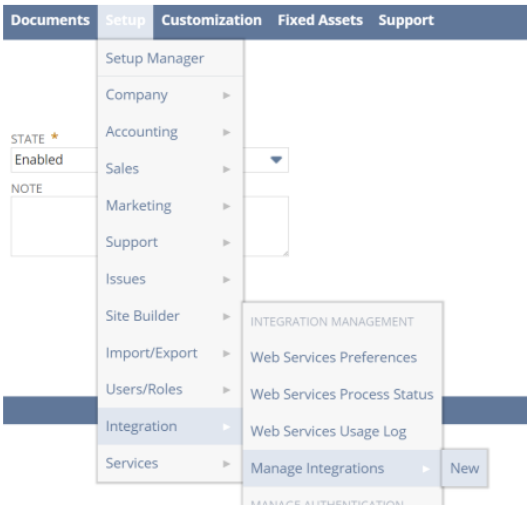


We need then assign the newly created role to the employee we created in the steps above.

Create a new integration

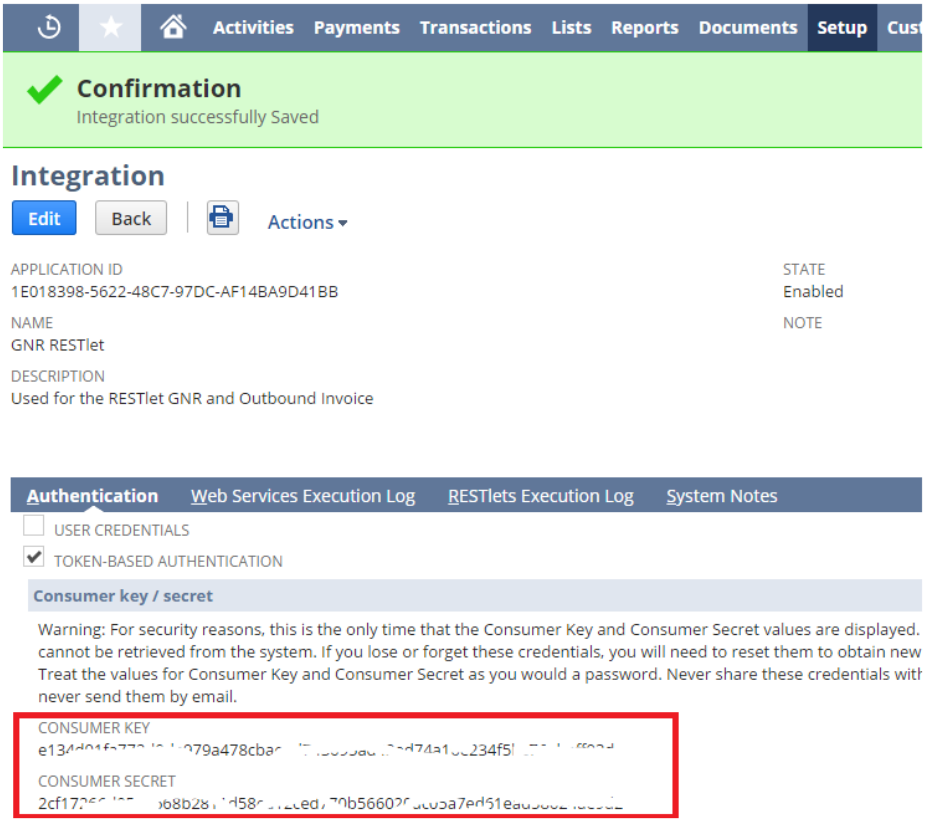
Before creating an access token we need to create a new integration which we will point to in the access token creation.

1. Navigate to Setup -> Integration -> Manage Integrations -> New



2. Fill in the form with the mandatory fields and mark only the checkbox ‘TOKEN-BASED AUTHENTICATION’ instead of ‘User Credentials’:

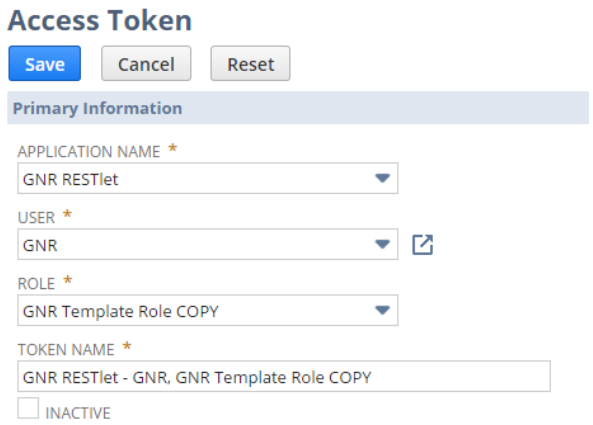
3. Save but REMAIN on the page.
4. Once saved we see Consumer Key and Consumer Secret at the bottom of the page. SAVE both of them as they only appear this one time only. If lost then a new integration needs to be created. Below is an example but they have been partly erased for security reasons.

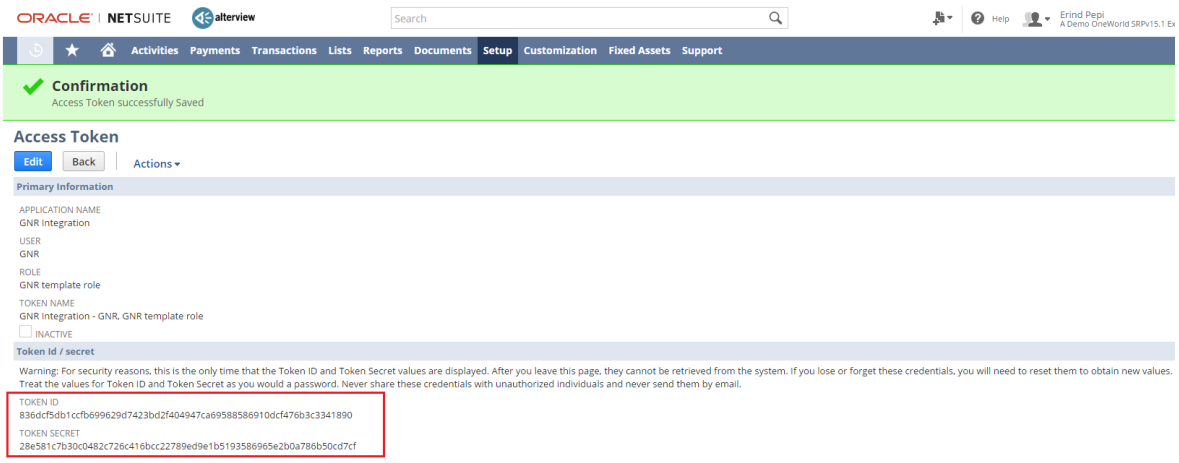


Create an Access Token

As next step we need to create the access tokens which will be provided to the party making the RESTlet requests to NS together with Consumer Key and Consumer Secret.

1. Navigate to Setup -> Users/Roles -> Access Tokens
2. Click on New Access Token
3. Application Name – We see that we have the name of the integration we saved earlier.
4. User – The employee we created earlier in this document.
5. Role – The bundled role that came with the bundle installation.
6. Save BUT STAY ON THE PAGE and **SAVE Token ID and Token Secret**.
   The below two screenshots illustrate step 3-6.



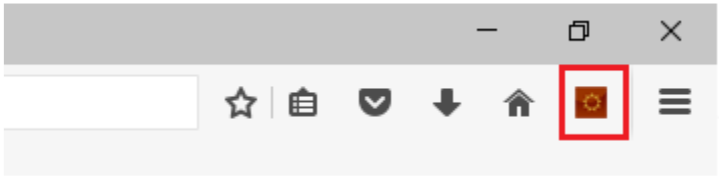


Testing the RESTlet integration

We are now ready to test the integration before we can hand over the tokens to the external party that will handle the RESTlet requests. As an easy and practical tool to test the integration we have found to be the free Firefox add-on RESTClient. Postman is also an alternative that works equally good.

Installation of RESTClient

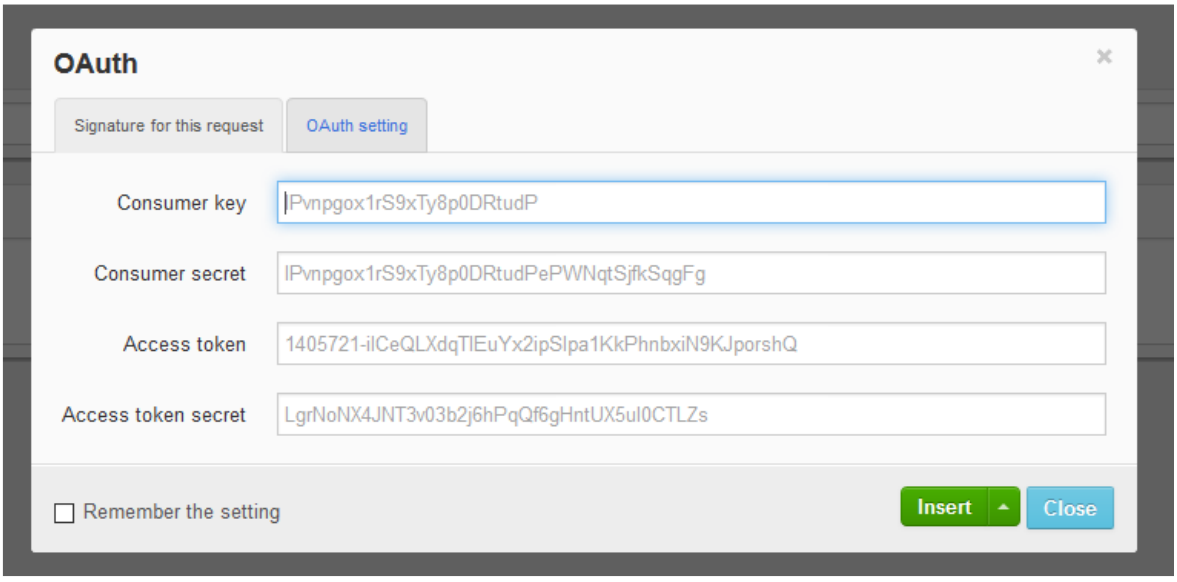
The add-on can be downloaded from the Mozilla add-on page:
https://addons.mozilla.org/en-US/firefox/addon/restclient/
After installation it should appear on the upper right corner in Firefox.



RESTClient request configuration

We need to now configure the RESTClient in order to make the RESTlet request. Open the add-on in Firefox and:

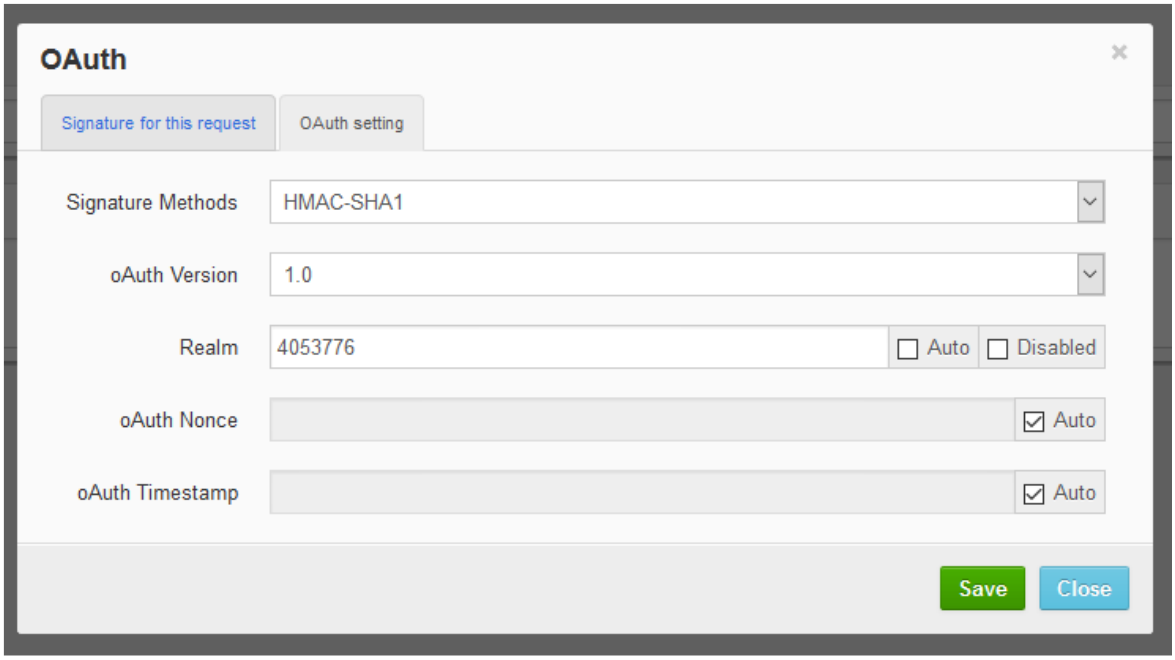
1. Navigate to Authentication -> oAuth (not oAuth2. The protocol we are using here is oAuth)
2. We see the following pop-up, tab 'Signature for this request' the fields of which we need to populate with our tokens and other details:



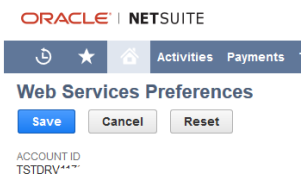
For tokens needed in the above screenshot, we have created them all earlier in this documentation:

a. Consumer Key
b. Consumer Secret
c. Access Token: In NS called TOKEN ID, created in the chapter creating an access token.
d. Access Token Secret: In NS called Token Secret.

Now that we are done with this tab we move on to the next 'oAuth setting':



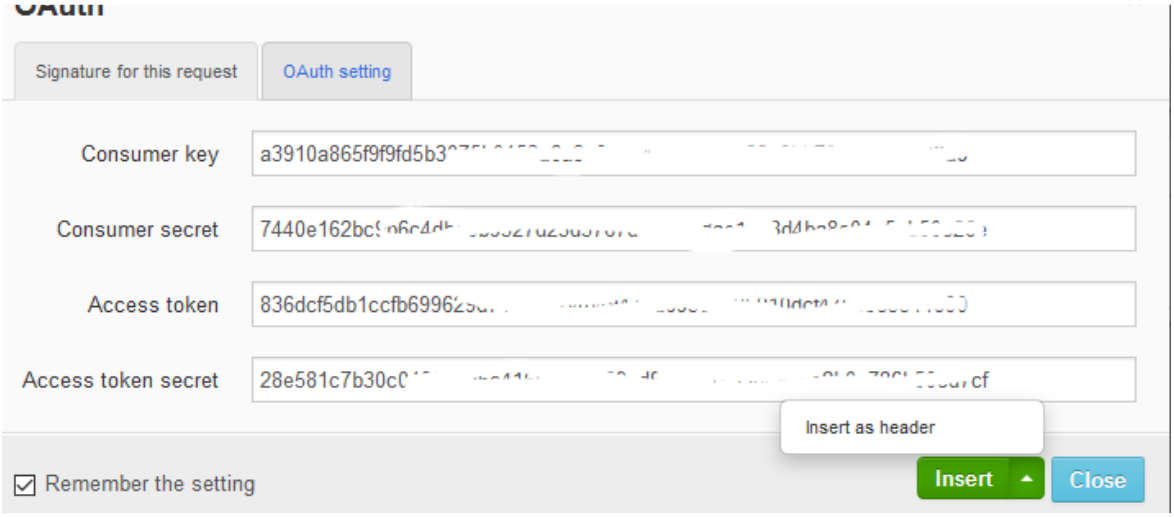
A. Signature Methods shall be set to HMAC-SHA1
B. oAuth Version set to 1.0
C. Realm is the NS Account ID. Can be found in NS under Setup -> Integration -> Web Service Preferences:



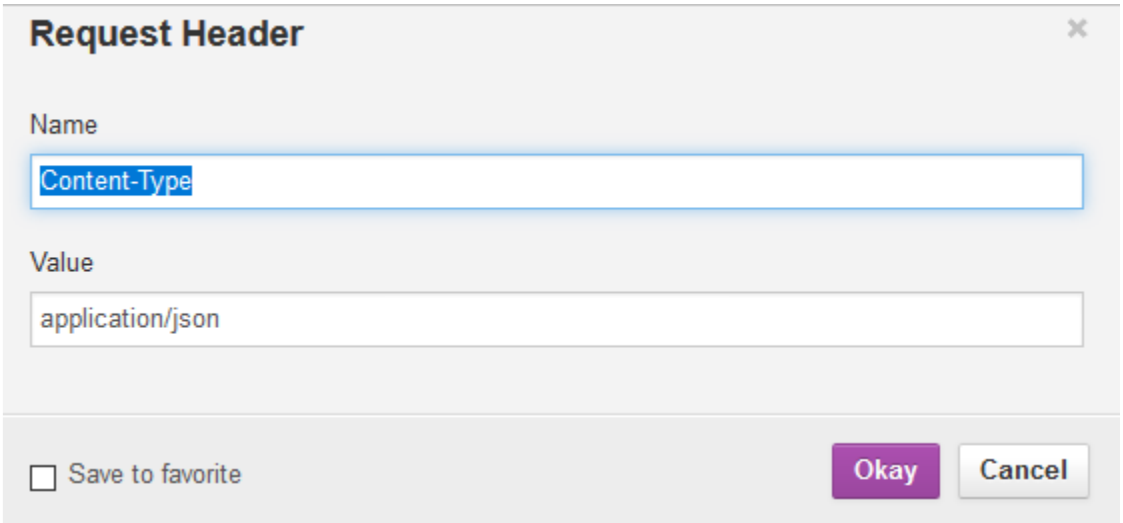
If Web Service is not available it can be activated by navigating to Setup -> Company –> Enable Features -> Subtab SuiteCloud -> Web Services.

D. oAuth Nonce: Auto
E. oAuth Timestamp: Auto

After we have filled in all the fields we can insert the signature as a header:

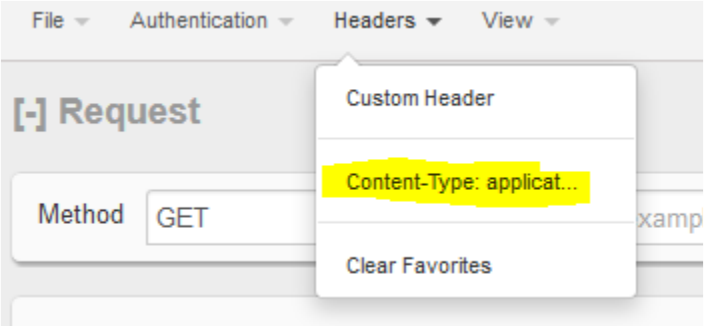


Answer Yes to the question 'Do you want RESTClient to refresh oAuth signature before sending your request'.
Next step is to add another header in the request. We navigate to Headers – Custom Headers and fill in the fields as the following:

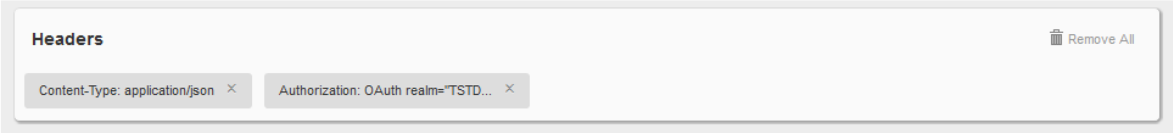


Name: Content-Type
Value: application/json

We click on Okay and we see the new custom header under Headers:



We click the highlighted header to insert it in our request's header. Now we can see two different objects in our header.



Depending on which RESTlet request we want to test we have to change the request body and the URL.

Fetch Invoices

The parameters below (marked with Bold) must be sent in the body of the request. A short explanation follows per each.

{

"type":"Invoice", - Instead of Invoice we can write the name of the post in the custom record which points to the search returning the results.

"size":"10", -Number of records/transactions we want to fetch.

"includeColumnDescription":"T", – An extra row in the JSON object in the answer is shown. Returns the custom label text excluding the IDs.

"includeTextualValue":"F", – Set to F and do not change. Not used at the moment.

"includeColumnName":"T", – Standard field names in NS are shown in the answer.

"includeSeparateColumnInformation":"T", – Separate information about all fields is shown in the answer's header.

"includeInvoicePDF":"T", – Fetch invoice PDF saved on INV 77. ONLY applicable if IAD bundle is installed, to be removed otherwise.

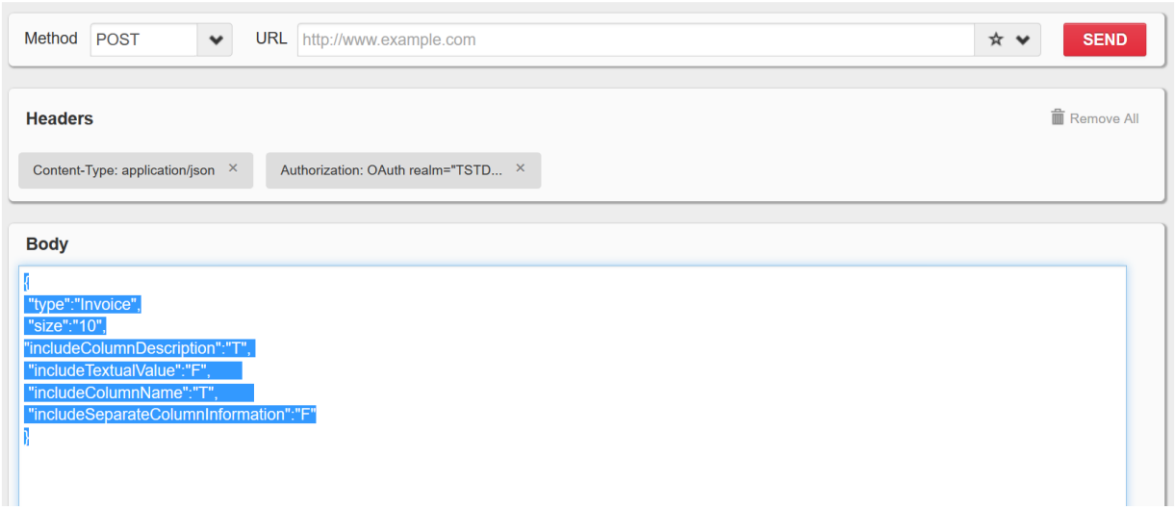
"showcomponents":"F"- Includes components (rows) of item groups. Otherwise only the start row for the item group will be sent.

}

Example, ready to paste in the client:

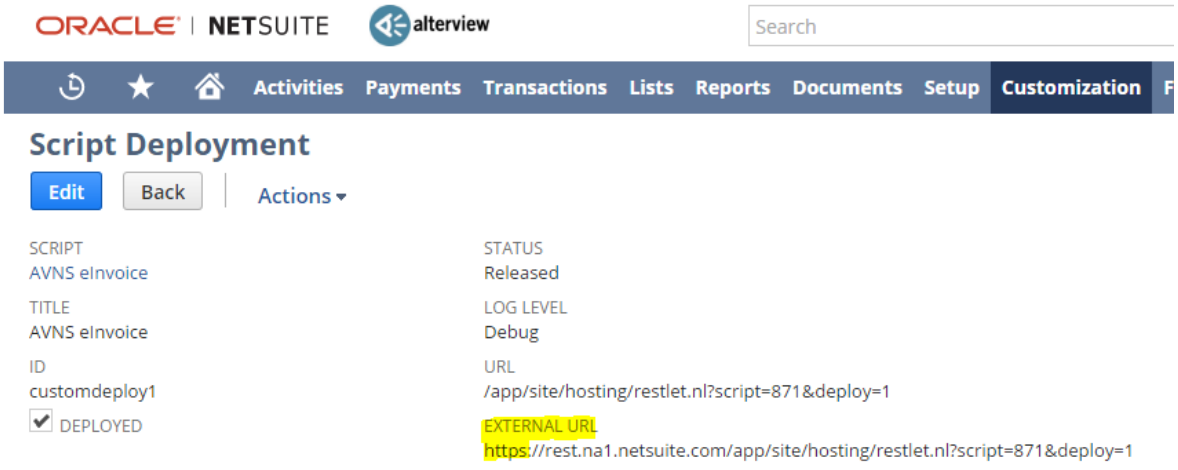
```
{
 "type":"Invoice",
 "size":"1",
"includeColumnDescription":"T",
 "includeTextualValue":"F",
 "includeColumnName":"T",
 "includeSeparateColumnInformation":"F",
"includeInvoicePDF":"T",
"showcomponents":"F"
}
```


We insert the above into the request's body in RESTClient:

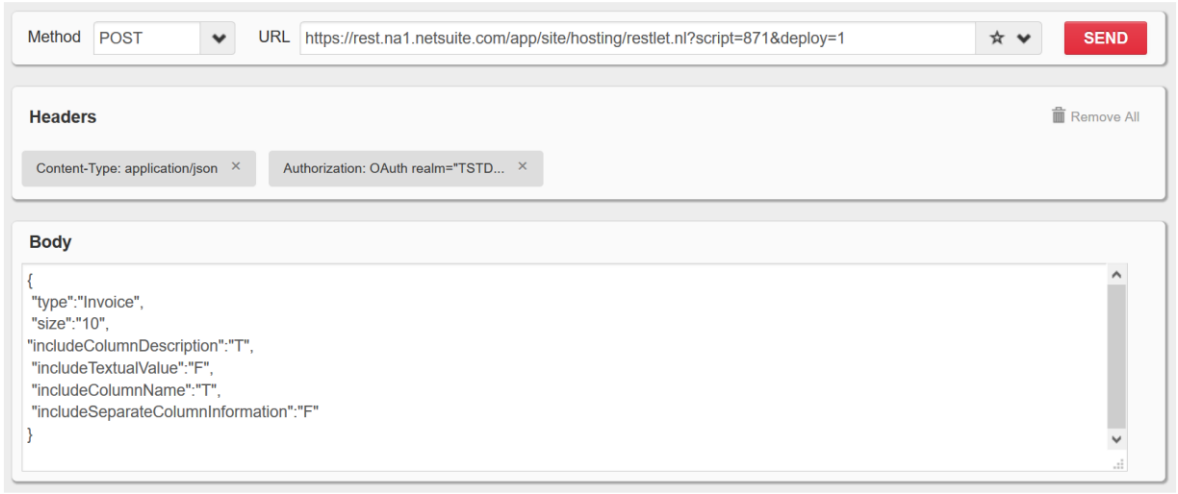


As next step we need to complete the request with the URL. We can find that in NS as it is part of the bundle Get NetSuite Record. Navigate to Customization -> Scripting -> Scripts. Locate thereafter the script called AVNS eInvoice. Open it and go to Deployments subtab. There should

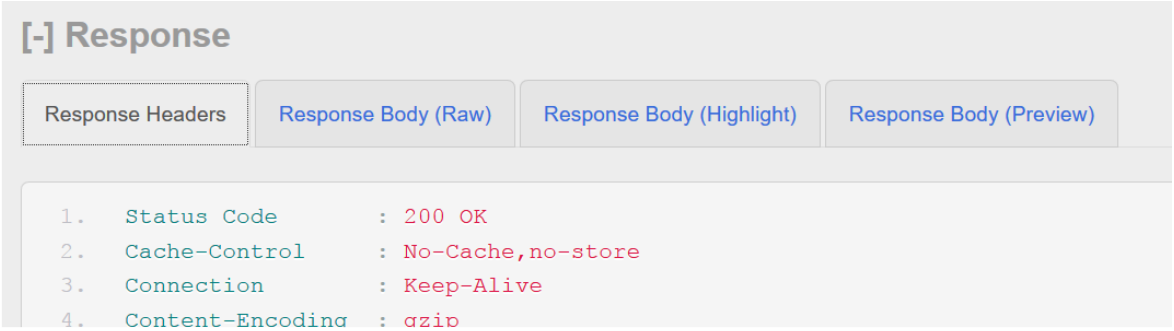
be a deployment with the same name as the script. Click on it and then copy the value under ‘External URL’:



Paste the copied External URL into the URL field in the Request header. Method must be set to POST. The complete request should look as the following:



Before clicking on Send make sure the search is returning result. We can make sure of this by changing the value of the field Invoice Delivery Method to eInvoice on some transactions. After running the request to fetch invoices Status Code in the Response Headers should be ‘200 OK’ if the request was correct. But this is not a confirmation that any records were fetched.



In order to see the fetched records in JSON format we click on the tab Response Body (Highlight).

If no records are fetched we will see 'Records: 0' and no JSON structure.

Acknowledge Invoices

This RESTlet request will populate the custom transaction body fields Acknowledge Timestamp and Acknowledge Log on the transactions. The request body must contain the following parameters:

{

"type":"Invoice", - acts the same way as fetching data, should be the same as the request for fetching invoices.

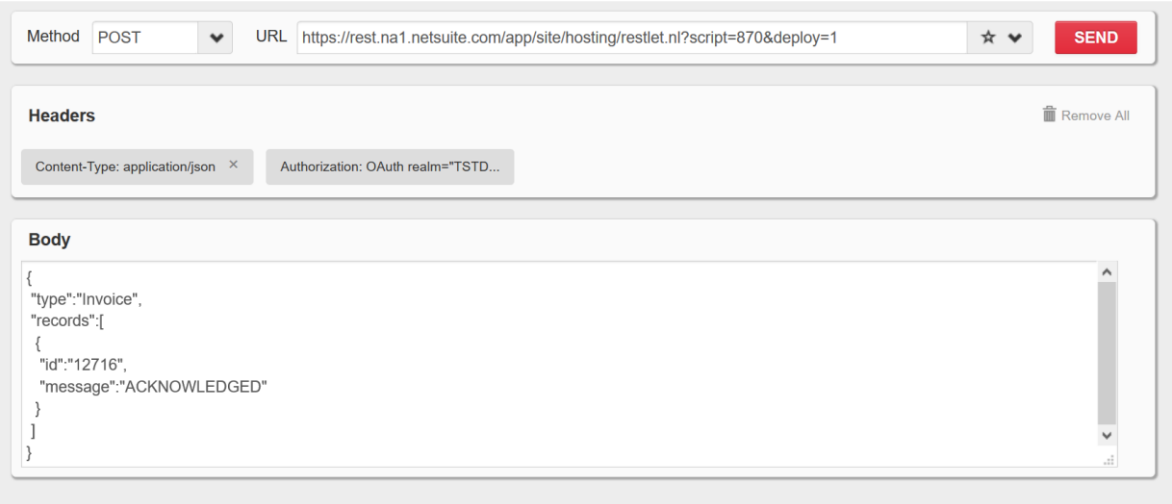
"records":[

{"id":"xxxx", - We type in the specific internal ID of a transaction instead of "xxxx"

"message":"Acknowledged" – Instead of 'Acknowledged' we can type another string which will show in Acknowloedge Log Custom transaction Body Field. The timestamp is populated automatically.

}
]
}
Example:
{
"type":"Invoice",
"records":[
{
"id":"36083",
"message":"ACKNOWLEDGED"
}
]
}

We need then change the URL. For this we need to get the External URL from the deployment AVNS eInvoice Acknowledge. We follow the same steps as for fetching invoices until we get to the list of scripts. Find there and click on AVNS eInvoice Acknowledge. We move on to the deployments subtab and copy the External URL which we will later paste into the RESTclient URL for the request. Method here is also POST as in fetching invoices. The request to acknowledge the records should look as the following:



Status Code in the Response Headers needs to be 200 OK even here, just as in fetching invoices. The body has however a different structure. It basically shows only a confirmation that the request was succeeded.

[-] Response

Response Headers | Response Body (Raw) | Response Body (Highlight) | Response Body (Prev


```
{
    "status": "OK",
    "details":
    [
    ],
    "elapsedTime": "2017-03-22T13:56:43.967Z",
    "records":
    [
        {
            "id": "12716",
            "status": "OK",
            "details":
            [
            ]
        }
    ]
}
```


To double check we find the transaction in NetSuite and locate the fields Acknowledge Timestamp and Acknowledge log to see if the first has been populated with a timestamp and if the second has the string in the parameter “message” we wrote in the request.

PLEASE NOTE: As there are some parties involved, the final data sent could slightly differ from the one being sent from Netsuite.