

## Web Forms API

The Web Forms API allows an external system to get data from a Futuro [ad hoc report](#). For example, an external system needs to get charge element data from Futuro. An ad hoc report is created to retrieve this data. The system administrator of the external system sends a request to Futuro for the report. Futuro authenticates the request and sends the report data.

To authenticate the user who is requesting the data from Futuro, the Web Forms API can be configured to use **BASIC**, **TOKEN\_BASED**, or **IDP\_TOKEN** authentication. With BASIC authentication, the external system provides a Login Name, Password, and API Key to authenticate every request it sends to Futuro. With TOKEN\_BASED authentication, the external system must first provide a Login Name, Password, and API Key to log into Futuro. Futuro will then issue a token that the external system must include with all requests it sends to Futuro. With IDP\_TOKEN authentication, the external system must provide an authenticated JWT session token issued by a third-party Identity Provider (IDP) with every request it sends to Futuro.

Note: You can also view an ad hoc report by opening the report's REST URL in a web browser. This method uses BASIC authentication with only a username and password. See [View the Ad Hoc Report Data in a Web Browser](#) for more information.

Configuration Steps:

[Create the Ad Hoc Report](#)

[Configure the web\\_services\\_authentication\\_method](#)

[IDP\\_TOKEN – Configure Certificate and Keystore](#)

[Configure the Interface Host](#)

[Configure the web\\_services Server Settings](#)

[Create the Report URL](#)

[Create and Send the Request](#)

See Also:

[Ad Hoc Reports Feature](#)

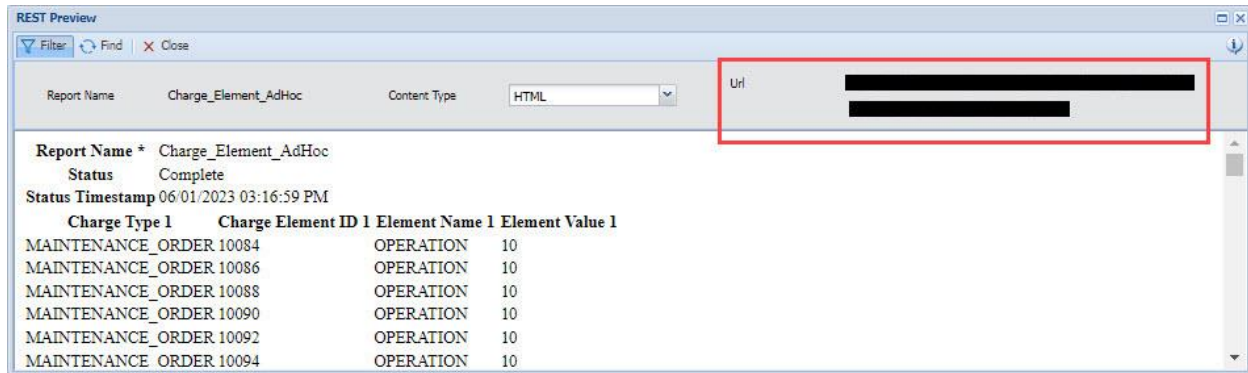
[Server Settings](#)

[Interface Host](#)

## Create the Ad Hoc Report

Use the [Ad Hoc Report Definition form](#) to create the ad hoc report.

Once you have created the report, select it in the Ad Hoc Report Definition form and click the **REST Preview** button. Copy the **URL** from the pop-up form. You will use this URL in the requests you sent to Futuro to get the report data.



## Configure the web\_services\_authentication\_method

The [web\\_services\\_authentication\\_method](#) Server Setting determines how the external system will be required to verify requests from Futuro to get ad hoc report data.

Set the **web\_services\_authentication\_method** setting to BASIC, TOKEN\_BASED, or IDP\_TOKEN for the WEB application.

With BASIC authentication, the external system provides a Login Name, Password, and API Key to authenticate every request it sends to Futuro. With TOKEN\_BASED authentication, the external system must first provide a Login Name, Password, and API Key to log into Futuro. Futuro will then issue a token that the external system must include with all requests it sends to Futuro. With IDP\_TOKEN authentication, the external system must provide an authenticated JWT session token issued by a third-party Identity Provider (IDP) with every request it sends to Futuro.

You will have to configure some additional Server Settings after you configure your Interface Host.

## IDP\_TOKEN Authentication – Configure the Certificate and Keystore

If you are using IDP\_TOKEN as your [web\\_services\\_authentication\\_method](#), you will need to obtain a certificate from your Identity Provider (IDP) and import it into a keystore.

When you save the keystore, take note of its password. You will need to add this to your Interface Host configuration.

The keystore file needs to be placed in the \certificate folder in your [FUTURO\\_HOME](#) folder.

The name of the keystore file and the name of the certificate alias will be used in your Interface Host configuration.

## Configure the Interface Host

Make a **copy** of the RWS\_SERVER Interface Host and save it with a new name.

Make sure the **Connection Type** is set to *Rest* and the **Host Type** is *Receiver*.

The configurations for the Interface Host will depend on your `web_services_authentication_method` setting – [BASIC](#), [TOKEN\\_BASED](#), or [IDP\\_TOKEN](#).

## **BASIC Authentication**

In the [Host Parameters tab](#), add the `GENERATE_API_KEY` parameter. This parameter will create an API Key which you will include as a Header in your request.

You can add the `BASIC_TIMESTAMP_OFFSET_IN_SECONDS` parameter to make the request more secure. When this setting is enabled, Futuro validates that the timestamp of the request from the external system is within this offset period from a timestamp configured in the request's header.

You can add the `AUTHENTICATION_METHOD` parameter (set to BASIC) if you want to override the `web_services_authentication_method` Server Setting.

## **TOKEN\_BASED Authentication**

In the [Host Parameters tab](#), add the `GENERATE_API_KEY` parameter. This parameter will create an API Key which you will include as a Header in the request for the token.

Add the `TOKEN_SESSION_TIMEOUT_IN_MINUTES` parameter and set it to the number of minutes that a token will be valid once Futuro issues it.

You can add the `AUTHENTICATION_METHOD` parameter (set to TOKEN\_BASED) if you want to override the `web_services_authentication_method` Server Setting.

## **IDP\_TOKEN Authentication**

For IDP\_TOKEN authentication, use the Interface Host's Password field to specify the keystore certificate password. This password will be used to obtain the token from the IDP.

In the [Host Parameters tab](#), add the `KEYSTORE_ALIAS` and `KEYSTORE_NAME` parameters. See [Configure the Certificate and Keystore for IDP\\_TOKEN Authentication](#) for information on obtaining these values. These parameters will be used to obtain the token from the IDP.

You can use the `LOGIN_NAME_CLAIM_KEY` parameter to specify which claim attribute in the JWT session token will be used to authenticate a Person record and log into Futuro. By default, the system will use the token's SUB attribute to authenticate the login. If you want to specify a different attribute, enter its name in the `LOGIN_NAME_CLAIM_KEY` parameter.

You can add the `AUTHENTICATION_METHOD` parameter (set to IDP\_TOKEN) if you want to override the `web_services_authentication_method` Server Setting.

## **Configure the web\_services Server Settings**

In addition to the `web_services_authentication_method` setting you configured earlier, you will need to configure the Server Settings shown below to use the Web Forms API to access ad hoc report data.

### 1. Set ad\_hoc\_report\_web\_services\_enabled to TRUE

Set ad\_hoc\_report\_web\_services\_enabled to TRUE for the WEB application.

### 2. Set the web\_services\_receiver\_name

Set the web\_services\_receiver\_name to the Interface Host record you configured earlier. The Interface Host's parameters will be used to authenticate the request for the ad hoc report data.

Note that if your web\_services\_authentication\_method is BASIC and you leave the web\_services\_receiver\_name blank, you can still view the ad hoc report data in a web browser. See [View the Ad Hoc Report Data in a Web Browser](#).

### 3. Configure the web\_services\_https Setting

Set the web\_services\_https Setting Type to TRUE or FALSE for the WEB application.

If this setting is TRUE, then the header of the request that the external system sends to Futuro to get the ad hoc report data must use HTTPS.

## Create the Report URL

When you send the request for the ad hoc report, you will need to use a URL with the following format:

```
http://{host}:{port}/{application}/rest/form/report_adhoc[.{content_type}]?report_name={report_name}[&{filter}]&is_preview=true
```

The REST URL you copied when you [created the ad hoc report](#) can be used as a basis for the request URL.

For example, the following URL is in a request for an ad hoc report called "Person":

```
https://localhost:8080/futuro/rest/form/report_adhoc.html?report_name=Person&is_preview=true
```

**http/https** – The URL will begin with *https* if the web\_services\_https Server Setting is TRUE.

**host** – The host name of the application server.

**port** – The port of the application server.

**application** – The name of the application (futuro).

**content\_type** – Format that determines how the report data will be displayed (html, jsn, xhtml, xsd, xml).

**report\_name** – The name of the report as defined in the Ad Hoc Report Definition form.

**filter** – If the report includes filters and you want to apply these filters to the report data, add the column name of each filter separated by an ampersand (&). For example, &person\_num=1101 means the person\_num filter set to "1101" will be used to display the report data.

**Example – URL for the html report named “Person” from localhost on port 8080:**

`http://localhost:8080/futuro/rest/form/report_adhoc.html?report_name=Person&is_preview=true`

**Example – URL for the html report named “Action” from localhost on port 8080 with the person\_num column filter set to 1102:**

`http://localhost:8080/futuro/rest/form/report_adhoc.html?report_name=Action&person_num=1102&is_preview=true`

**Formatting the Date Filter in a Report URL**

If your report URL includes a date or timestamp filter, you will need to format the date you enter. The date format must match the [rest\\_date\\_format](#) or [rest\\_datetime\\_format](#) Application Settings. If the date/time you enter in the filter is not formatted according to these settings, the date filter will not be used to obtain the report data.

For example, this URL is a request for an ad hoc report called Action\_Report that includes a filter for the Posting Date:

`http://kasgvm:8080/futuro/rest/form/report_adhoc.html?report_name=Action_Report&posting_date=`

You want to view the report data for October 6, 2022 and the `rest_date_format` setting is `yyyy-MM-dd`. Your URL would be:

`http://kasgvm:8080/futuro/rest/form/report_adhoc.html?report_name=Action_Report&posting_date=2022-10-06&is_preview=true`

**Create and Send the Requests**

The configuration and steps for sending a request for an ad hoc report will depend on whether you are using BASIC, TOKEN\_BASED, or IDP\_TOKEN [authentication](#).

The screenshots below show requests being sent from an application called Postman. If you are using a different application to configure and send requests, your screens will be different.

**BASIC Authentication**

This section explains how to configure and send a request for an ad hoc report using BASIC authentication.

Make sure the authentication method is set to BASIC in your `web_services_authentication_method` Server Setting or the `AUTHENTICATION_METHOD` parameter for your Interface Host.

The `web_services_receiver_name` must be set to the Interface Host you are using.

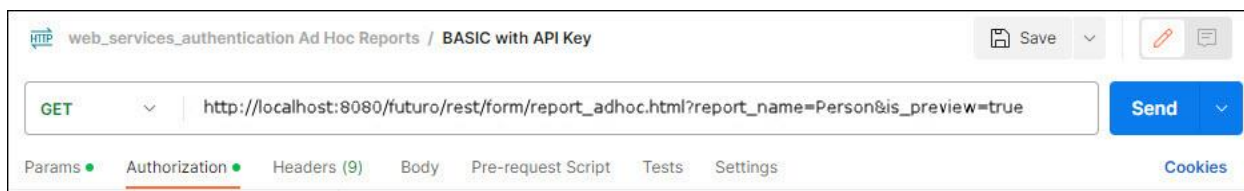
**Create the Request for the Report**

You will need to configure a GET request to retrieve the report data from Futuro.

The request will use the URL you created [earlier](#).

**Example:**

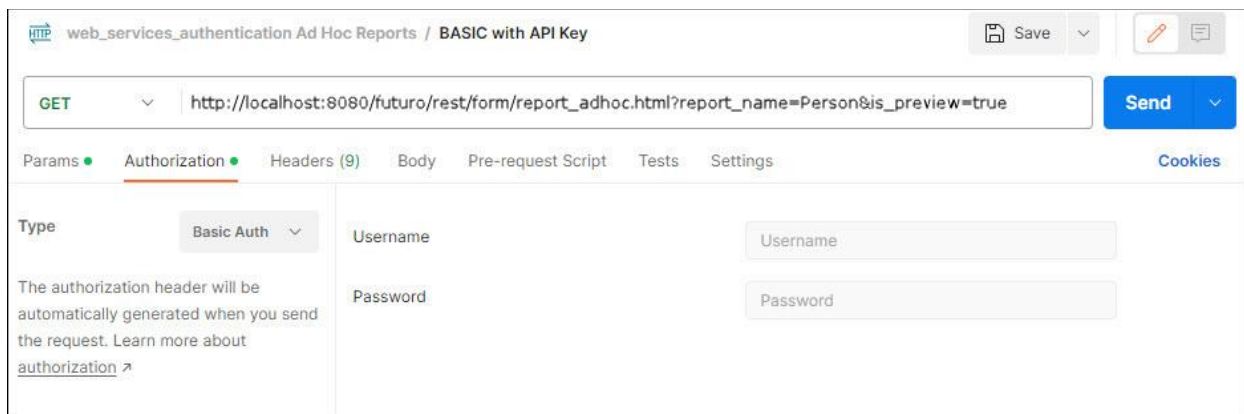
`http://localhost:8080/futuro/rest/form/report_adhoc.html?report_name=Person&is_preview=true`



The screenshot shows the Postman interface with the URL bar set to `http://localhost:8080/futuro/rest/form/report_adhoc.html?report_name=Person&is_preview=true`. The **Authorization** tab is selected, and the **Basic Auth** type is chosen. The Username and Password fields are empty.

In the **Authorization** section, enter the username and password that will log into Futuro. The **Type** should be *Basic Auth*.

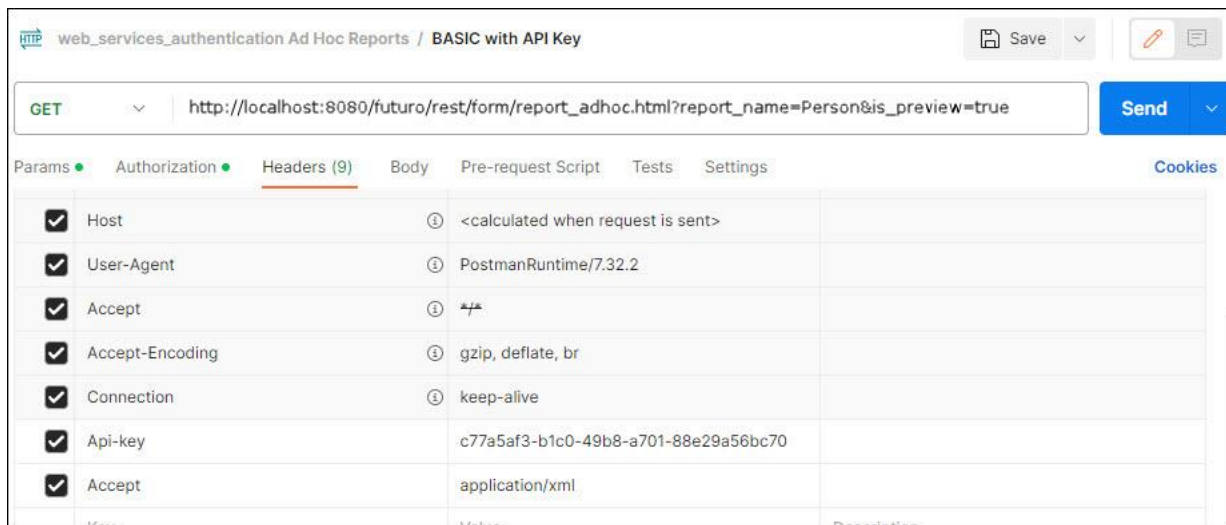
Make sure you enter a valid Login Name and password defined in the Person form. The person will need to have access to the Ad Hoc Reports form in their Security Role.



The screenshot shows the Postman interface with the **Authorization** tab selected. The **Basic Auth** type is chosen. The Username and Password fields are empty.

In the **Headers** section, add a record for the **Api-key** and set the value to the Interface Host's API Key. The API Key comes from the GENERATED\_API\_KEY parameter for the Interface Host.

You also need a Header record for the **Accept** key set to `application/xml`.



The screenshot shows the Postman interface with the **Headers** tab selected. The following headers are defined:

Key	Value	Description
Host	<calculated when request is sent>	
User-Agent	PostmanRuntime/7.32.2	
Accept	*//*	
Accept-Encoding	gzip, deflate, br	
Connection	keep-alive	
Api-key	c77a5af3-b1c0-49b8-a701-88e29a56bc70	
Accept	application/xml	

## Send the Request

When you send the request successfully, Futuro sends back a response with the report data.



Report Name *	Person
Status	
Status Timestamp	
Person Num	First Name
Last Name	
ADMIN	Admin
PORTAL_TEMPLATE_USER	PORTAL_TEMPLATE_USER
IMPORT_USER	IMPORT
1001	Stevie
1002	Sonia
2000	Vince
2001	Vicky
3000	John
170	Steve

## TOKEN\_BASED Authentication

This section explains how to configure and send a request for an ad hoc report using TOKEN\_BASED authentication. You will need to configure (1) a POST request to log into Futuro and obtain the token and (2) a GET request to obtain the report data.

Make sure the authentication method is set to TOKEN\_BASED in your web\_services\_authentication\_method Server Setting or the AUTHENTICATION\_METHOD parameter for your Interface Host.

The web\_services\_receiver\_name must be set to the Interface Host you are using.

## Request for the Token

You will need to create a POST request to log into Futuro and obtain a token. This token will be used in the GET request for the report data.

To create the token request, use the following URL. The URL will begin with *https* if HTTPS is enabled for the Restful Web Service application.

*http://<application server>:<port number>/IHTTPLCE/REST/login*

Example: *https://qap-asgreh8:8543/IHTTPLCE/REST/login*

## Headers

You need to configure the following Headers in your token request:

**Content-Type:** The Content-Type Header indicates the format of the data being sent in the request. Set the Content-Type to application/xml, or application/json. The Content-Type should match the format of the Body of the request.

**Api-key:** The API Key is used to authenticate the request. The API Key comes from the [GENERATED API KEY parameter for the Interface Host](#).

The screenshot shows the Postman interface for a POST request to `http://localhost:8080/futuro/rest/form/login`. The 'Headers' tab is selected, showing 10 headers. The headers are as follows:

Key	Value	Description
Postman-Token	<calculated when request is sent>	
Content-Type	text/plain	
Content-Length	<calculated when request is sent>	
Host	<calculated when request is sent>	
User-Agent	PostmanRuntime/7.32.2	
Accept	*/*	
Accept-Encoding	gzip, deflate, br	
Connection	keep-alive	
Api-Key	a64b0022-731e-446d-8662-0c02152a23f8	
Content-Type	application/xml	

## Body

The Body of the request must contain the information shown below. You can use the XML or JSON version. The username and password are the Login Name and Password from the [Interface Host you configured earlier](#).

### XML format:

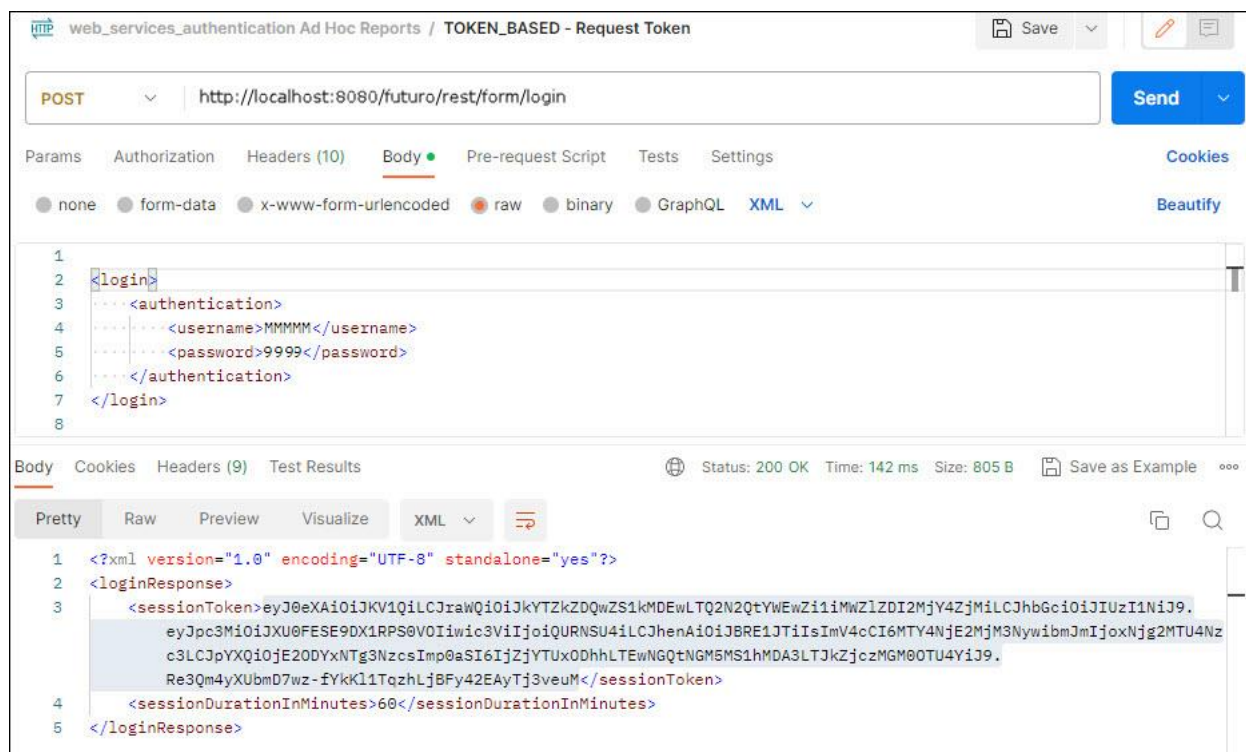
```
<login>
  <authentication>
    <username>futuro-user</username>
    <password>futuro-pwd</password>
  </authentication>
</login>
```

### JSON format:

```
{
  "authentication": {
    "username": "futuro-user",
    "password": "futuro-pwd"
  }
}
```



When the request is sent successfully, the response will include a token. You need to copy this token so you can include it in the request for the report data.



The token will be valid for the number of minutes specified in the `TOKEN_SESSION_TIMEOUT_IN_MINUTES` parameter in your Interface Host.

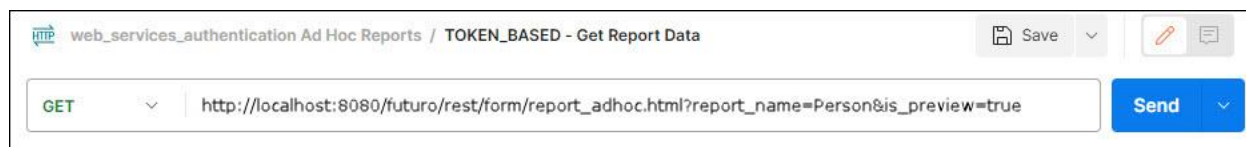
## Request for the Report

You will need to configure a GET request to retrieve the report data from Futuro.

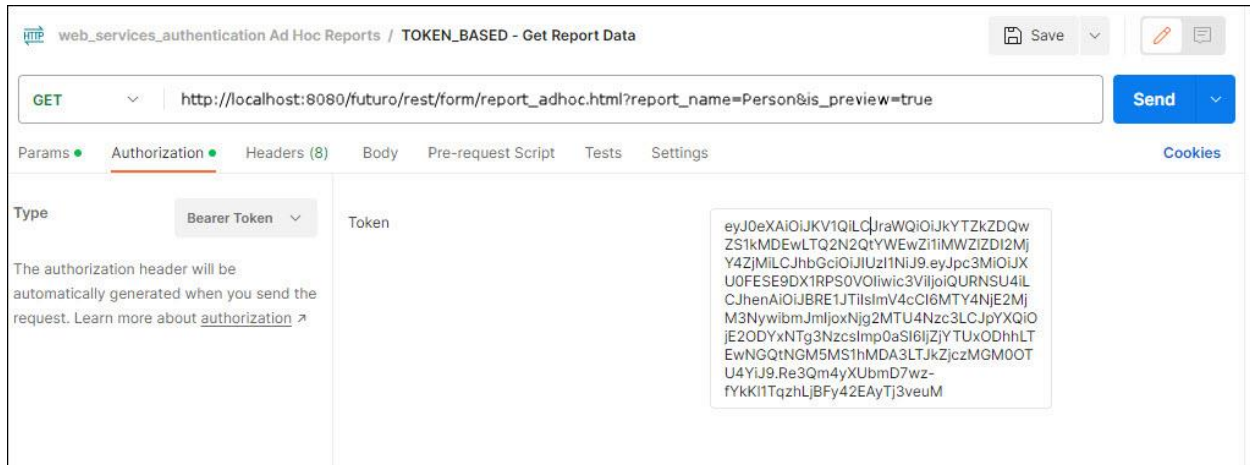
The request will use the URL you created [earlier](#).

**Example:**

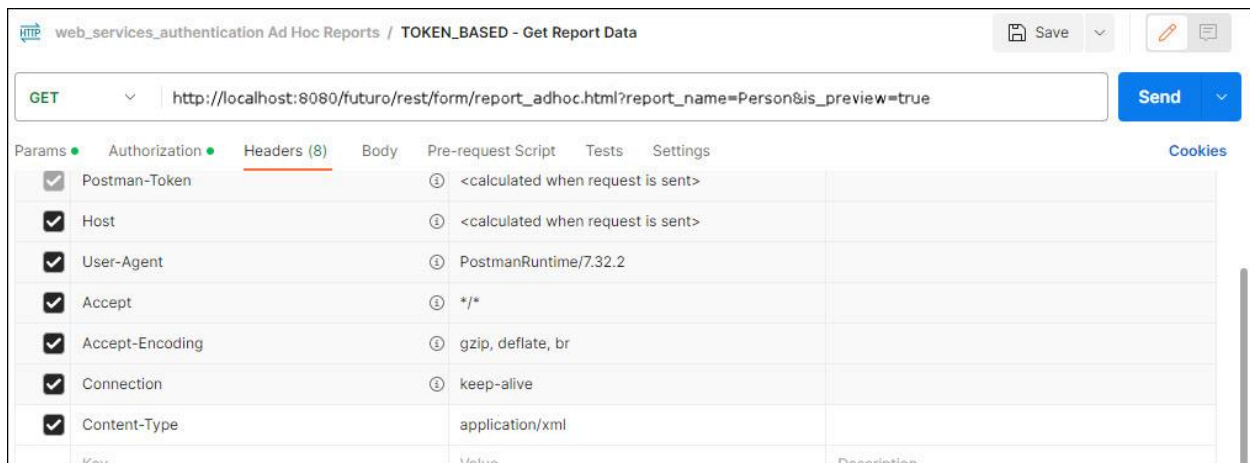
`http://localhost:8080/futuro/rest/form/report_adhoc.html?report_name=Person&is_preview=true`



In the **Authorization** tab, set the Type to *Bearer Token* and enter the token you obtained earlier.



In the **Header** tab, add a record for Content-Type and set the value to application/xml.



When you send the request successfully, Futuro sends back a response with the report data.

Body Cookies Headers (10) Test Results				
200 OK 4.75 s 7.17 KB Save as Example				
Pretty Raw Preview Visualize				
Report Name * Person				
Status				
Status Timestamp				
Facility *				
Person Num	First Name	Last Name	Facility	
ADMIN	Admin	User		
PORTAL_TEMPLATE_USER	PORTAL_TEMPLATE_USER	User		
IMPORT_USER	IMPORT	User		
1001	Stevie	Contractor	Pembroke Pines	
1002	Sonia	Contractor	Pembroke Pines	
2000	Vince	Contractor	Pembroke Pines	
2001	Vicky	Contractor	Pembroke Pines	
3000	John	Contractor	Pembroke Pines	
170	Steve	Pines	Pembroke Pines	

## IDP\_TOKEN Authentication

This section explains how to configure and send a request for an ad hoc report using IDP\_TOKEN authentication.

You will need to obtain the token from the IDP and use this token in your request to get the report data from Futuro.

Make sure the authentication method is set to IDP\_TOKEN in your web\_services\_authentication\_method Server Setting or the AUTHENTICATION\_METHOD parameter for your Interface Host.

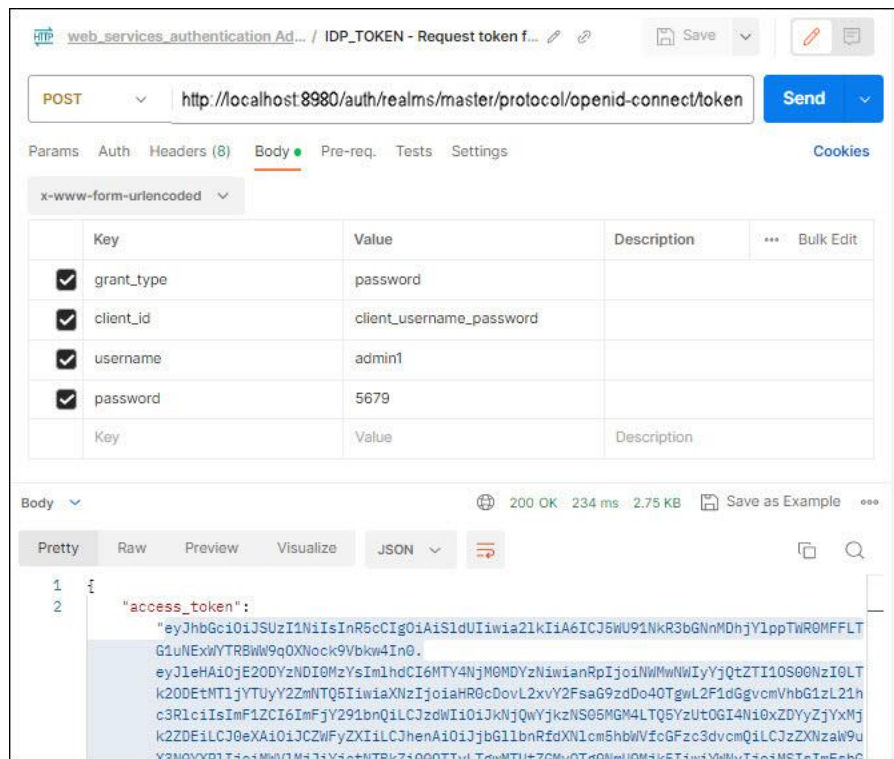
The web\_services\_receiver\_name must be set to the Interface Host you are using.

### Obtain the Token from the IDP

You will need to obtain an authenticated JWT session token from your Identity Provider (IDP). This token must be included with every request you send to Futuro.

The method for obtaining this token will depend on which IDP you are using.

In the illustration below, a request is sent to a Red Hat SSO IDP and the token is returned. This request uses the Password configured in the Interface Host, as well as the KEYSTORE\_ALIAS and KEYSTORE\_NAME Interface Host Parameters, to obtain this token.



## Configure the Person Login Name

Once you have obtained the token (see above), you need to find the claims in the token and use the value of one of these claims to authenticate a person in Futuro.

In the illustration below, an IDP token and its claim attributes are shown.



### LOGIN\_NAME CLAIM KEY Host Parameter

## Create the Request for the Report

After you obtain the token and configure the Person Login Name, you can create the GET request to retrieve the report data from Futuro.

The request will use the URL you created [earlier](#).

**Example:**

http://localhost:8080/futuro/rest/form/report\_adhoc.html?report\_name=Charge\_Element\_AdHoc&is\_previ  
ew=true

web\_services\_authentication Ad Hoc Reports / IDP\_TOKEN - Get Report Data

GET http://localhost:8080/futuro/test/form/report\_adhoc.html?report\_name=Charge\_Element\_AdHoc&is\_preview=true

Params Authorization Headers (8) Body Pre-request Script Tests Settings Cookies

The [token you received from the IDP](#) needs to be placed in the **Authorization** section. The Type should be *Bearer Token*.

[illegible]

In the request's **Headers**, add a record for the **Content-Type** key and set it to `application/xml`.



web\_services\_authentication Ad Hoc Reports / IDP\_TOKEN - Get Report Data

GET  Send

Params • Authorization • **Headers (8)** Body Pre-request Script Tests Settings Cookies

Headers Hide auto-generated headers

	Key	Value	Description	...	Bulk Edit	Presets
<input checked="" type="checkbox"/>	Authorization	Bearer eyJhbGciOiJSUzI1NiIsInR5cCIgOiAiSldU...				
<input checked="" type="checkbox"/>	Postman-Token	<calculated when request is sent>				
<input checked="" type="checkbox"/>	Host	<calculated when request is sent>				
<input checked="" type="checkbox"/>	User-Agent	PostmanRuntime/7.32.2				
<input checked="" type="checkbox"/>	Accept	*/*				
<input checked="" type="checkbox"/>	Accept-Encoding	gzip, deflate, br				
<input checked="" type="checkbox"/>	Connection	keep-alive				
<input checked="" type="checkbox"/>	Content-Type	application/xml				

## Send the Request

When you send the request successfully, Futuro sends back a response with the report data.

Body Cookies Headers (10) Test Results 200 OK 4.92 s 10.35 KB Save as Example

Pretty Raw Preview Visualize

**Report Name \*** Charge\_Element\_AdHoc

**Status**

**Status Timestamp**

Charge Type 1	Charge Element ID 1	Element Name 1	Element Value 1
MAINTENANCE_ORDER	10084	OPERATION	10
MAINTENANCE_ORDER	10086	OPERATION	10
MAINTENANCE_ORDER	10088	OPERATION	10
MAINTENANCE_ORDER	10090	OPERATION	10
MAINTENANCE_ORDER	10092	OPERATION	10
MAINTENANCE_ORDER	10094	OPERATION	10
MAINTENANCE_ORDER	10096	OPERATION	10
MAINTENANCE_ORDER	10098	OPERATION	10
MAINTENANCE_ORDER	10100	OPERATION	10