informer

Informer 5 REST API

Updated September 2023

Table of Contents

What is a REST API?

API stands for Application Programming Interface, and it simply refers to the methods that one program can use to communicate with another program. APIs can vary widely, and there are several helpful standards that make it easier to learn how to use a given API. One such standard is the Representational State Transfer (or REST) architecture that Informer utilizes.

In short, Informer's REST API is how other programs can make requests to Informer without requiring a User to interact with Informer through a browser. When referring to Informer's REST API, Informer is considered the server and the program that makes the request is called the client.

Authentication

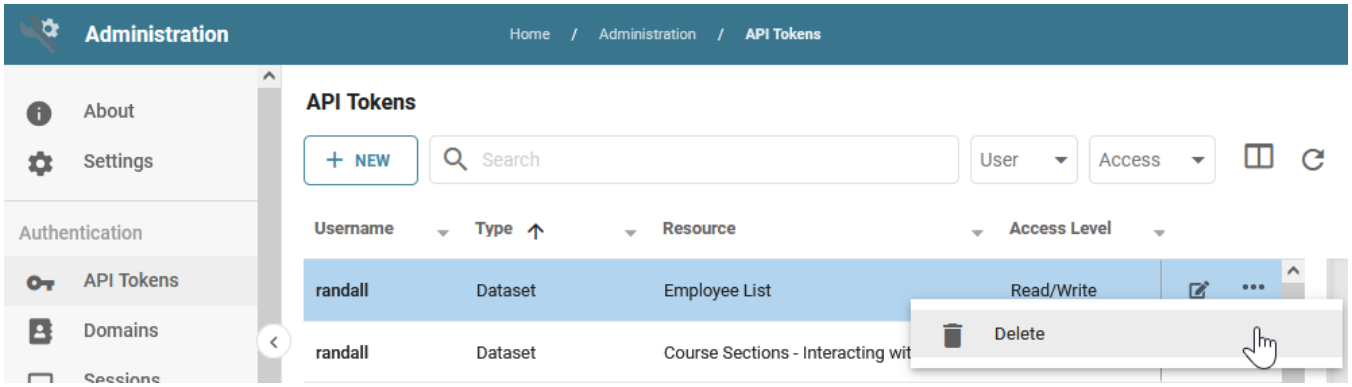
Informer's REST API does require authentication. There are two methods to authenticate when making REST requests to Informer: User and Token authentication.

User Authentication

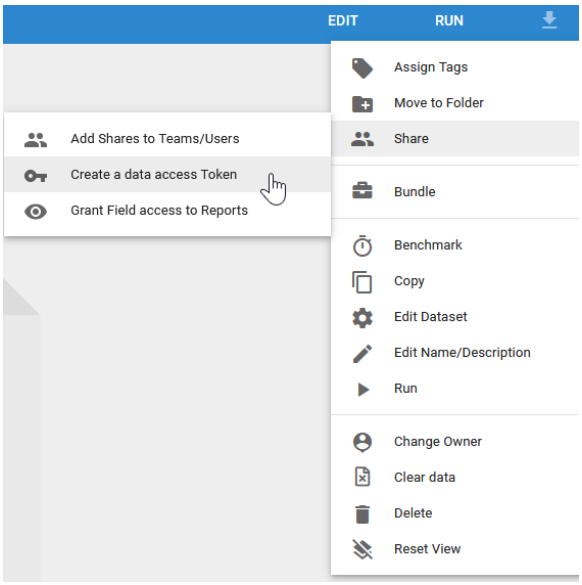
A Local User logs in with Basic Authentication, but Users from Domains may use a different authentication type. In either case, this authentication can be passed along with a REST request. This is typically only used with testing to avoid saving a User's password in a third-party program. It is less secure than the Token authentication and should only be used if the User is prompted each time the request is made.

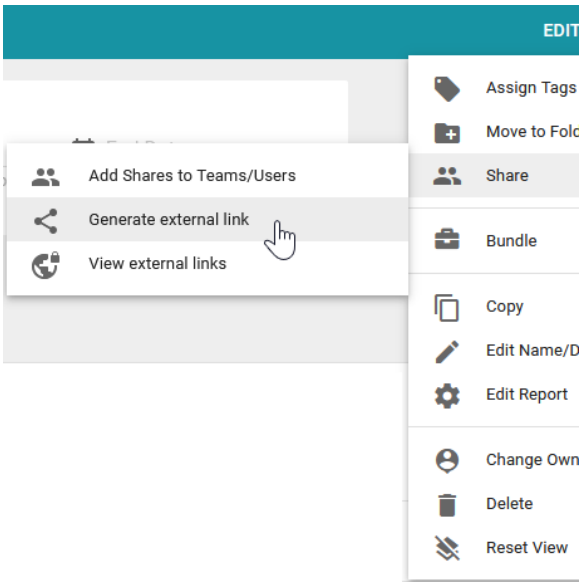
Token Authentication

Token Authentication is a far more secure method of interacting with Informer via the REST API. Tokens are generated in Informer, and each Token can be tracked and even deleted on the API Tokens screen.

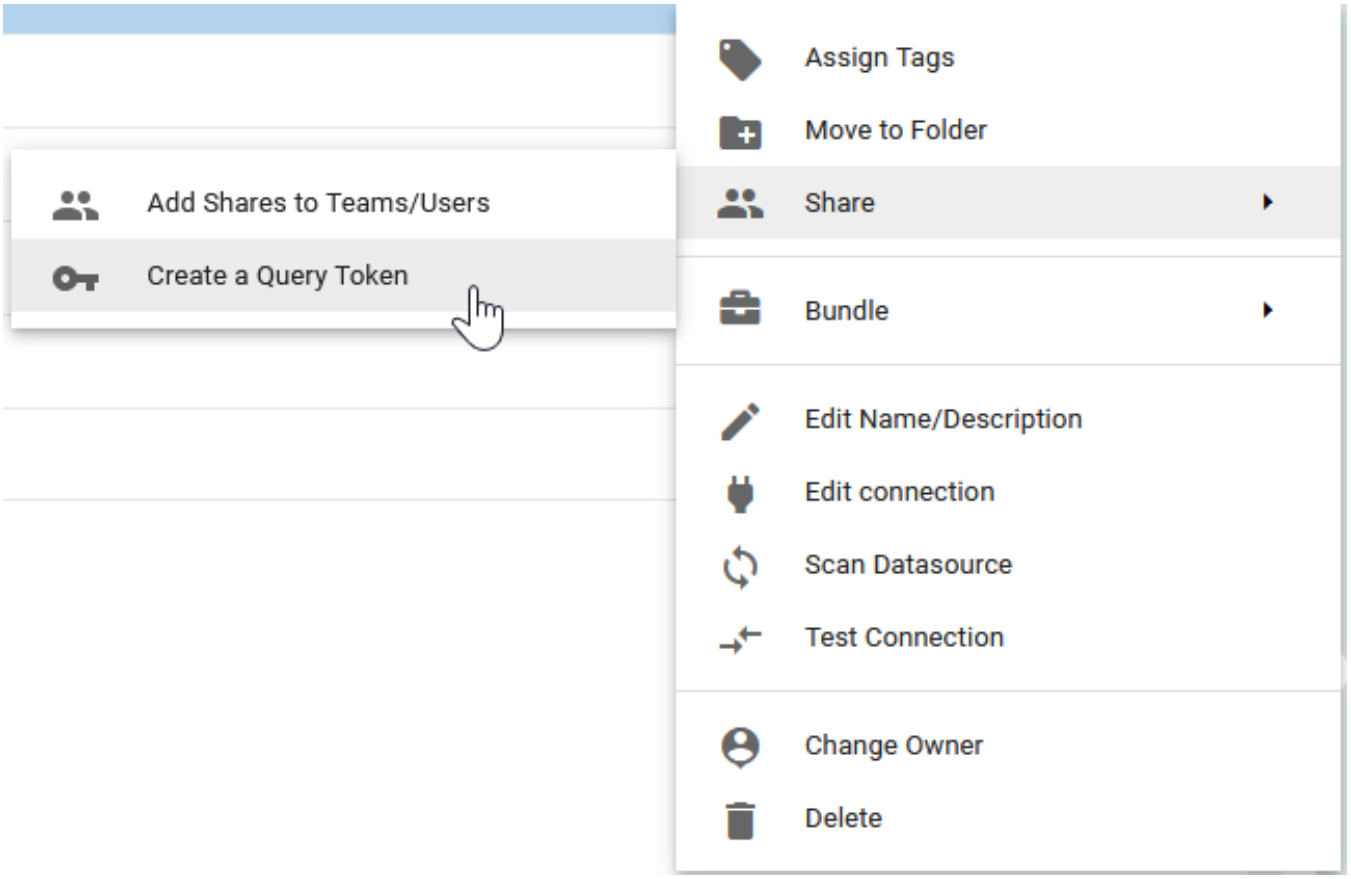


Tokens can also be limited when they are created. The most secure type of Token is resource-specific. When a Publisher shares a Dataset or Report, they can “Create a data access token” or “Generate an External Link”, depending on the resource type.



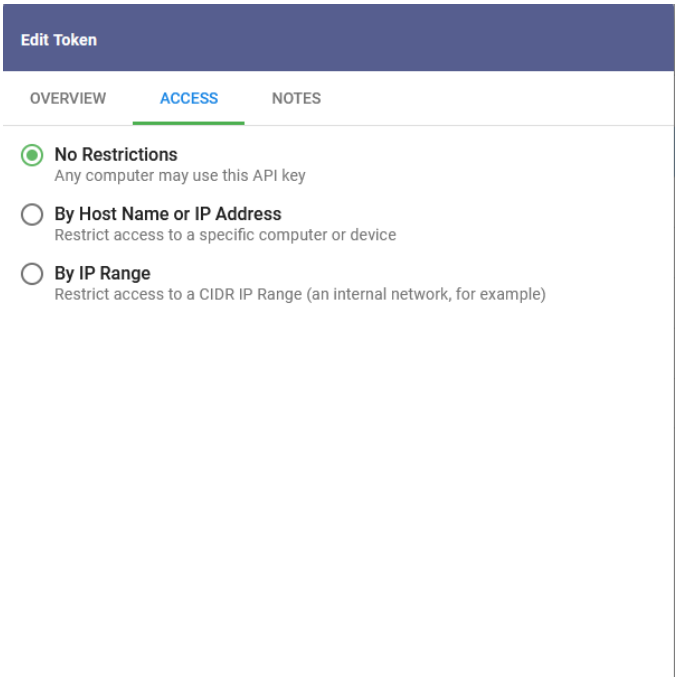


Ad hoc Query Reports and Datasources have a “Create a Query token” option.

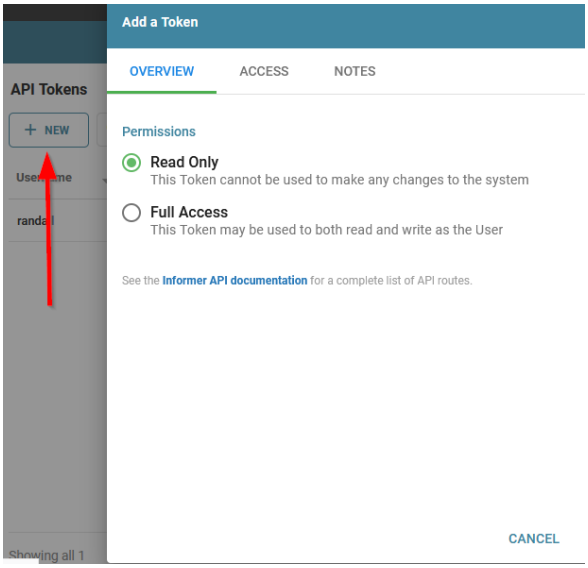


Each of these sharing options creates a Token that can be used to interact with that specific resource. A Token created from one Dataset cannot be used to access the data in another Dataset. This level of segmentation enables external programs to access only the data that they need to function.

Furthermore, each Token can be restricted to only allow requests from a specific host or IP address. This means that a User expecting a third party to make REST requests can provide a Token that works only from the third party's IP address. If the third party leaks the Token, data security can still be preserved.



There is also a Full API Token that is not resource specific. This token grants access to all parts of the system, and it can only be created by a Super User on the API Tokens page. A Full API Token can be created with Read Only permission or Full Access, which effectively makes the holder of that Token a Super User. Use the Full Access option with caution.



Understanding Requests

A REST API accepts requests formatted in a certain way to elicit certain responses. Each request is made up of several different parts: the method, the host (or base URL), the route (or URI), the headers, the parameters (or query - a term that is not typically used in reference to Informer's REST API to avoid conflicting with Informer Queries), and the payload (or body).

The Method

- GET requests simply ask the server for some resource or information. For example, issue a GET request to retrieve the list of Datasets.
- PUT requests usually provide the server with some information and calls an action or replaces some resource with the provided information. For example, issue a PUT request to replace the data in a Dataset.
- POST requests are like PUT requests. The primary difference between the two request types is that it is not usually desirable to issue the same POST request multiple times, but it is safe to do so with a PUT request. For example, issue a POST request to append to the data in a Dataset.
- DELETE requests remove some resource from the server. For example, issue a DELETE request to delete a Dataset.
- PATCH requests update some resource by making a change to it. These requests are far less common than the others.

The Host

The host or base URL is just like the hostname of a website. It is the very beginning of a full URL, such as https://www.example.com or https://informer.example-company.com. This is dependent on each specific instance and sometimes each tenant of Informer, so most examples will not include the hostname in order to be instance-agnostic.

The Route

The route or URI identifies the resource that is the focus of the request. It is in the format of the end of the URL, after the host. The route is independent of the instance of Informer, and so examples will include the full route. Sometimes the route will include a variable that should be replaced, such as:

```
{datasetId}
```

Almost all routes will start with:

```
/api/
```

The Headers

The headers are a set of key and value pairs that provide some information about the request. The most common example of a header is authentication.

The Parameters

The parameters are another set of key and value pairs that provide some information about the desired resource. Parameters are often used to request parts of a resource, such as filtered data or a certain page of a multi-page request.

The Payload

The payload or body of a request is a potentially large chunk of data that can provide a lot of information to the server. A payload is only available with PUT, POST, and PATCH requests, and usually details a change that should be made to a resource. Payloads in requests to Informer's API are typically in JSON format.

Making the Requests

Many programs can make REST requests, including some databases, command lines such as Power Shell and Bash, and even Informer itself (through Web Queries). However, the simplest tools for testing and discovery are dedicated REST clients. The screenshots later use the free version of a REST client called Postman (https://www.postman.com/).

Finding the Right Route

The Documentation Route

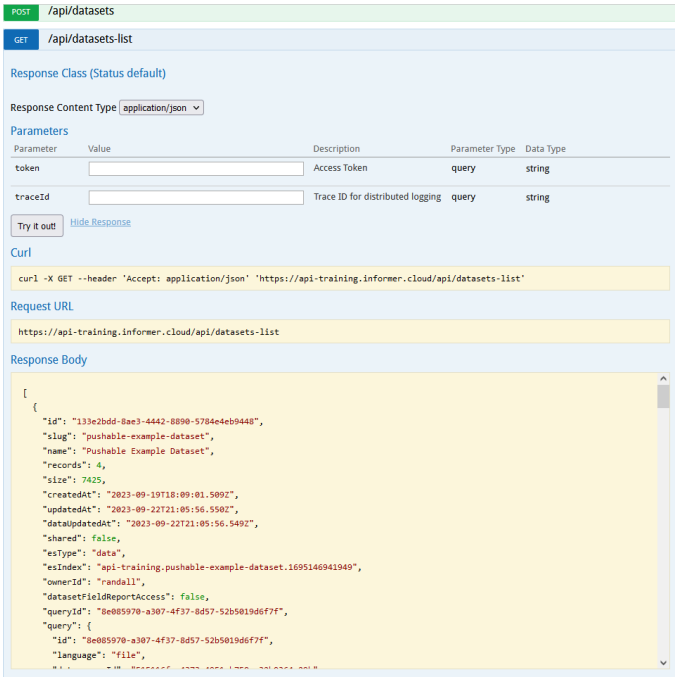
In the browser, append /documentation to the Informer Base URL. For example:

https://api-training.informer.cloud/documentation

This page is automatically generated by a tool called Swagger and lists every available route in Informer.

Click on a route to see additional details and try the route out.

Note: Trying a route in Swagger without providing a Token will use the current session of Informer as its authentication.

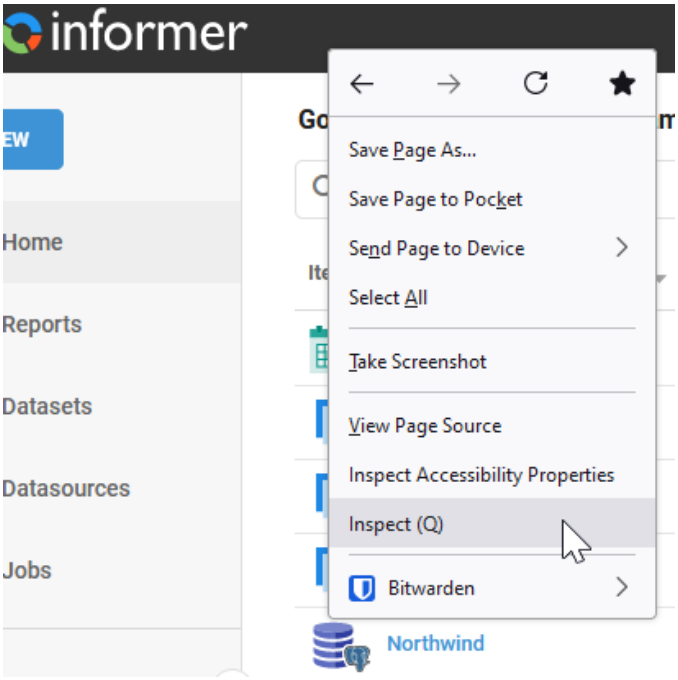


Inspecting Inside the Browser

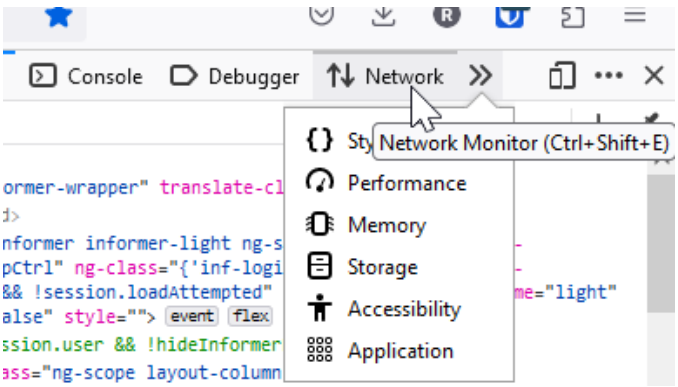
Informer makes REST requests to itself as the User clicks around in the browser. This provides an excellent opportunity to inspect Informer and see what routes are getting called behind the scenes with the browser inspector.

Note: These menus may vary from browser to browser. The following screenshots and instructions were taken from Mozilla Firefox.

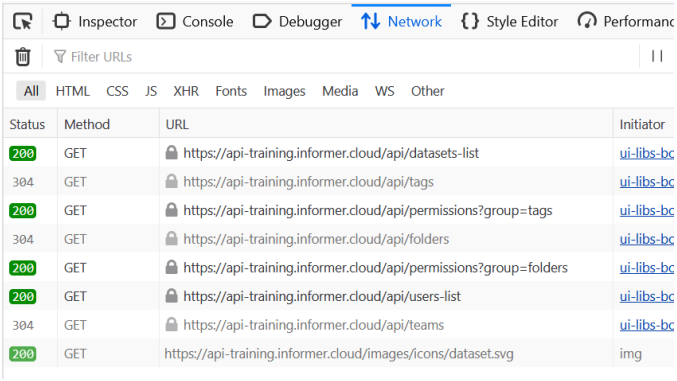
1. Right-click anywhere in the background (not on a button or row of data) of any Informer page and choose Inspect.



2. In the new panel, click the Network tab. It may be under a 'More options' dropdown that looks like two arrows (>>).



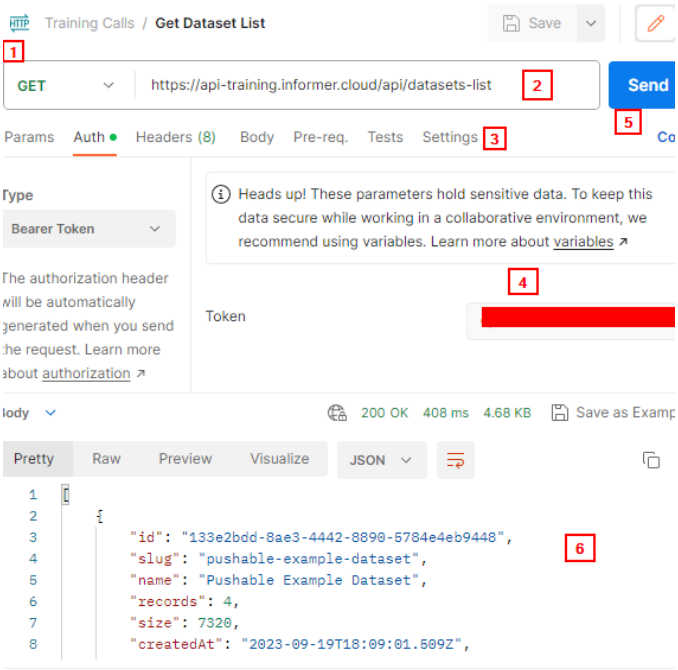
3. Clicking in Informer will cause requests to appear in the network panel.
4. Go to the Dataset Listing page for the first example. Several entries may appear, but one of them will show https://...../api/datasets-list. See the first entry in the screenshot.



5. Click that entry to see additional details of the request. This is the best way to learn to recreate something that a User could do in the browser.

Example 1

To get a list of all Datasets via the REST API in Postman:

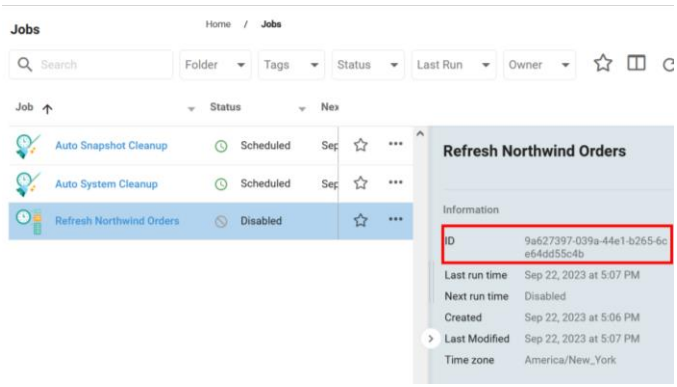


1. Select “GET” for the Method.
2. Enter “/api/datasets-list” as the Host and Route.
3. Various options for the request. This example only needed authentication but notice that there are 8 headers defined automatically. Postman will automatically define some headers, and not all of them may be required.
4. Enter a read-only Full API token (or use the less secure Basic Auth). This example has the token censored with a red block. Take note of Postman’s suggestion about using variables; that is outside the scope of this guide, but it is a useful tool and can assist with security.

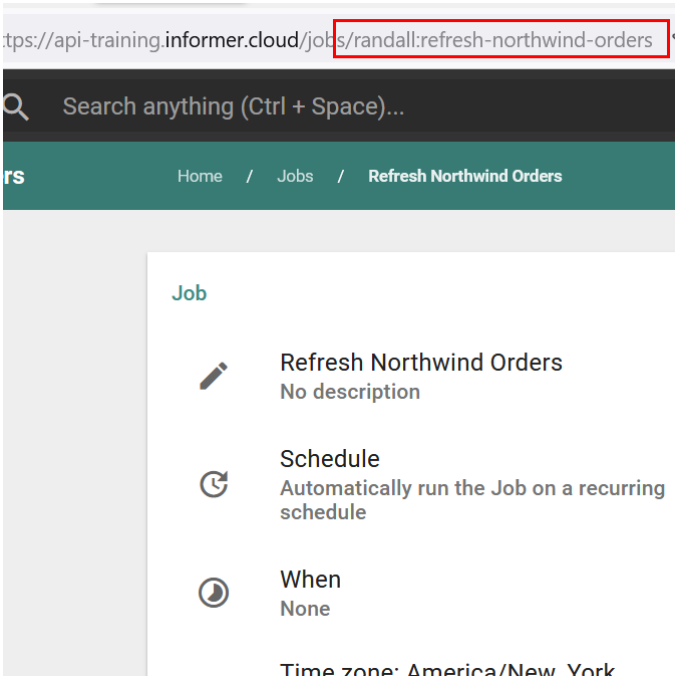
5. Click the “Send” button to send the request to Informer.
6. This is the Response information Informer sends back to the program making the request. In this case, it is a list of all Datasets in the system formatted as a JSON blob. The program could now take that information and use it for the purposes it was designed.

Example 2

To fire an existing Job in Informer, find the Job in Informer and take note of its ID by selecting the Job.

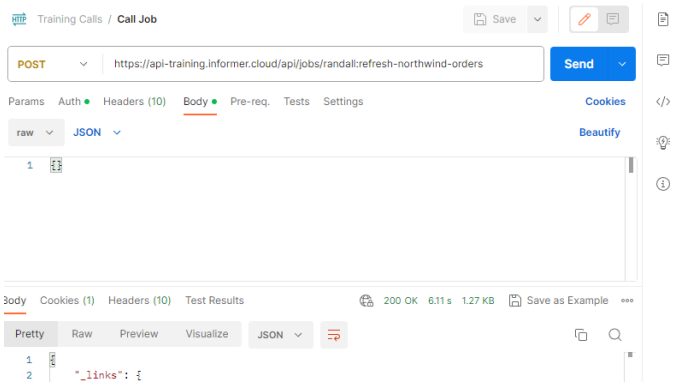


Alternatively, open the job and note its slug in the URL. Most resources in Informer have a slug and an ID. While either one will work for API requests, be aware that changing something’s name or owner can change its slug and break pre-existing REST API requests. A resource’s ID will not change.



Next, build the call with the following information for the request:

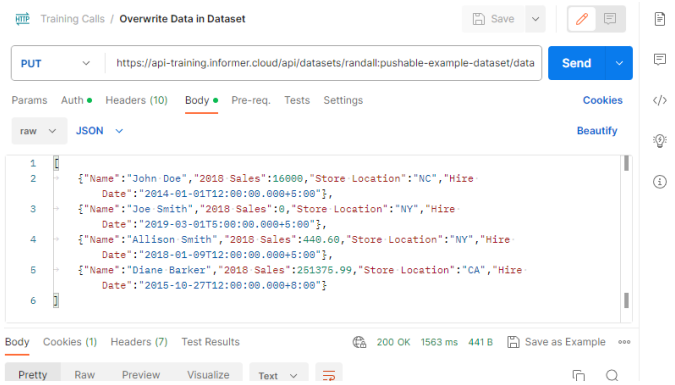
- Method: POST
- Host and Route: /api/jobs/{id}
- Body: {}



While there is technically a payload or body to this call (formatted as a raw JSON object), it is simply an empty JSON object denoted by an open and closed curly brace ({}).

Example 3

Overwriting a Dataset's results with JSON data.
Note: Researching this request through the inspector is slightly more difficult than others. Many Queries in Informer run without exposing all their details to the browser, and so this request is best researched through the /documentation page.



For this example, the request is as follows:

- Method: PUT /api/datasets/{id}/data
- Body:

```
[
{'field1':'value1','field2':'value2'},
{'field1':'value3','field2':'value4'}
```

```
]
```

The payload this time around is still a raw JSON object. It is an array of row-like objects, where each of the keys is a Field alias and each of the values is data for that row.
Note: A successful request has an empty response. Responses on this request will likely provide an error message.

Next Steps

Once a request has been successfully tested in a REST client like Postman, it can be implemented in any number of other client programs. One of the most common use cases is to automate some of these requests on a schedule. Consider building these requests in a Power Shell script and calling the script through Windows Task Scheduler. Similarly, consider building the request with cUrl in a Unix Environment and calling it with a cron job.