REST API data source | WeWeb Documentation

|  |  |
| --- | --- |
| URL source | <https://docs.weweb.io/plugins/data-sources/rest-api.html> |
| Date de scraping | 2025-04-08T15:55:26.171098+00:00 |

## REST API data source ​

## Add plugin ​

To work with external APIs that don't have a native WeWeb integration, you can go to Plugins > Data sources > REST API :  
  
There is no configuration required.

### Images

<https://docs.weweb.io/assets/rest-plugin.DRVwISU-.png>

## Make API calls ​

Once you have added the REST API plugin to a WeWeb project, you will be able to:  
create a data collection that fetches data from a REST API create a workflow that makes an API request

## Security ​

When you make an API call with the REST API plugin in WeWeb the request is made by your application (i.e. client-side request from the frontend, your browser).  
You should never use private API keys in REST API calls made through the client.  
Learn more about public and private API keys .

## Create a collection ​

To get and display data sets that might need to be filtered or paginated, you can create a collection in WeWeb.  
In the example below, we fetched a list of characters from the Rick & Morty API :  
  
You can see that:  
in the Data tab, we clicked on New to create a new collection, we gave it a name and chose REST API as a data source, we configured our API calls as we would in tools like Postman we have options to filter and paginate the collection in the frontend, decide when the collection will be fetched, and if we want it to be preserved on navigation

### Images

<https://docs.weweb.io/assets/rest-collection.BIGAsHtO.png>

## Headers ​

If your data can only be accessed by authenticated users, don't forget to add Authorization headers to your API call and login as a user of your WeWeb app. Otherwise, the API will return an error when you try to create the collection.

## Result key ​

Sometimes, APIs will return the data you need in a nested object. When that happens, it can be helpful to use the Result key field to fetch only the information you need.  
For example, when we get a list of characters from the Rick & Morty API, the API returns two objets:  
info , with info that we could use to setup backend pagination, and results , with a list of Rick & Morty characters.  
If we don't intend to setup backend pagination on that collection, the info object is useless to us and adds uneccessary complexity when working with this data set.  
To access the information inside the results object directly, we can add results as a value in our Result key field:

### Images

<https://docs.weweb.io/assets/rest-collection-result.DAYTJVUw.png>

## API calls in workflows ​

Once you've added the REST API plugin to your project, you will have access to the REST API Request action.  
In the example below, you can see we:  
chose the REST API Request action type, decided to make a POST request, entered the URL or our endpoint, bound the name of two fields to update with values, and configured authorization headers to tell our API who is making the call

### Images

<https://docs.weweb.io/assets/rest-workflow.B9hACfKy.png>

## Resolving CORS Issues ​

When you make an API call in WeWeb the request is made by your application (i.e. client-side request from the frontend, your browser).  
But sometimes, APIs only accept requests that come from a server (i.e. server-side requests from the backend) and return CORS errors. This is the case for the Twitter API for example.  
You can bypass CORS issues by enabling the Proxy the request to bypass CORS issues option on your API call:

### Images

<https://docs.weweb.io/assets/rest-proxy.C9GtmBfE.png>

## Debugging API Requests ​

When you start working with APIs, you'll run into a lot of errors.  
It's completely normal and nothing to worry about 🙂  
WeWeb will display error messages to help you figure out what went wrong.  
The top 4 reasons for errors are:  
you're trying to make an http request when WeWeb only accepts https requests the API you are working doesn't accept client-side requests (see "Resolving CORS issues" section above) you are trying to get or udpate data that is protected in your backend but you are not logged in as an authorized user to get or update this data you are sending data to the API in a format it does not recognize. For example, trying to update an integer field but sending a string.  
Learn more about debugging API requests .