Airtable data source | WeWeb Documentation

|  |  |
| --- | --- |
| URL source | <https://docs.weweb.io/plugins/data-sources/airtable-data.html> |
| Date de scraping | 2025-04-08T15:55:23.409628+00:00 |

## Airtable data source ​

## Add the Airtable plugin ​

In order to get data from Airtable, you first need to add Airtable as a data source in WeWeb:  
Go to Plugins Browse through Data sources Select Airtable Add your personal access token Make sure to click Continue to add the plugin Then Add a collection of Airtable data

### Images

<https://docs.weweb.io/assets/airtable-data1.DJoPQbXh.png>

## Access token vs API key ​

You can create Airtable access tokens here.  
Make sure to set it up with the access to the base you want to use in WeWeb, and the proper scopes:  
data.records:read so that your app can read data from Airtable data.records:write so that your app can write data to Airtable schema.bases:read so that your app can read the schema of your Airtable base and know which types to use in workflows' actions

### Images

<https://docs.weweb.io/assets/airtable-scopes.PZsbARg7.png>

## Access token vs token ID ​

When configuring the Airtable plugin in WeWeb, a common mistake is to add the TOKEN ID of an Airtable access token:  
  
That won’t work.  
You need to make sure you copy the access token itself when you first create it:  
  
You won’t be able to see it again so if you didn’t copy the Personal Access Token when you first created a token, you might need to create a new one.

### Images

<https://docs.weweb.io/assets/airtable-warning-tokenid.bU_pD2tw.png>

<https://docs.weweb.io/assets/airtable-warning-accesstoken.CyA_7nTZ.png>

## Create a collection ​

Once you’ve connected an Airtable account to WeWeb, you will be able to create Airtable data collections in WeWeb:

### Images

<https://docs.weweb.io/assets/airtable-collection-create.CUMVhxl7.png>

## Select data to fetch ​

Once you've selected a source for your collection, you will get access to the Configuration tab, you can select very precisely what data you want to fetch from Airtable.  
Required fields include the base, table, view, and fields you want to fetch:

### Images

<https://docs.weweb.io/assets/airtable-collection-configuration.BXUdx_tc.png>

## Filters by formula ​

With filters by formula, you can query data with filters so that Airtable only sends specific data to your WeWeb frontend.  
The filters available are documented in Airtable's own API documentation .  
For example, you can user a filter by formula to search a text through a list of items: SEARCH(stringToFind, whereToSearch)  
In WeWeb, it would look something like this: 'SEARCH(" ' + variable + ' ", Car\_name)'  
The tricky part here will be getting the quotes right because the Airtable API expects the search term to be in between quotes like it is here:  
  
Indeed, in the Current value displayed in WeWeb, you can see the entire formula is in between quotes and so is the search term "Volk". That is the data format expected by Airtable's API. Make sure to stick to it.

### Images

<https://docs.weweb.io/assets/airtable-data-3.DCFZ6SzV.png>

## Fetch data ​

Once you've configured the data you want to fetch from Airtable, you can decide if you want to fetch this collection automatically and if you want to preserve it on navigation:

### Images

<https://docs.weweb.io/assets/airtable-collection-fetch-data.DejQlqUG.png>

## Fetch automatically ​

When Fetch this collection automatically is enabled, the collection will automatically be fetched when a page that contains the collection is loaded in the user's navigator.  
Sometimes, you want to disable this option to have finer control of where and when the data is fetched.  
For example, you might want to fetch a collection when a user clicks on a specific button in your app.

## Preserve on navigation ​

When Preserve on navigation is enabled, the collection is only fetched the first time a user loads a page with that collection.  
Sometimes, you want to disable this option to have finer control of where and when the data is preserved in the user's navigator.  
For example, you might want to make sure that a user's data is not preserved on navigation so that, if two users login from the same computer, user A can't see user B's data after they've logged out.

## Sort, filter, and paginate the data in WeWeb ​

At this stage, you have fetched the data from Airtable. It is available for use in WeWeb.  
Any sort, filter, or pagination you add here is done in the frontend. This means that, even if the data is not displayed on the page, it is accessible in the user's browser. Learn when to use frontend vs backend filters .  
Learn how to display Airtable collection data in WeWeb

## Update Airtable data ​

Use case: when a customer returns a rental car, you want employees at the rental location to update the mileage of that car.  
In order to do this, you will have to create a workflow that allows users to create, update, or delete a record in Airtable:  
  
Once you have chosen the type of action you want to trigger, you will be invited to send information to Airtable:  
  
In the example above, you can see:  
we will update the record in the Fleet - Dynamic collection that has the same id as the id in our selectedLine variable we will update the Mileage column with the the 506 value

### Images

<https://docs.weweb.io/assets/airtable-crud-actions.DD76JeiI.png>

<https://docs.weweb.io/assets/airtable-crud-fields.B08lJz5d.webp>

## Partial update ​

If you want to update specific fields in Airtable and not an entire record, make sure to only select the fields you want to update.  
In the example below, you can see (on the right) that we only selected the Mileage field. As a result, when we test our action, we can see in the payload sent to Airtable (on the left) that we are only sending data to update the Mileage field of the record:  
  
If we select all the fields without filling in values, we are telling Airtable, please replace the values in these columns with null .  
In the example below, the columns Car\_name , Image , PDF , etc. will now be empty for that record in Airtable because we told Airtable the value for these fields are null :

### Images

<https://docs.weweb.io/assets/airtable-update1.lhhjt2qs.png>

<https://docs.weweb.io/assets/airtable-update2.DJ4moAIq.png>

## Host Airtable images and files on a CDN ​

It is no longer possible to use Airtable's attachment hosting because links will only last 2 hours before they need to be regenerated.  
What's the impact on your WeWeb projects?  
When using dynamic collections, you can still use Airtable attachments as image or file URLs in WeWeb because the links are fetched when you fetch a collection (on page load by default or when you trigger the fetch in a Workflow).  
However, if you are using static or cached collections, you'll need to upload your Airtable attachments to a CDN like Cloudflare or AWS S3, and use the links from the CDN in WeWeb.  
We realize it's hard tedious work, so we made a small tutorial on how to fix this automatically on Airtable by using Xano to host images, and a small Airtable automation.  
Step 1: make sure you've got a Xano instance and an endpoint to upload content .  
Step 2: in Airtable, create a new text column on your table that'll be used to store the images' URLs in the CDN:  
  
Step 3: in Airtable, create an automation that is triggered "When record updated" and watch your former image/file column:  
  
Step 4: add an action to run a script and copy/paste this script:  
Step 5: replace xanoBaseUrl by your own Xano instance and change the column names in the script to match the ones in your table (they are marked with comments in the script).  
Step 6: activate the automation in Airtable. Now, every time an attachment is added to the column in Airtable, the newly created URL column will display the CDN's URLs, joined by a comma (",").  
To use the images in WeWeb, simply use this newly created column as your images source inside your project (you can get back the URLs by splitting the string on the commas, like so):

### Code

Langage: unknown

const xanoBaseUrl = 'https://xc0b-vcze-d4we.n7.xano.io' // Edit this variable to match your Xano instance subdomain  
let table = base.getTable('property')  
  
let config = input.config()  
let recordId = config.updatedRecordId  
  
let record = await table.selectRecordAsync(recordId, {fields: table.fields})  
  
let images = record.getCellValue('images') // Update this value to match your image/file attachment column  
let imagesUrlCdn = []  
  
for (let image of images) {  
 let imageUrl = image.url  
 let result = await fetch(xanoBaseUrl + '/api:SJOHbPGi/upload/image', { // Update this path to match your Xano upload file endpoint  
 method: 'POST',  
 headers: {  
 'Accept': 'application/json',  
 'Content-Type': 'application/json'  
 },  
 body: JSON.stringify({'content': imageUrl})  
 })  
 let response = await result.json()  
 let imageUrlCdn = xanoBaseUrl + response.path  
 imagesUrlCdn.push(imageUrlCdn)  
}  
  
await table.updateRecordAsync(record, {  
 'images\_urls': imagesUrlCdn.join() // Update the key to match your newsly created URL column  
})

### Images

<https://docs.weweb.io/assets/airtable-cdn1.Cqlw20ef.png>

<https://docs.weweb.io/assets/airtable-cdn2.DDUM36T1.png>

<https://docs.weweb.io/assets/airtable-cdn3.C_BCGDOR.png>