

Spotify Radar - EN (PDF)

[Spotify Developer](#)

[Artists IDs](#)

[Telegram Bot](#)

[Make.com](#)

[Node 1 - Variable](#)

[Node 2 - Iterator \(1\)](#)

[Node 3 - HTTP Request](#)

[Node 4 - Iterator \(2\)](#)

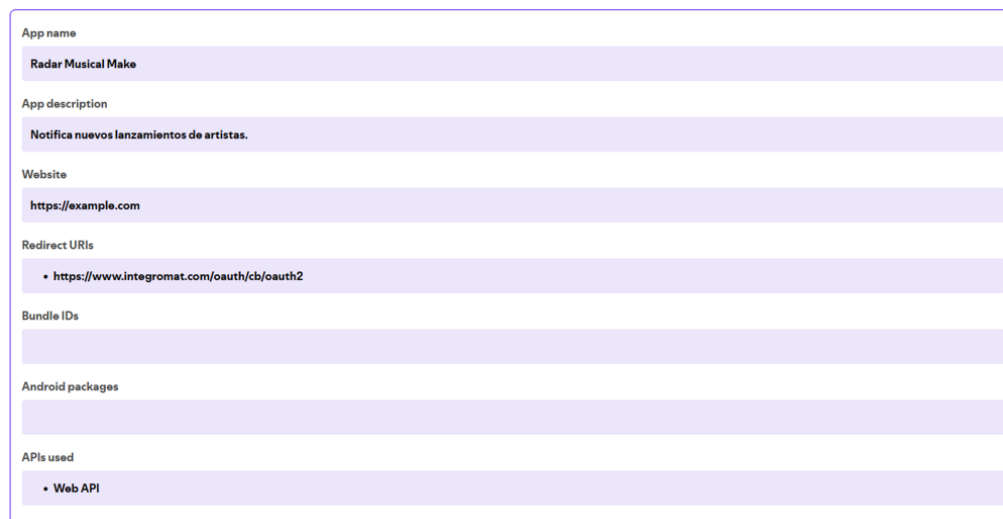
[Node 5 - Mensaje de Telegram](#)

Spotify Developer

✓ First, we need a Spotify developer account.

[Home](#) | [Spotify for Developers](#)

Once the account has been created, go to the Dashboard and create an APP. Here, you must select the following options (**Website** does not matter):



The screenshot shows the Spotify Developer Dashboard form for creating a new application. The form is divided into several sections, each with a label and a text input field. The sections are: App name (Radar Musical Make), App description (Notifica nuevos lanzamientos de artistas.), Website (https://example.com), Redirect URIs (https://www.integromat.com/oauth/callback), Bundle IDs (empty), Android packages (empty), and APIs used (Web API).

App name	Radar Musical Make
App description	Notifica nuevos lanzamientos de artistas.
Website	https://example.com
Redirect URIs	• https://www.integromat.com/oauth/callback
Bundle IDs	
Android packages	
APIs used	• Web API

App name → Any name you want

App description → Any description you want

Website → <https://example.com> (this is fake, it doesn't matter what we put here, but it can't be empty)

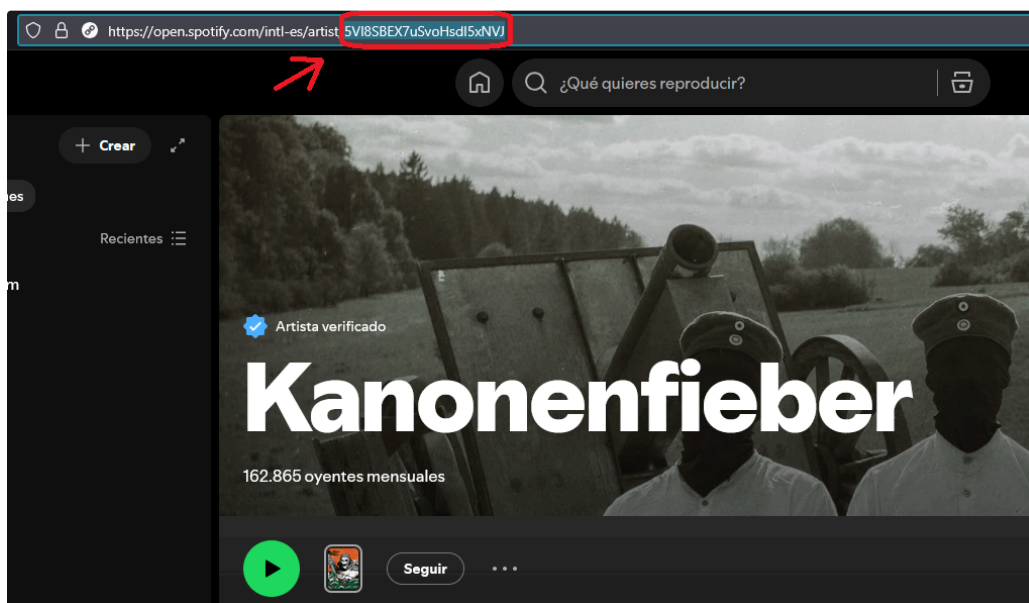
Redirect URIs → `https://www.integromat.com/oauth/cb/oauth2`

APIs used → Web API

Above this information, we will also see the **Client ID** and **Client Secret**. This is important for later, so make a note of it. We are now done with this part.

Artists IDs

Before going to [Make.com](#), you need to find the IDs of the artists you are interested in on Spotify. To do this, go to each artist's page on Spotify and look at the URL. The ID will be after `/artist/`.

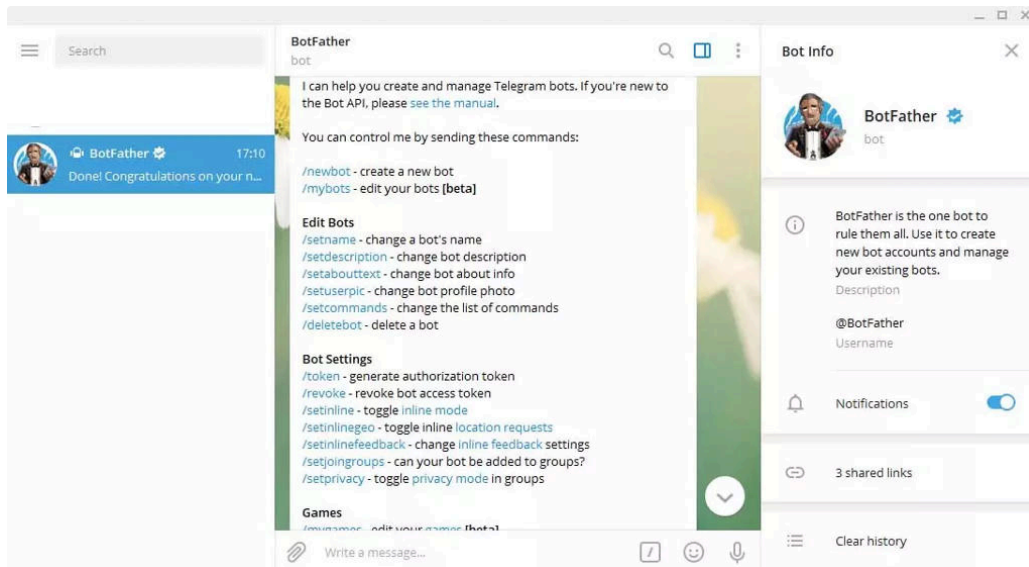


These IDs must be copied and listed in this format (each ID must be pasted without the brackets):

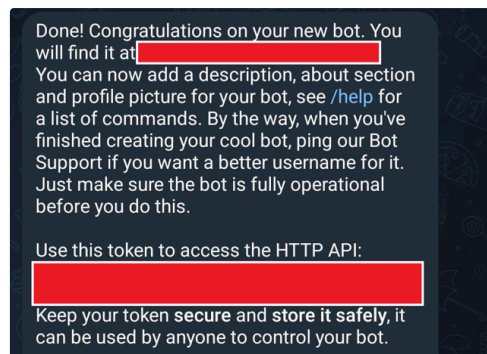
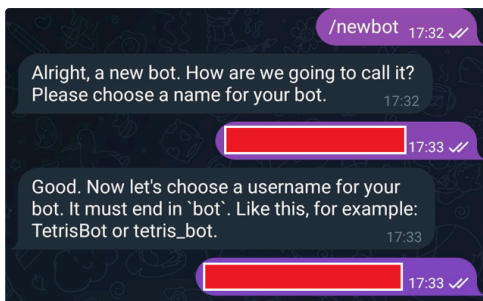
```
1 [Artist_ID1],[Artist_ID2],[Artist_ID3]
2 --- EXAMPLE ---
3 5VI8SBEX7uSvoHsdI5xNVJ,05fG473iIaoy82BF1aGhL8,240iw7B1v01BETecDLJt6m
```

Telegram Bot

To create a bot on Telegram, we will start a chat with **@BotFather**. You can search for it using that exact name.



- Click Start to begin the conversation.
- As we can see, the command to create a new bot is **/newbot**.
- Therefore, we will use this command, giving it a **name and a username**.
- Telegram will automatically give us a token or authorization code. This code is necessary for you to use the Bot API of the messaging app. On the other hand, we will have to keep this token safe, since if another user accessed it, they could control that bot.



Now we need the bot's Chat ID. To get it, we need to talk to the bot we have created. Then, open a browser and go to the following page, replacing **[TOKEN]** with the token that has just been generated:

```
1 https://api.telegram.org/bot[TOKEN]/getUpdates
```

We will copy the number after **id:** and save it, like we saved the token.

Make.com

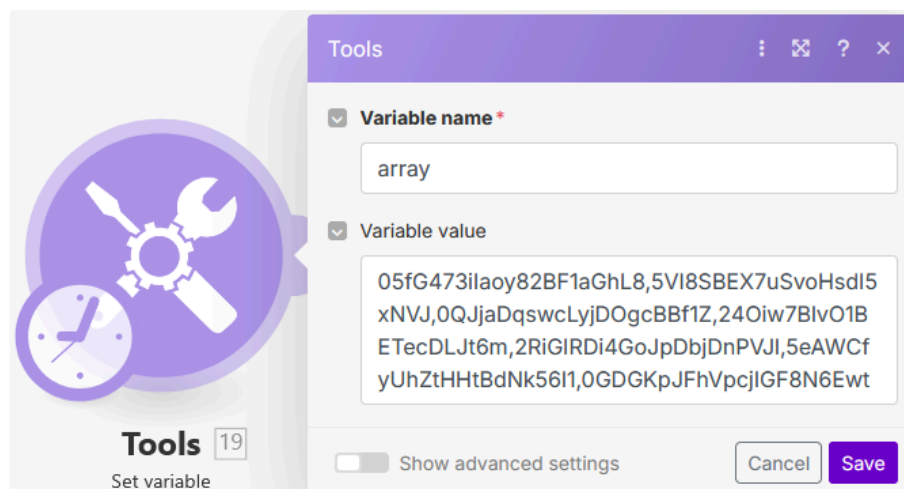
It's time to set up the automation. This is the final organizational chart we should end up with:



Node 1 - Variable

Tools > Set variable

Here we must give it the name we want, in this case `array`. The value of the variable must be the list we created earlier in the section **Artists IDs**.



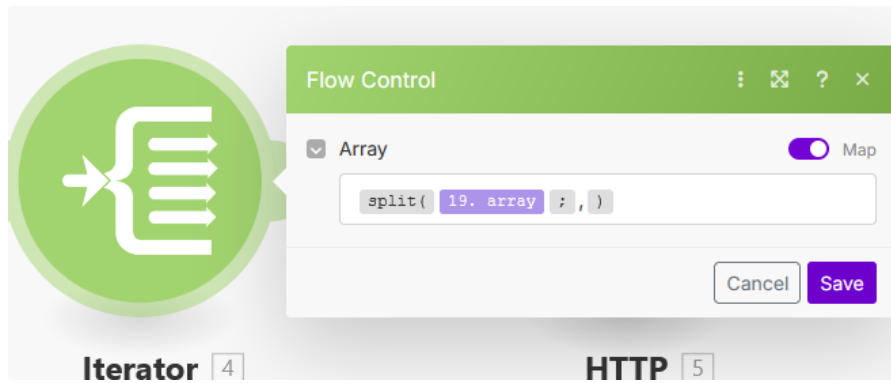
Node 2 - Iterator (1)

Flow Control > Iterator

Here we must look for the `split` function. Then we look for the variable that comes out of the previous section (in my case `19.array`) and put it after the opening parenthesis `(`. We also have to add the array separator, which in this case is a comma. It goes just before the closing parenthesis `)`.

```
1 {{split([VARIABLE]; ",")}}
```

⚠ It is important that the **Map** option is enabled.



Node 3 - HHTP Request

HTTP > Make an OAuth 2.0 request

This part is a little more complex. First, we must add the connection to Spotify as follows:

- **Connection name** → Give it any name, “Spotify Connection” for example
- **Flow type** → Authorization Code
- **Authorize URI** → `https://accounts.spotify.com/authorize`
- **Token URI** → `https://accounts.spotify.com/api/token`

- **Scope** → Empty
- **Client ID** → See the app created in **Spotify Developer**
- **Client Secret** → See the app created in **Spotify Developer**

Once the connection has been made and verified, we will enter the following, but for

`4.value` we must select what came out of the previous Iterator. Where it says `market=ES`, we must change it to our country code (ES = Spain, MX = Mexico, US = United States, etc.):

```
1 https://api.spotify.com/v1/artists/{{[VARIABLE]}}/albums?include_groups=album,single,ep&market
```

⚠ It is important that **Parse response** is set to **YES**.

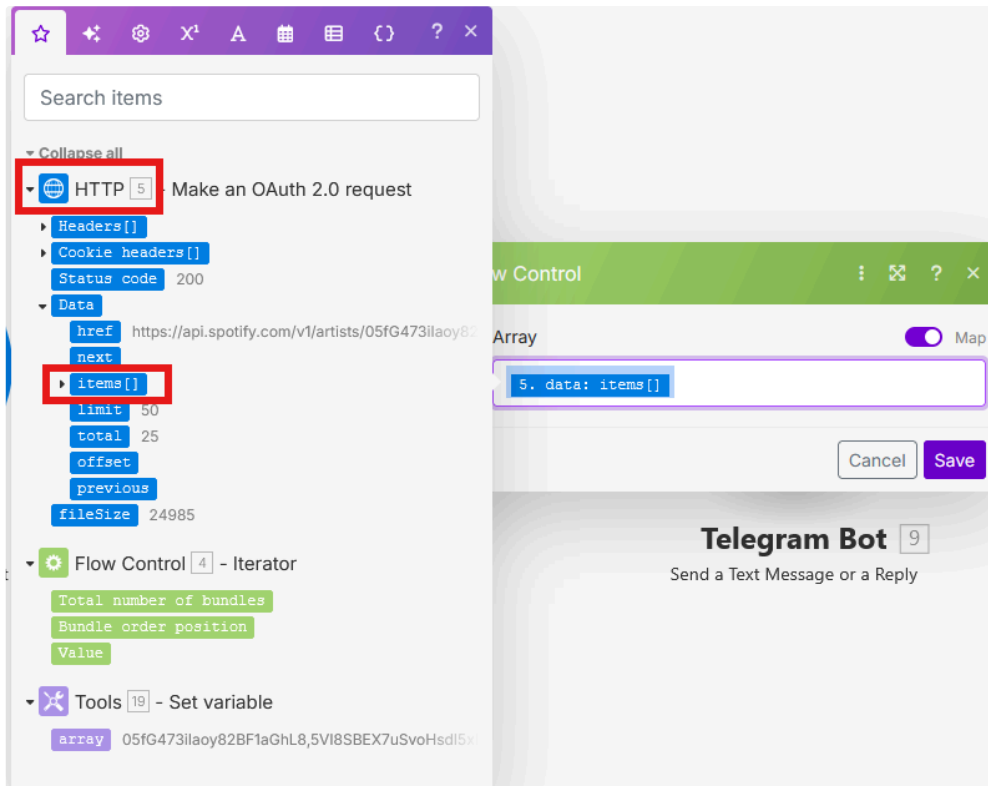
The screenshot shows a configuration panel for an HTTP request. It includes sections for URL, Method, Headers, Query String, Body type, and Parse response. The URL is configured with a variable reference [4. Value] and specific query parameters. The Method is set to GET. The Parse response option is enabled and set to 'Yes'.

Node 4 - Iterator (2)

Flow Control > Iterator

Before this, you will have to run the automation at least once, to make sure the HTTP request gets the information required. Here we simply have to enter the data that came out of the previous operation, in this case `items[]`:

⚠ It is important that the **Map** option is enabled.

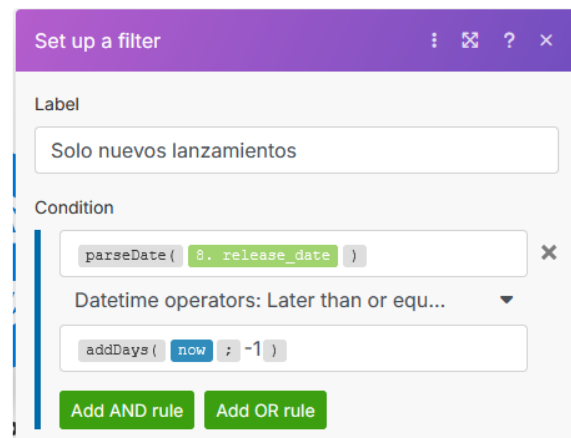
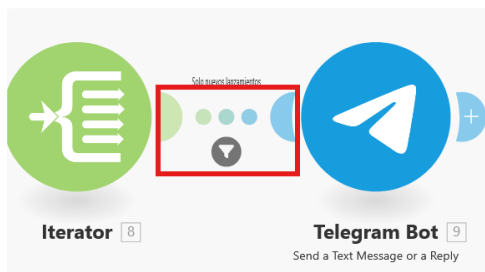


Node 5 - Mensaje de Telegram

Telegram Bot > Send a Text Message or a Reply

Before we start configuring this module, we will click on the connection between the **Iterator** and the **Telegram Bot** to apply the following filter. Where it says **8. release_date**, we must enter the release date data from the previous Iterator. This filter is designed to only capture releases from the last 24 hours, so this automation would need to be run every day, but it can be configured to run weekly by changing **-1** to **-7**, or to whatever number of days you want:

```
1 {{parseDate([VARIABLE])}}
2 {{addDays(now; -1)}}
```



Now, let's configure the Telegram message. First, we create the connection with the bot. In token, we must enter the token we have obtained from the Telegram bot in the section **Telegram Bot**:

Click in **Add**

Once the connection is made, we configure the Chat ID that we also obtained previously in the section **Telegram Bot**, and then we will simply write the message as we wish, using the data obtained from the last Iterator: `artists[]:name` (artist name) and `name` (song/album name):

```
1 🎵 New release:
2 {{8.artists[].name}} - {{8.name}}
```

Once everything is done, we can test it by setting the filter in this same section to `-365` instead of `-1` so that it picks up the releases from the last year, to see if it sends the message. Then remember to set it back to `-1` or whatever you want.

Finally, you need to activate the scenario and set it to run at regular intervals. This depends on your preferences.

⚠ Remember to adjust the aforementioned filter depending on how often you run it, because if you run it once a week but the filter is set to `-1`, it will run every week but will only catch the releases from the last day, so in this case you would have to set it to `-7`.

Schedule settings

Run scenario:
Every day

Time:
09:00

Time zone: Europe/Madrid
Format: H:mm

☐ Show advanced settings

Cancel Save

Run once ☒ Daily at 9:00

Messages example:

