

Memoria Práctica

Jaime Parra Jiménez

INTEGRACIÓN DE TECNOLOGÍAS Y SERVICIOS INFORMÁTICOS

Universidad de Almería

Correo: jpj451@inlumine.ual.es

17 de noviembre de 2025

Índice general

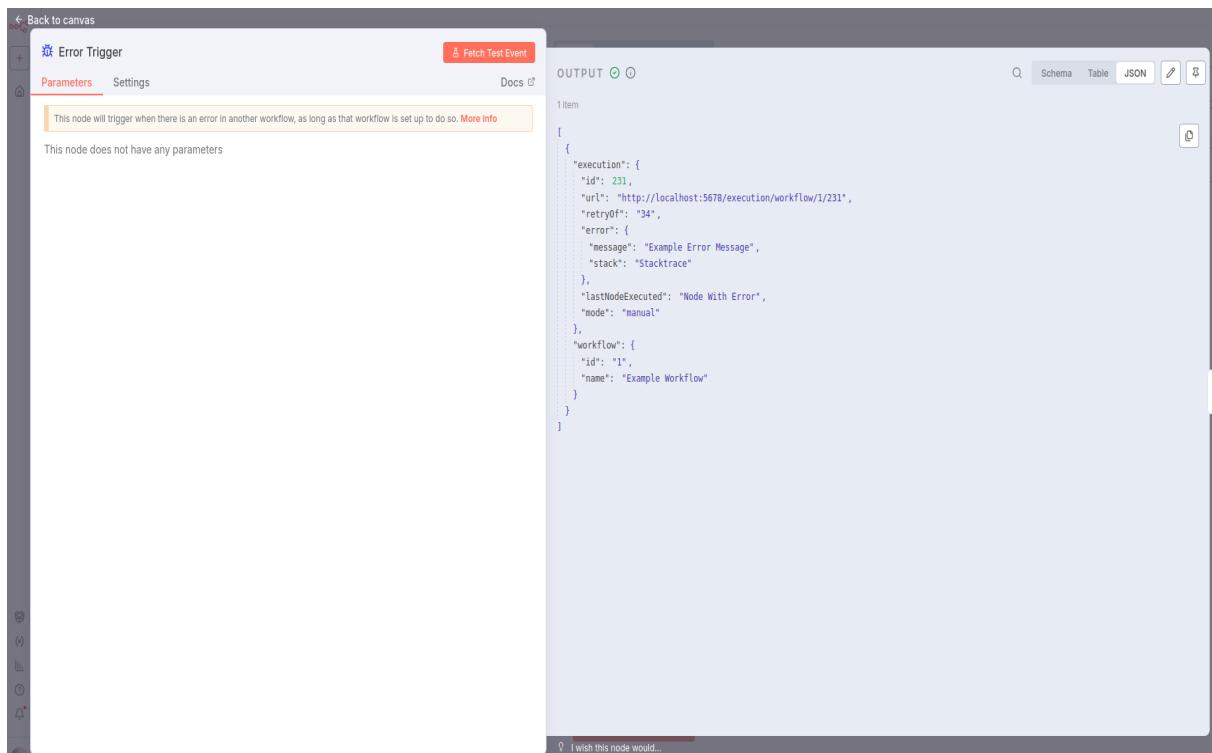
1. Ejercicio Guiado 1	2
2. Ejercicio 1	8
3. Ejercicio 2	9
4. Ejercicio 3	13

Capítulo 1

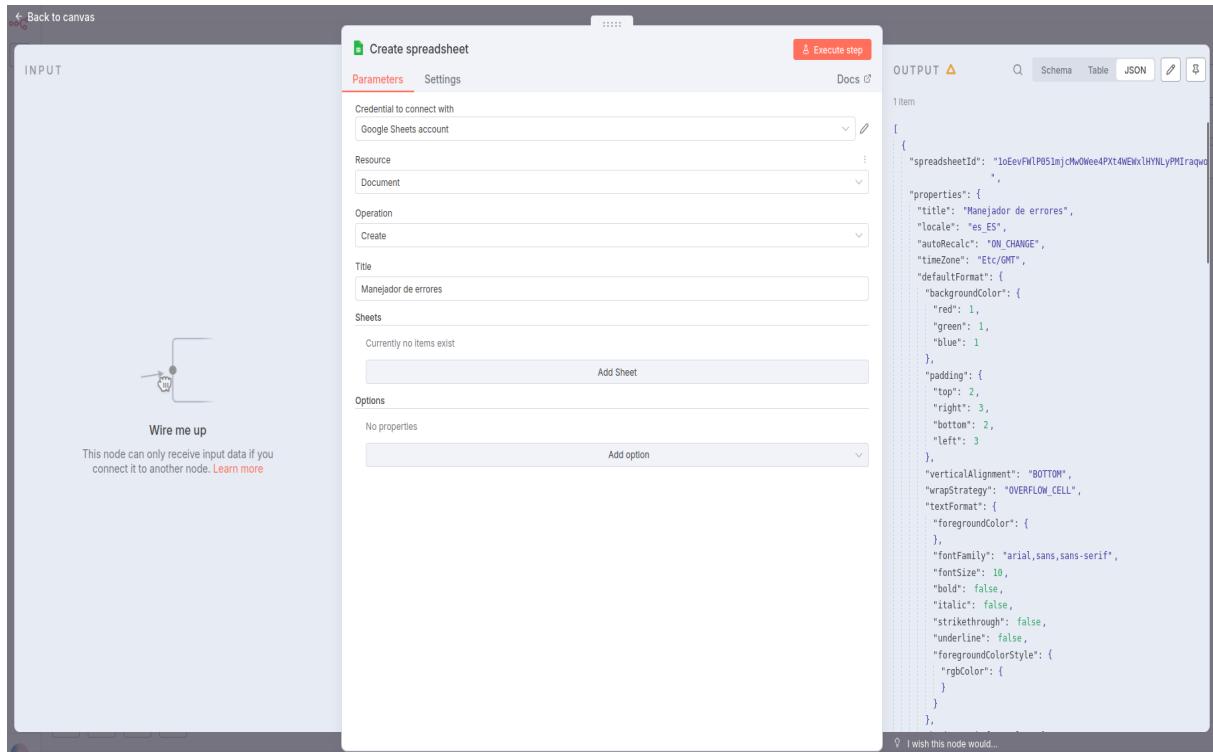
Ejercicio Guiado 1

El objetivo del ejercicio guiado es crear un flujo principal que procesa una lista de URLs de imágenes, un sub-flujo que intenta descargar cada imagen, y un flujo de errores que registra cualquier fallo en el proceso en una hoja de Google Sheets.

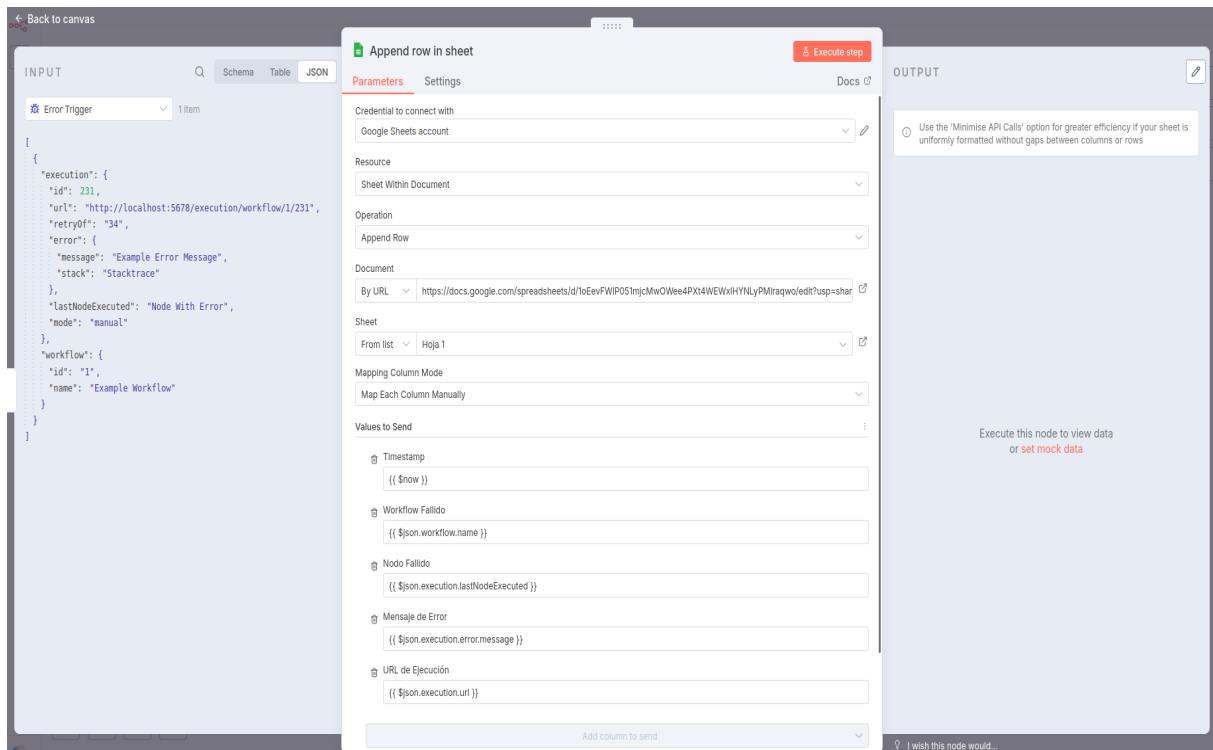
Primero se creará el flujo de trabajo de errores llamado Manejador de Errores. Para ello, lo primero que se debe hacer es añadir un nodo Error Trigger para obtener los errores.



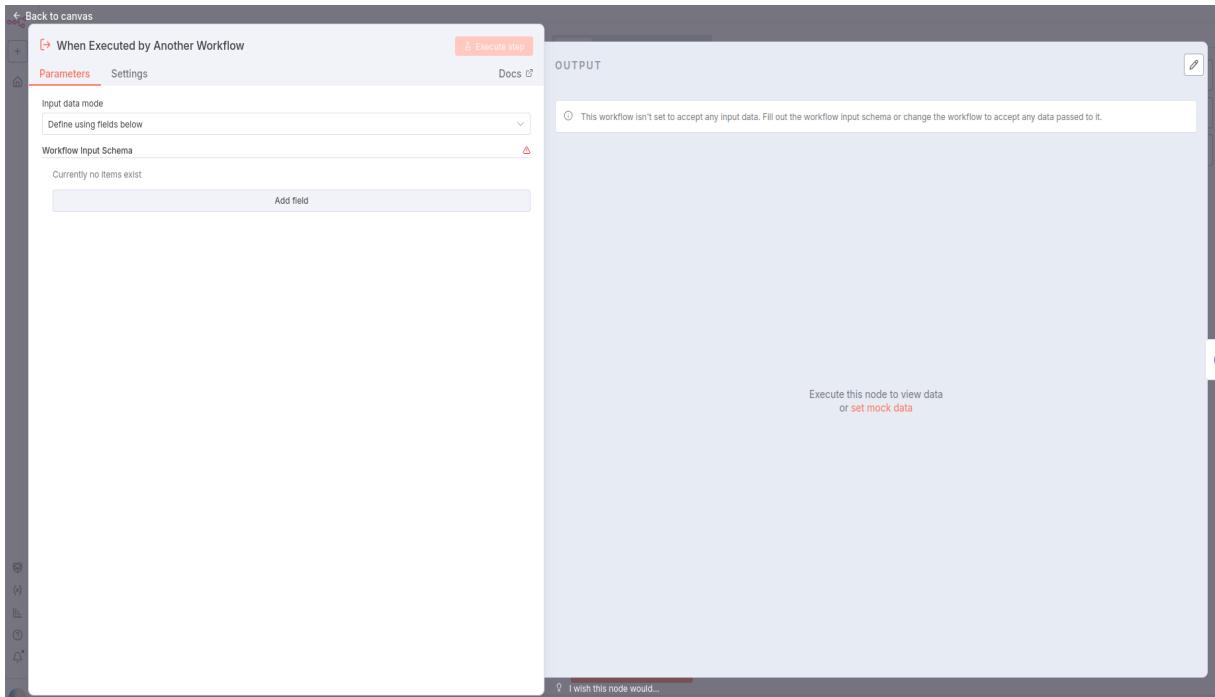
A continuación se creara una hoja en Google Sheets con las cabeceras Timestamp, Workflow Fallido, Nodo Fallido, Mensaje de Error, URL de Ejecución.



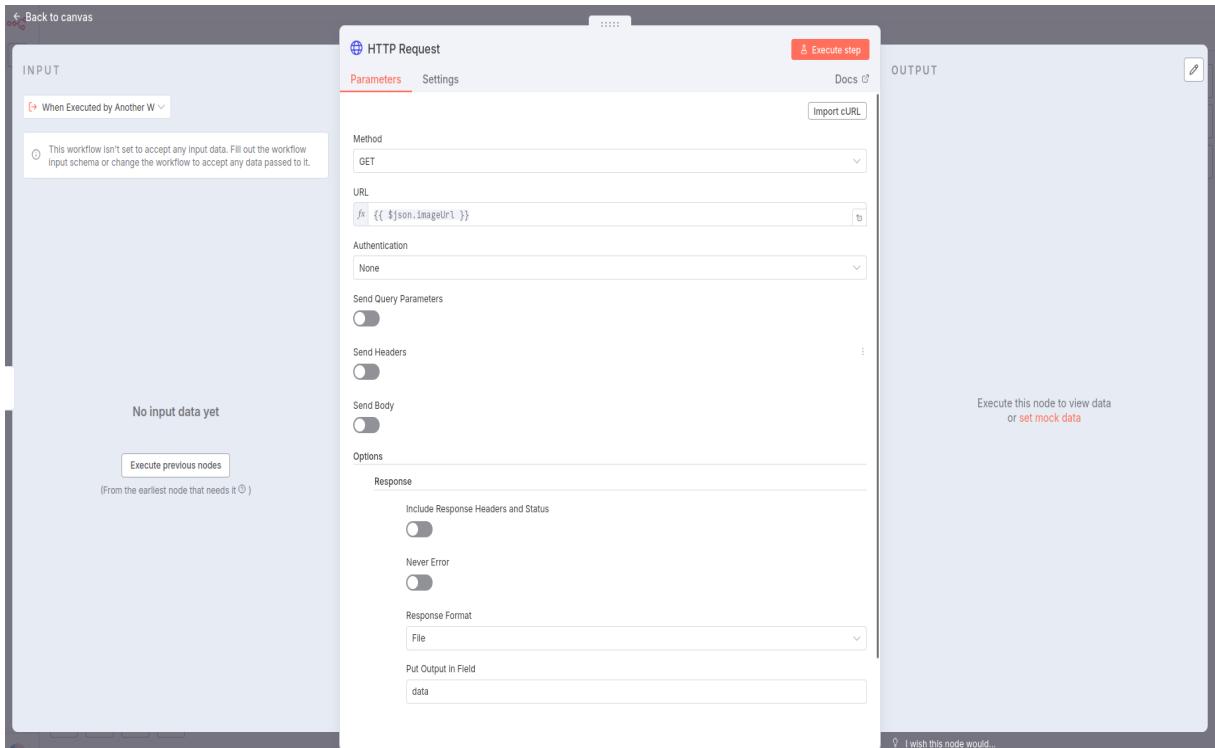
El siguiente paso será añadir un nodo de Google Sheets con la operación de Append Row para escribir en la hoja los errores.



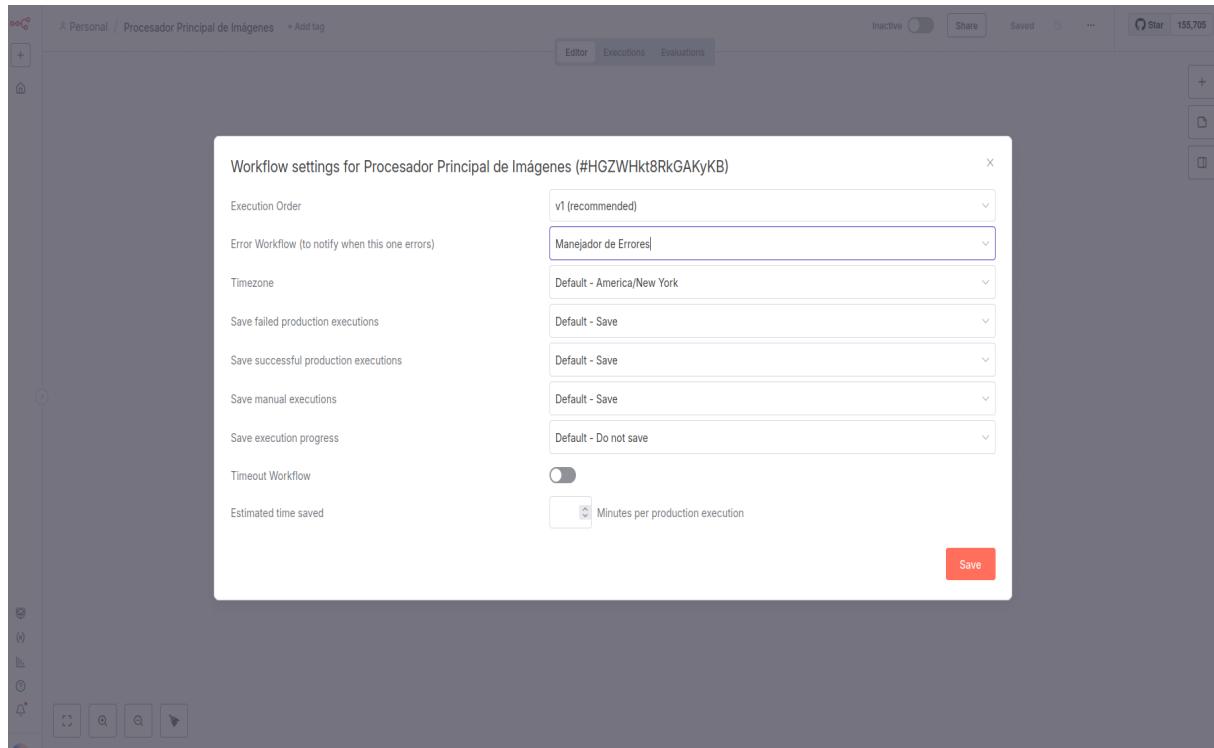
Una vez creado el manejador de errores el siguiente paso es crear el sub-flujo de trabajo. Se le asignará el nombre de Descargador de Imágenes. Primero, se añade un nodo Execute Workflow Trigger. Este nodo recibirá la URL de la imagen a descargar.



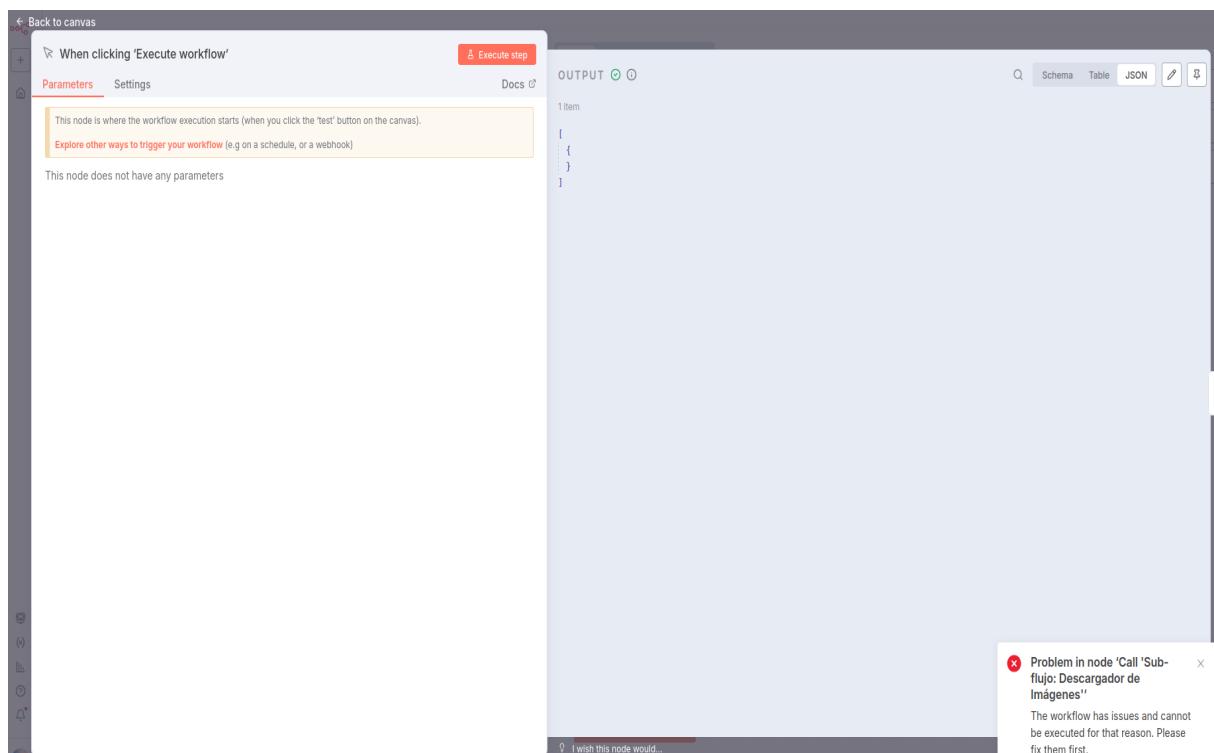
A continuación, se añade un nodo HTTP Request para obtener las URLs de las imágenes.



El último paso es crear el flujo de trabajo principal. El nombre asignado es Procesador Principal de Imágenes. Primero se debe configurar los ajustes para que en caso de errores se llame al flujo Manejador de Errores.



Para comenzar el flujo de trabajo se añade un nodo Manual Trigger.



Se conectará un nodo Edit Fields(Set) para crear una lista de URLs, esta lista será erronea para usar el Manejador de Errores.

INPUT

When clicking 'Execute workflow' ↗ 1 item

```
[{"imageUrls": "https://random.dog/woof.json"}, {"imageUrls": "https://randomfox.ca/floof/"}, {"imageUrls": "https://esta-url-no-existe.xyz/imagen.jpg"}]
```

Edit Fields

Parameters Settings

Mode: Manual Mapping

Fields to Set:

imageUrls = Array

```
[{"imageUrls": "https://random.dog/woof.json"}, {"imageUrls": "https://randomfox.ca/floof/"}, {"imageUrls": "https://esta-url-no-existe.xyz/imagen.jpg"}]
```

Drag input fields here or Add Field

Include Other Input Fields:

Options: No properties

OUTPUT

1 item

```
[{"imageUrls": [{"imageUrl": "https://random.dog/woof.json"}, {"imageUrl": "https://randomfox.ca/floof/"}, {"imageUrl": "https://esta-url-no-existe.xyz/imagen.jpg"}]}
```

Problem in node 'Call 'Sub-flujo: Descargador de imágenes''
The connection cannot be established, this usually occurs due to an incorrect host (domain) value

Para procesar la lista se añadirá un nodo Split Out con el campo imageUrls para separar la lista en ítems individuales.

INPUT

When clicking 'Execute workflow' ↗ 1 item

```
[{"imageUrls": "https://random.dog/woof.json"}, {"imageUrls": "https://randomfox.ca/floof/"}, {"imageUrls": "https://esta-url-no-existe.xyz/imagen.jpg"}]
```

Split Out

Parameters Settings

Fields To Split Out: imageUrls

Use \$binary to split out the input item by binary data

Include: No Other Fields

Options: No properties

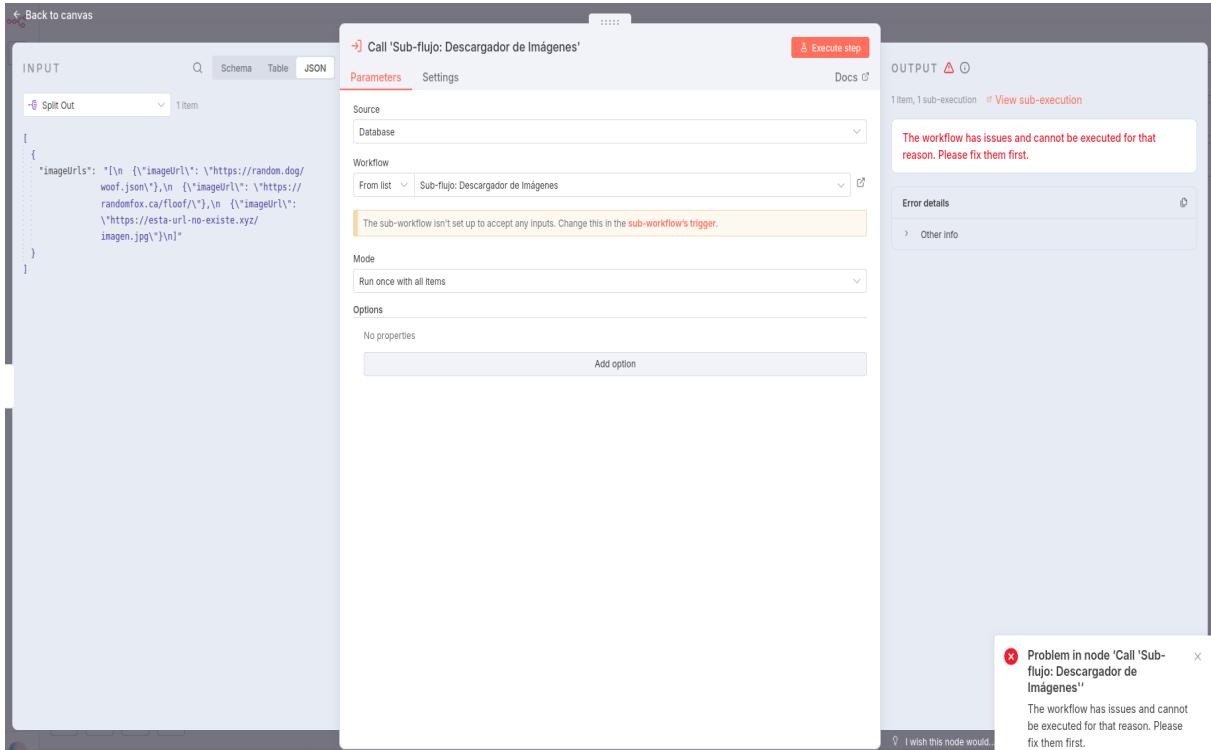
OUTPUT

1 item

```
[{"imageUrls": [{"imageUrl": "https://random.dog/woof.json"}, {"imageUrl": "https://randomfox.ca/floof/"}, {"imageUrl": "https://esta-url-no-existe.xyz/imagen.jpg"}]}
```

Problem in node 'Call 'Sub-flujo: Descargador de imágenes''
The workflow has issues and cannot be executed for that reason. Please fix them first.

El último paso será llamar al sub-flujo. Para ello añadirá un nodo Execute Workflow y seleccionaremos el sub-flujo Descargador de Imágenes.



Al ejecutar el flujo principal se creará un error que se añadirá a la hoja de Google Sheets creada anteriormente. Esto nos permitirá poder localizar el error.

	A	B	C	D	E	F	G	H	I	J
1	Timestamp	Workflow Fallido	Nodo Fallido	Mensaje de Error	URL de Ejecución					
2	2025-11-12T11:01:10.095-05:00	Example Workflow	Node With Error	Example Error Message	http://localhost:5678/execution/workflow/1/231					
3										
4										
5										
6										
7										
8										
9										
10										
11										
12										
13										
14										
15										
16										
17										
18										
19										
20										
21										
22										
23										
24										
25										
26										
27										
28										
29										
30										
31										
32										
33										
34										
35										

Capítulo 2

Ejercicio 1

El objetivo del ejercicio 1 es mejorar el "Manejador de Errores" para que, además de registrar el error en Google Sheets, envíe una notificación por correo electrónico. Siguiendo la estructura del Manejador de Errores creado anteriormente, con un nodo Error Trigger para recibir los errores y un nodo de Google Sheets para escribir los errores en la hoja, el siguiente paso será añadir un nodo Send Email para enviar los errores. Para ello se necesita especificar el emisor y receptor del mensaje, el asunto y el cuerpo. Después de tenerlo configurado al producirse el error al receptor le llegará un mensaje con el error.

[Back to canvas](#)

Send email

Parameters **Settings** **Execute step**

Credential to connect with: SMTP account

Operation: Send

From Email: jpj451@inlumine.ual.es

To Email: jpj451@inlumine.ual.es

Subject: ¡Alerta de Error en n8n!

Email Format: HTML

HTML:

```
Fecha: {{${now}}}
{{${json["Workflow Fallido"]}}
{{${json["Nodo Fallido"]}}
{{${json["Mensaje de Error"]}}
{{${json["URL de Ejecución"]}}
```

Options

No properties

Add option

OUTPUT **Docs** **Schema** **Table** **JSON** **CSV**

1 item

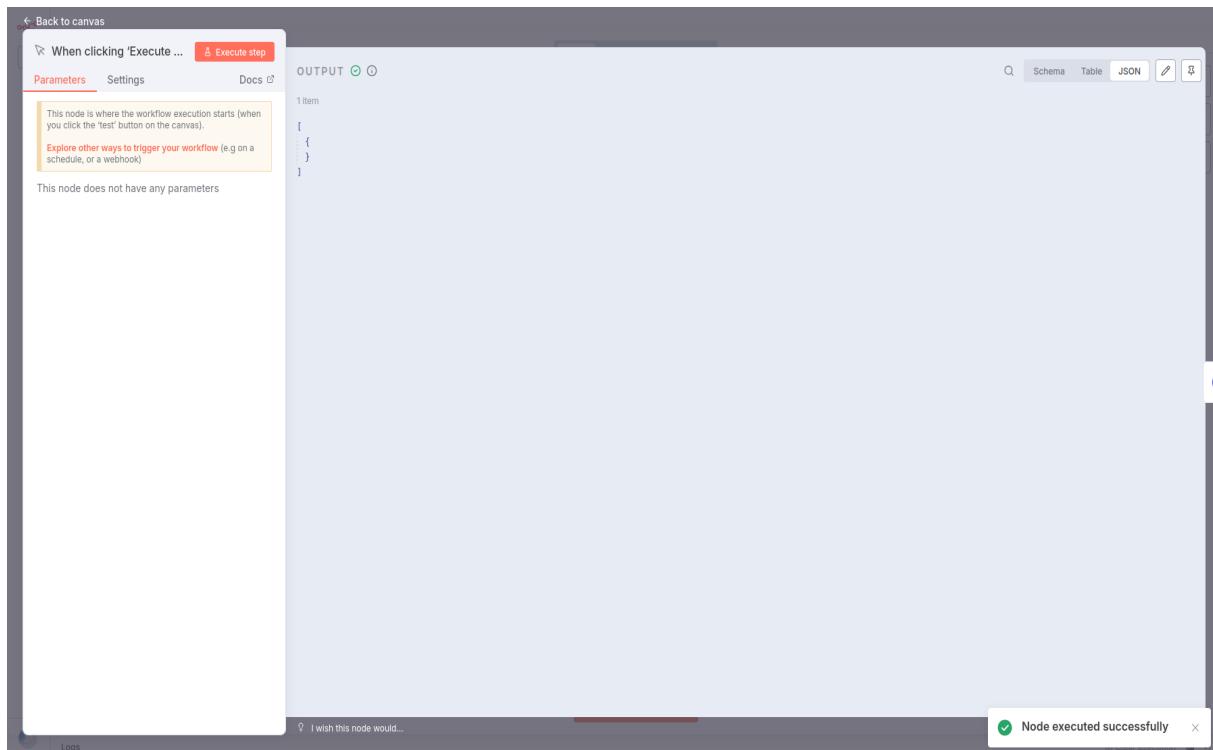
```
{
  "accepted": [
    "jpj451@inlumine.ual.es"
  ],
  "rejected": [
  ],
  "ehlo": [
    "SIZE 35882577",
    "8BITMIME",
    "AUTH LOGIN PLAIN XOAuth2 PLAIN-CLIENTTOKEN OAUTHBEARER XOAuth",
    "ENHANCEDSTATUSCODES",
    "PIPELINING",
    "CHUNKING",
    "SMTPUTF8"
  ],
  "envelopeTime": 165,
  "messageTime": 543,
  "messageSize": 813,
  "response": "250 2.0.0 OK 1762964922
5d1f17b1804b1-47789239a4dsm15262185e9.12 - gsmtp",
  "envelope": {
    "from": "jpj451@inlumine.ual.es",
    "to": [
      "jpj451@inlumine.ual.es"
    ]
  },
  "messageId": "<2ac0148b-
ddef-28d5-3bb0-56f5c45ab183@inlumine.ual.es>"
}
```

I wish this node would... **Node executed successfully**

Capítulo 3

Ejercicio 2

El objetivo del ejercicio 2 es crear un sub-flujo de trabajo reutilizable que valide la estructura de los datos de un producto y lance un error si faltan campos clave. Para comenzar el flujo principal, llamado Fake Store API, se añadirá un nodo Manual Trigger.



A continuación, se usará un nodo HTTP Request para obtener los datos de la API.

The screenshot shows the MuleSoft Anypoint Studio interface with the following details:

- INPUT:** A JSON object representing a product item:

```
[{"id": 1, "title": "Fjallraven - Foldsack No. 1 Backpack, Fits 15 Laptops", "price": 109.95, "description": "Your perfect pack for everyday use and walks in the forest. Stash your laptop (up to 15 inches) in the padded sleeve, your everyday", "category": "men's clothing", "image": "https://fakestoreapi.com/img/81fPKd-2AYL.AC_SL1500.t.png", "rating": {"rate": 3.9, "count": 120}}]
```
- HTTP Request Node Configuration:**
 - Method:** GET
 - URL:** <https://fakestoreapi.com/products/1>
 - Authentication:** None
 - Send Query Parameters:** Off
 - Send Headers:** Off
 - Send Body:** Off
 - Options:** No properties
- OUTPUT:** The response from the HTTP request, displayed as JSON:

```
[
  {
    "id": 1,
    "title": "Fjallraven - Foldsack No. 1 Backpack, Fits 15 Laptops",
    "price": 109.95,
    "description": "Your perfect pack for everyday use and walks in the forest. Stash your laptop (up to 15 inches) in the padded sleeve, your everyday",
    "category": "men's clothing",
    "image": "https://fakestoreapi.com/img/81fPKd-2AYL.AC_SL1500.t.png",
    "rating": {
      "rate": 3.9,
      "count": 120
    }
  }
]
```

Mediante un nodo Edit Fields(Set) se modificará la salida de los datos para que el producto no tenga el precio y así poder forzar el error.

The screenshot shows the MuleSoft Anypoint Studio interface with the following details:

- INPUT:** The same JSON object as the previous screenshot:

```
[{"id": 1, "title": "Fjallraven - Foldsack No. 1 Backpack, Fits 15 Laptops", "price": 109.95, "description": "Your perfect pack for everyday use and walks in the forest. Stash your laptop (up to 15 inches) in the padded sleeve, your everyday", "category": "men's clothing", "image": "https://fakestoreapi.com/img/81fPKd-2AYL.AC_SL1500.t.png", "rating": {"rate": 3.9, "count": 120}}]
```
- Edit Fields Node Configuration:**
 - Mode:** Manual Mapping
 - Fields to Set:**

price	T String	=	[empty]
title	T String	=	<code>{{{ \$json.title }}}</code>
 - Options:** No properties
- OUTPUT:** The modified JSON output where the price field is removed:

```
[
  {
    "title": "Fjallraven - Foldsack No. 1 Backpack, Fits 15 Laptops"
  }
]
```

A message at the bottom right indicates: "Node executed successfully".

Una vez que la salida de datos está modificada, se hará una llamada al sub-flujo para que se ejecute mediante un nodo Execute Workflow.

The screenshot shows a workflow interface with two main panels: INPUT and OUTPUT.

INPUT Panel:

- Header: Call 'Sub-workflow: Validador de Productos'
- Source: Database
- Workflow: From list - Sub-workflow: Validador de Productos
- Mode: Run once with all items
- Options: No properties

OUTPUT Panel:

- Header: View sub-execution
- Content: 1 Branch (1 item) - 2 Branch
- JSON Output: [{ "price": "", "title": "Fjallraven - Foldsack No. 1 Backpack, Fits 15 Laptops" }]

At the bottom right, there is a message: "I wish this node would... Node executed successfully".

El sub-flujo recibe el nombre de Sub-flujo: Validador de Productos y comienza con un nodo Execute Workflow Trigger.

The screenshot shows a workflow interface with two main panels: INPUT and OUTPUT.

INPUT Panel:

- Header: When Executed by Anon...
- Parameters: Define using fields below
- Workflow Input Schema:

Name	title
Type	Allow Any Type

Name	price
Type	Allow Any Type
- Add field button

OUTPUT Panel:

- Header: I wish this node would...
- Content: Clear execution

Después se añade un nodo IF para comprobar si los campos title y price existen y no están vacíos.

The screenshot shows the MuleSoft Anypoint Studio canvas with an 'If' node selected. The input is a JSON array with one item: { "title": null, "price": null }. The conditions are set to check if both 'price' and 'title' fields are not empty. The output section shows a 'True Branch' and a 'False Branch' containing one item. A tooltip indicates 'No output data in this branch'.

En la salida del false se añadirá un nodo Stop and Error con el mensaje Validación fallida: Faltan campos de producto obligatorios.

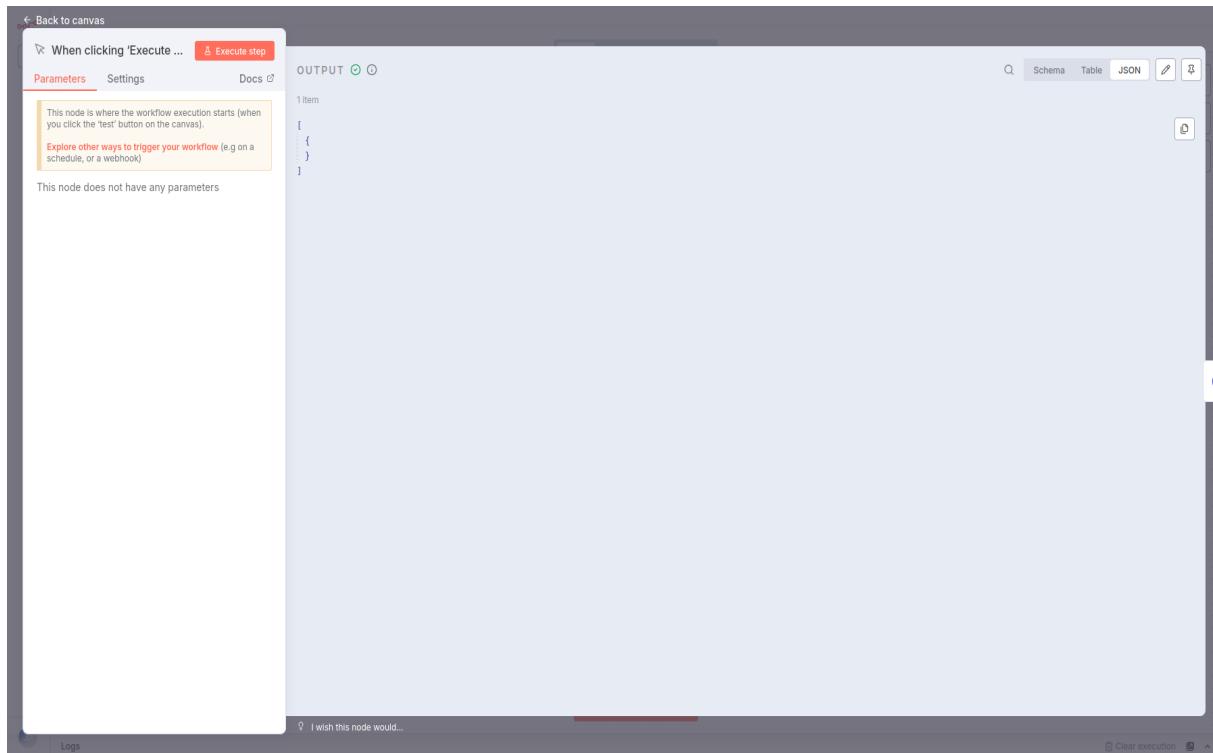
The screenshot shows the MuleSoft Anypoint Studio canvas with a 'Stop and Error' node added to the flow. The input is the same JSON array as the previous step. The error type is set to 'Error Message' with the message 'Validación fallida: Faltan campos de producto obligatorios'. The output shows this message in the log. A tooltip indicates 'Problem in node 'Stop and Error''.

Capítulo 4

Ejercicio 3

El objetivo de ejercicio 3 es crear un flujo principal que decide qué tarea realizar y llama al sub-flujo apropiado para ejecutarla.

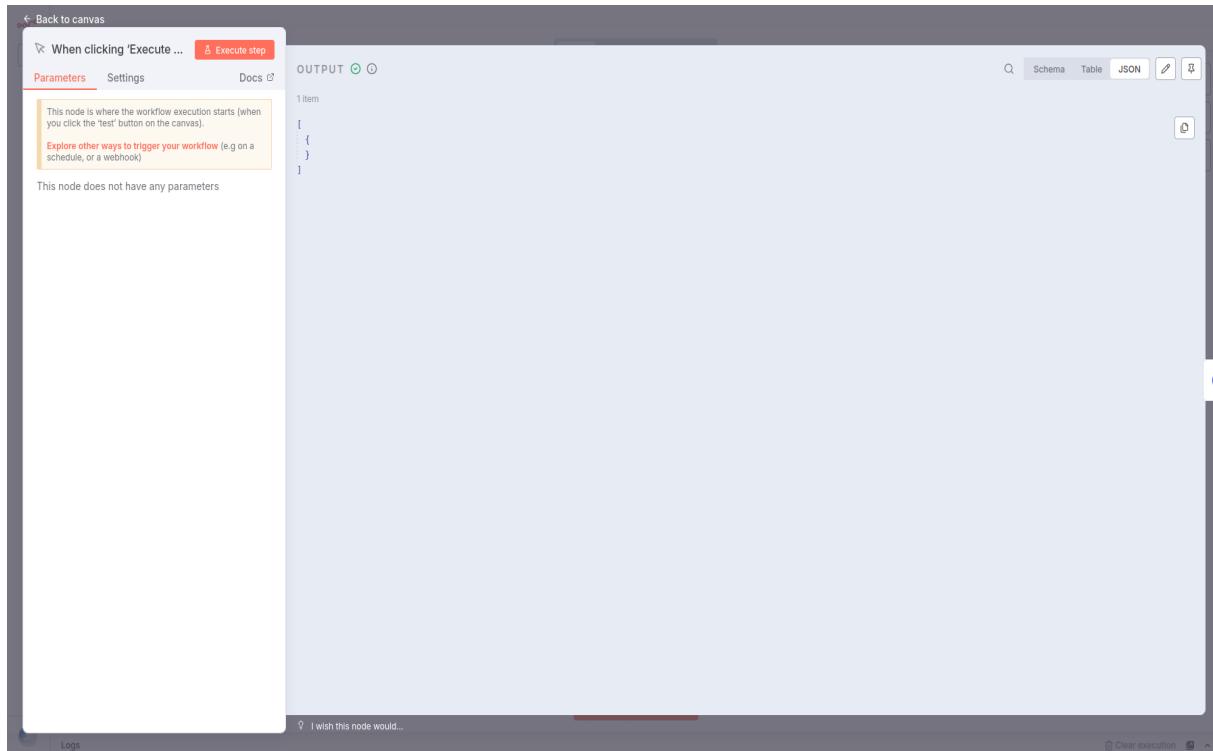
Primero se creará un sub-flujo llamado Sub-flujo: Obtener Festivos para poder obtener los festivos. El primer paso es añadir un nodo Manual Trigger para poder iniciar el flujo.



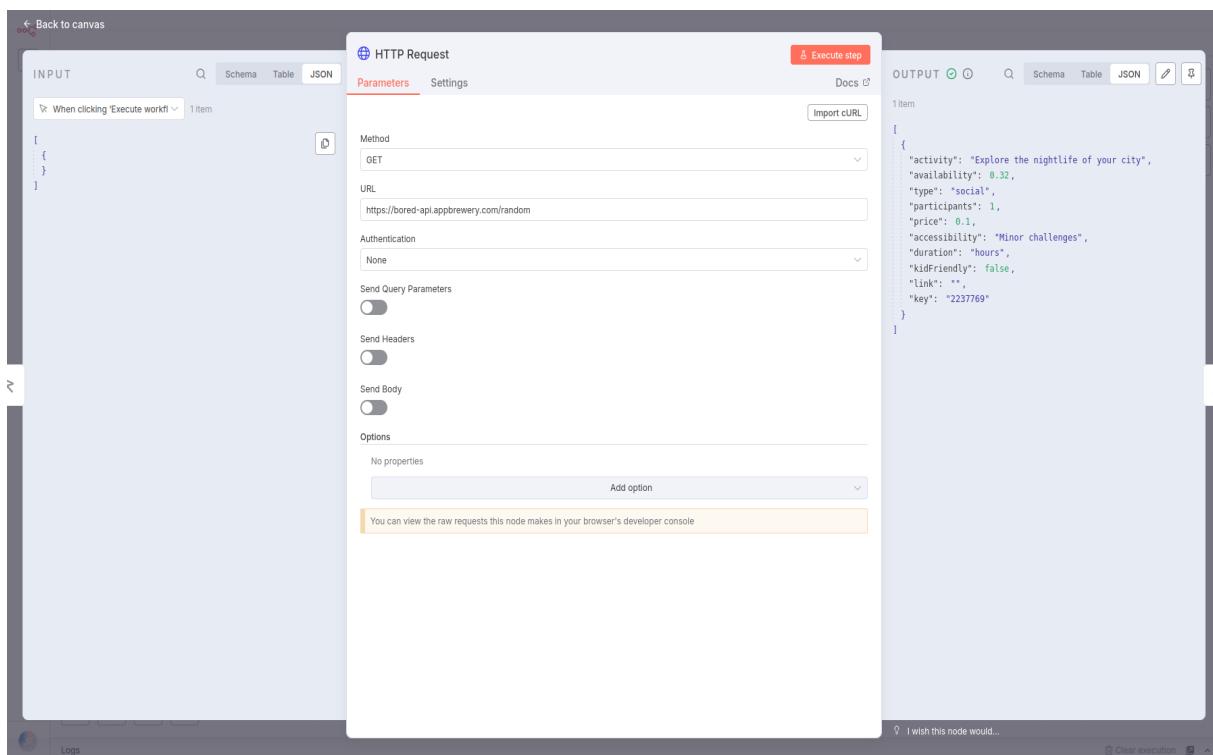
A continuación se añade un nodo HTTP Request para obtener los festivos.

El último paso es añadir un nodo Edit Fields(Set) para guardar solamente la fecha y el nombre del festivo.

También se creará el segundo sub-flujo llamado Sub-flujo: Obtener Actividad para poder obtener las actividades. El primer paso es añadir un nodo Manual Trigger para poder iniciar el flujo.



A continuación se añade un nodo HTTP Request para obtener las actividades.



El último paso es añadir un nodo Edit Fields(Set) para guardar solamente el nombre de la actividad

The screenshot shows the Zapier workflow builder interface. On the left, the 'INPUT' section displays an 'HTTP Request' node with a JSON payload containing fields like 'activity', 'availability', 'type', 'participants', 'price', 'accessibility', 'duration', 'link', and 'key'. On the right, the 'OUTPUT' section shows a JSON object with a single field 'actividad' set to the value of the 'activity' field from the input. The 'Edit Fields' node configuration includes a 'Mode' dropdown set to 'Manual Mapping' and a 'Fields to Set' table where 'activity' is mapped to 'actividad'.

Con los dos sub-flujos creados es hora de crear el flujo principal, el cual comenzará con un nodo Manual Trigger.

The screenshot shows the Zapier workflow builder interface. On the left, the 'Parameters' tab of a 'Manual Trigger' node is active, displaying instructions for triggering the workflow. The 'OUTPUT' section on the right shows an empty JSON object with a single item placeholder. The 'Logs' tab at the bottom is visible.

A continuación se añadirá un Edit Fields(Set) que se usará como variable de control para llamar al sub-flujo de actividades o de festivos.

The screenshot shows the MuleSoft Anypoint Studio interface with the 'Edit Fields' node selected. The 'INPUT' tab on the left shows a single item with the JSON path 'When clicking "Execute workfl"'. The main panel displays the 'Edit Fields' configuration with the following settings:

- Mode:** Manual Mapping
- Fields to Set:** A table with one row: 'tarea' (String type) mapped to 'actividad'.
- Options:** No properties.

The 'OUTPUT' tab on the right shows the resulting JSON: [{ "tarea": "actividad" }].

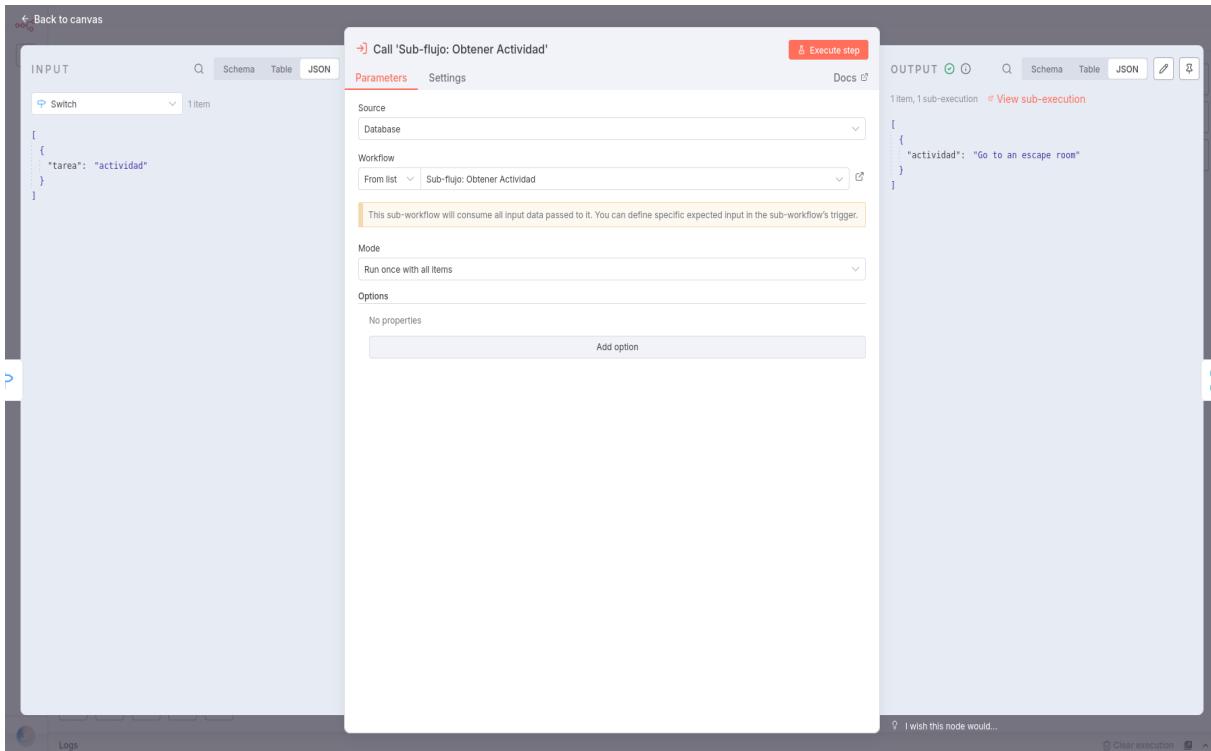
El siguiente paso es añadir un nodo Switch para evaluar la tarea.

The screenshot shows the MuleSoft Anypoint Studio interface with the 'Switch' node selected. The 'INPUT' tab on the left shows the output from the previous 'Edit Fields' node. The main panel displays the 'Switch' configuration with the following routing rules:

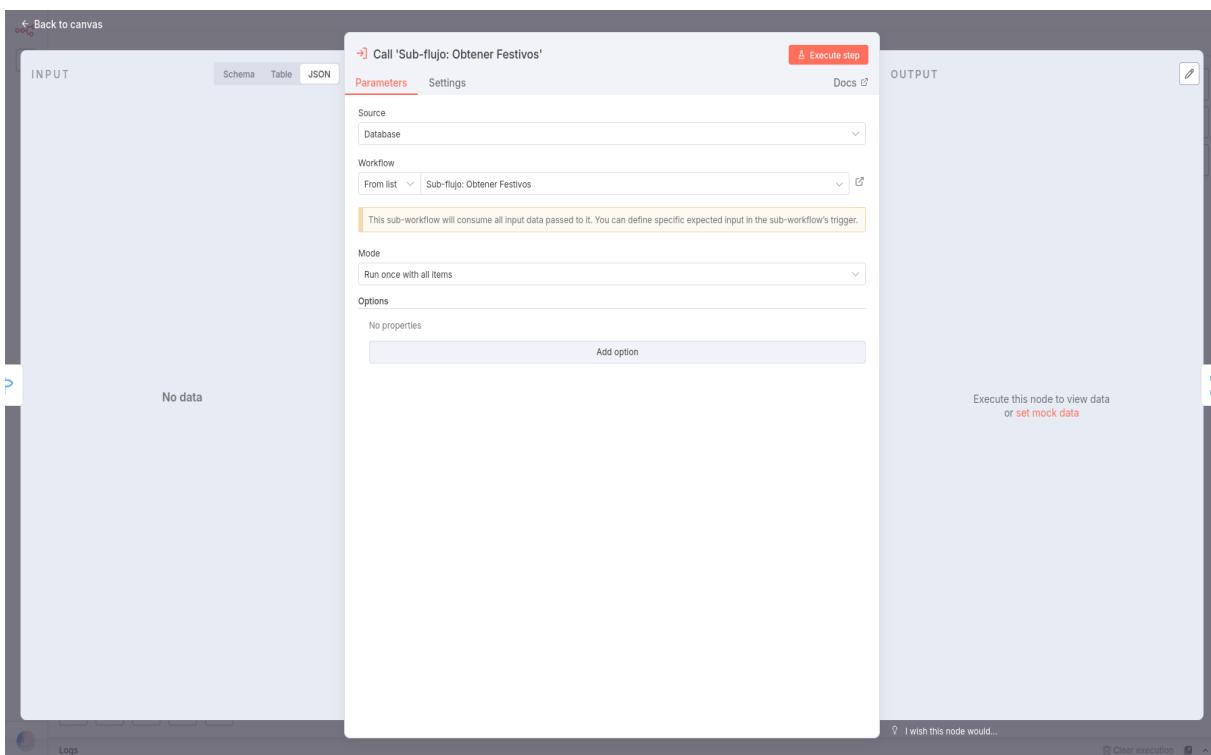
- Rule 1: Condition: `${$json.tarea} == "actividad"`, Action: `is true`.
- Rule 2: Condition: `${$json.tarea} == "festivos"`, Action: `is true`.

The 'OUTPUT' tab on the right shows two outputs: Output 0 (1 item) and Output 1, both containing the JSON: [{ "tarea": "actividad" }].

Si la tarea es actividad, se conectará a un nodo Execute Workflow que llame al sub-flujo Obtener Actividad.



En cambio, si la tarea es festivos, se conectará a un nodo Execute Workflow que llame al sub-flujo Obtener Festivos.



El último paso será conectar un nodo Merge a las salidas de ambos nodos Execute Workflow para consolidar el resultado. A continuación se podrá ver el resultado cuando la tarea es actividad y cuando la tarea es festivo.

The screenshot shows the Data Pipeline interface with a Merge node selected. The INPUT tab displays the results of a 'Call Sub-flujo: Obtener Festi.' execution, showing 32 items. The OUTPUT tab shows the resulting JSON array of 32 documents, each representing a holiday with fields 'fecha' and 'nombre'. The Merge node parameters are set to Mode: Append and Number of Inputs: 2.

```

[{"fecha": "2025-01-01", "nombre": "Año Nuevo"}, {"fecha": "2025-01-06", "nombre": "Día de Reyes / Epifanía del Señor"}, {"fecha": "2025-02-28", "nombre": "Día de Andalucía"}, {"fecha": "2025-03-01", "nombre": "Día de les Illes Balears"}, {"fecha": "2025-04-17", "nombre": "Jueves Santo"}, {"fecha": "2025-04-18", "nombre": "Viernes Santo"}, {"fecha": "2025-04-21", "nombre": "Lunes de Pascua"}, {"fecha": "2025-04-23", "nombre": "Día de Castilla y León"}, {"fecha": "2025-04-23", "nombre": "San Jorge (Día de Aragón)"}
]

```

The screenshot shows the Data Pipeline interface with a Merge node selected. The INPUT tab displays the results of a 'Call Sub-flujo: Obtener Activ.' execution, showing 1 item. The OUTPUT tab shows the resulting JSON array of 1 document, representing an activity with field 'actividad'. The Merge node parameters are set to Mode: Append and Number of Inputs: 2.

```

[{"actividad": "Go to an escape room"}]

```