

# Memoria Práctica 2

**Jaime Parra Jiménez**

INTEGRACIÓN DE TECNOLOGÍAS Y SERVICIOS INFORMÁTICOS

Universidad de Almería

Correo: [jj451@inlumine.ual.es](mailto:jj451@inlumine.ual.es)

5 de noviembre de 2025

# Índice general

1. Ejercicio Guiado 1	2
2. Ejercicio 1	5
3. Ejercicio 2	7
4. Ejercicio 3	11

# Capítulo 1

## Ejercicio Guiado 1

El objetivo del ejercicio guiado fue construir un flujo de trabajo que se ejecuta cada minuto, obtenga una lista de cinco chistes y los clasifique si son seguros o no. Para comenzar el flujo de trabajo es necesario añadir un nodo inicial Schedule Trigger. Este nodo se ha configurado para ejecutar el nodo automáticamente cada minuto, como se muestra a continuación.

The screenshot displays the configuration interface for a 'Schedule Trigger' node. On the left, the 'Parameters' tab is active, showing a 'Trigger Rules' section with a 'Trigger Interval' set to 'Minutes' and 'Minutes Between Triggers' set to '1'. An 'Add Rule' button is at the bottom. A red 'Execute step' button is in the top right. On the right, the 'OUTPUT' tab shows a JSON array with one item, containing timestamp and date information.

**Schedule Trigger Configuration:**

- Trigger Rules
  - Trigger Interval: Minutes
  - Minutes Between Triggers: 1
  - Add Rule

**OUTPUT (JSON):**

```
[{"timestamp": "2025-10-29T11:17:13.308-04:00", "Readable date": "October 29th 2025, 11:17:13 am", "Readable time": "11:17:13 am", "Day of week": "Wednesday", "Year": "2025", "Month": "October", "Day of month": "29", "Hour": "11", "Minute": "17", "Second": "13", "Timezone": "America/New_York (UTC-04:00)"}]
```

Una vez inicializado el nodo, hay que añadir un nodo HTTP Request para obtener los chistes desde la app JokeAPI. Para ello utilizaremos la URL dada.

The screenshot shows the Node-RED interface with the HTTP Request node configured. The left sidebar shows the 'INPUT' tab with a 'Schedule Trigger' node. The main workspace shows the 'HTTP Request' node with the following settings:

- Method: GET
- URL: `https://v2.jokeapi.dev/joke/Programming?type=single&amount=5`
- Authentication: None
- Send Query Parameters: ☐
- Send Headers: ☐
- Send Body: ☐
- Options: No properties

The right sidebar shows the 'OUTPUT' tab with the following JSON output:

```
{
  "error": false,
  "amount": 5,
  "jokes": [
    {
      "category": "Programming",
      "type": "single",
      "joke": "Eight bytes walk into a bar.\nThe bartender asks,\n\"Can I get you anything?\"\n\"Yeah,\" reply the\nbytes.\n\"Make us a double.\"",
      "flags": {
        "nsfw": false,
        "religious": false,
        "political": false,
        "racist": false,
        "sexist": false,
        "explicit": false
      },
      "id": 34,
      "safe": true,
      "lang": "en"
    },
    {
      "category": "Programming",
      "type": "single",
      "joke": "\"Honey, go to the store and buy some eggs.\n\\\"n\\\"OK.\\\"n\\\"Oh and while you're there, get some\nmilk.\\\"n\\\"He never returned.\"",
      "flags": {
        "nsfw": false,
        "religious": false,
        "political": false,
        "racist": false,
        "sexist": false,
        "explicit": false
      }
    }
  ]
}
```

El siguiente paso fue añadir el nodo Split Out para poder dividir el array jokes en elementos individuales, para así poder procesar cada chiste por separado.

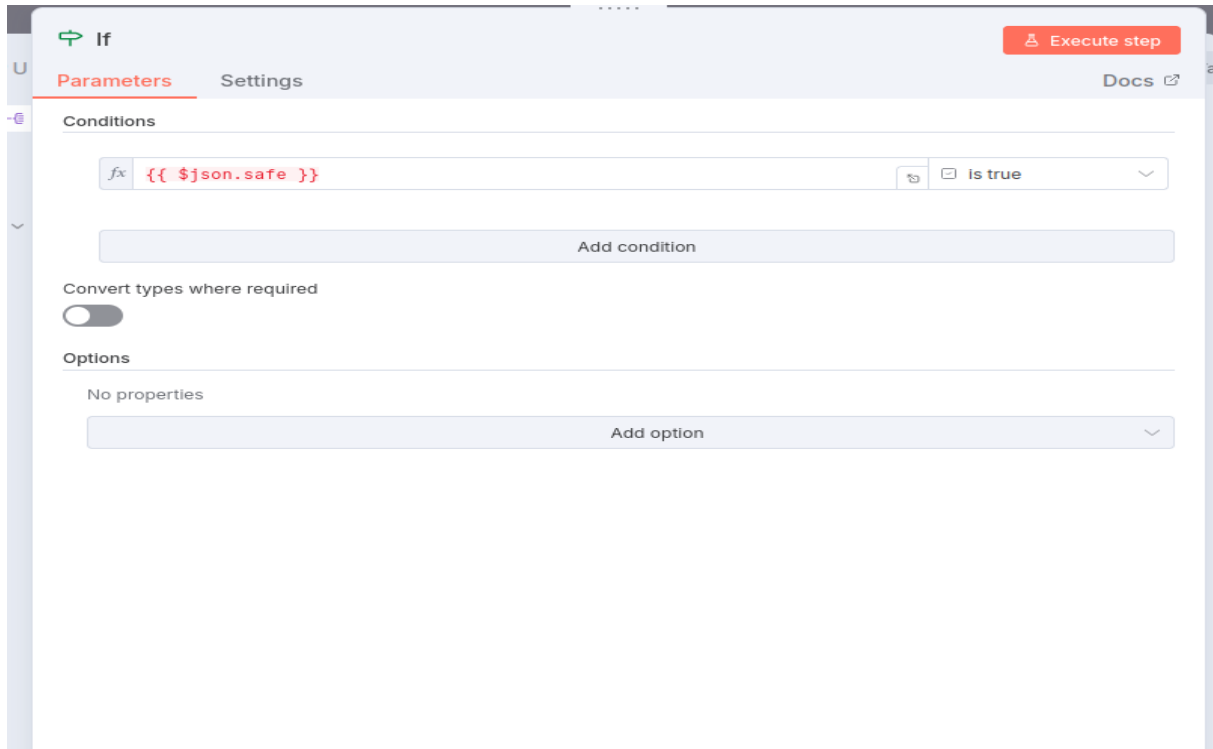
The screenshot shows the Node-RED interface with the Split Out node configured. The left sidebar shows the 'INPUT' tab with the 'HTTP Request' node. The main workspace shows the 'Split Out' node with the following settings:

- Fields To Split Out: jokes
- Use \$binary to split out the input item by binary data: ☐
- Include: All Other Fields
- Options: No properties

The right sidebar shows the 'OUTPUT' tab with the following JSON output:

```
{
  "error": false,
  "amount": 5,
  "jokes": [
    {
      "category": "Programming",
      "type": "single",
      "joke": "Eight bytes walk into a bar.\nThe bartender asks,\n\"Can I get you anything?\"\n\"Yeah,\" reply the\nbytes.\n\"Make us a double.\"",
      "flags": {
        "nsfw": false,
        "religious": false,
        "political": false,
        "racist": false,
        "sexist": false,
        "explicit": false
      },
      "id": 34,
      "safe": true,
      "lang": "en"
    },
    {
      "category": "Programming",
      "type": "single",
      "joke": "\"Honey, go to the store and buy some eggs.\n\\\"n\\\"OK.\\\"n\\\"Oh and while you're there, get some\nmilk.\\\"n\\\"He never returned.\"",
      "flags": {
        "nsfw": false,
        "religious": false,
        "political": false,
        "racist": false
      }
    }
  ]
}
```

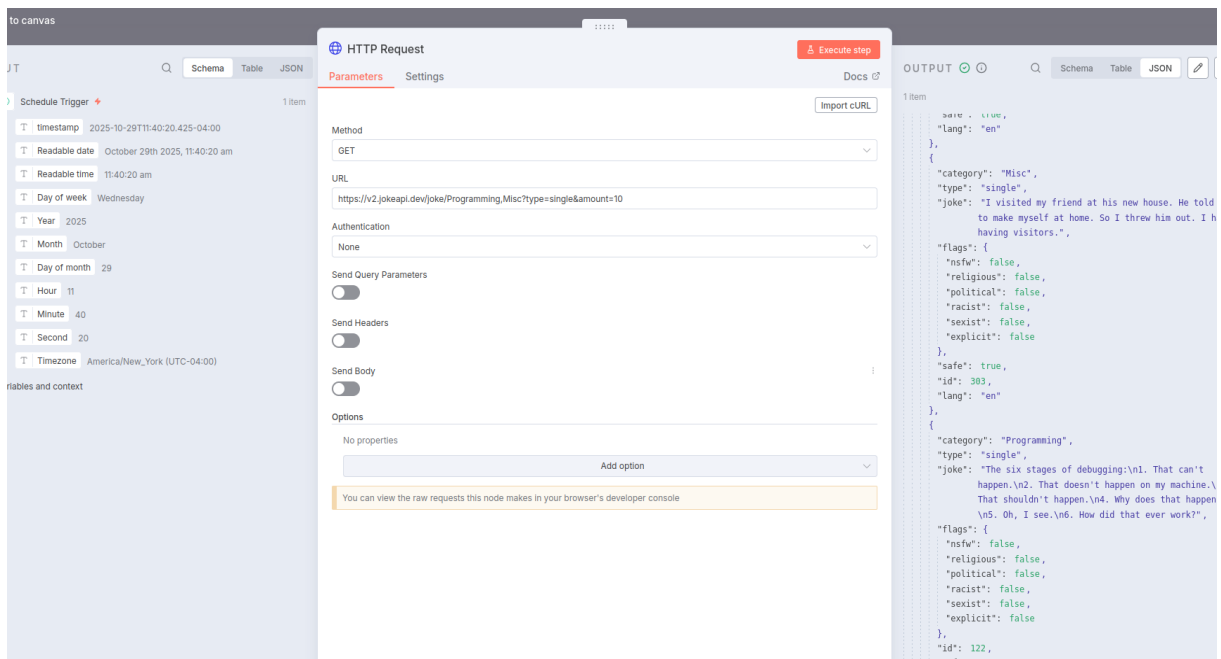
Para clasificar los chistes por su nivel de seguridad se añadió un nodo IF, donde se evaluó si el campo booleano safe es verdadero o no. En caso de ser verdadero se distribuirá a través de la rama true a un nodo Edit Fields (Set) donde se pondrá un mensaje de Chiste seguro, en el caso de que no sea verdadero se mandará a través de la rama false a un nodo Edit Fields (Set) donde se mostrará el mensaje Chiste NO seguro.



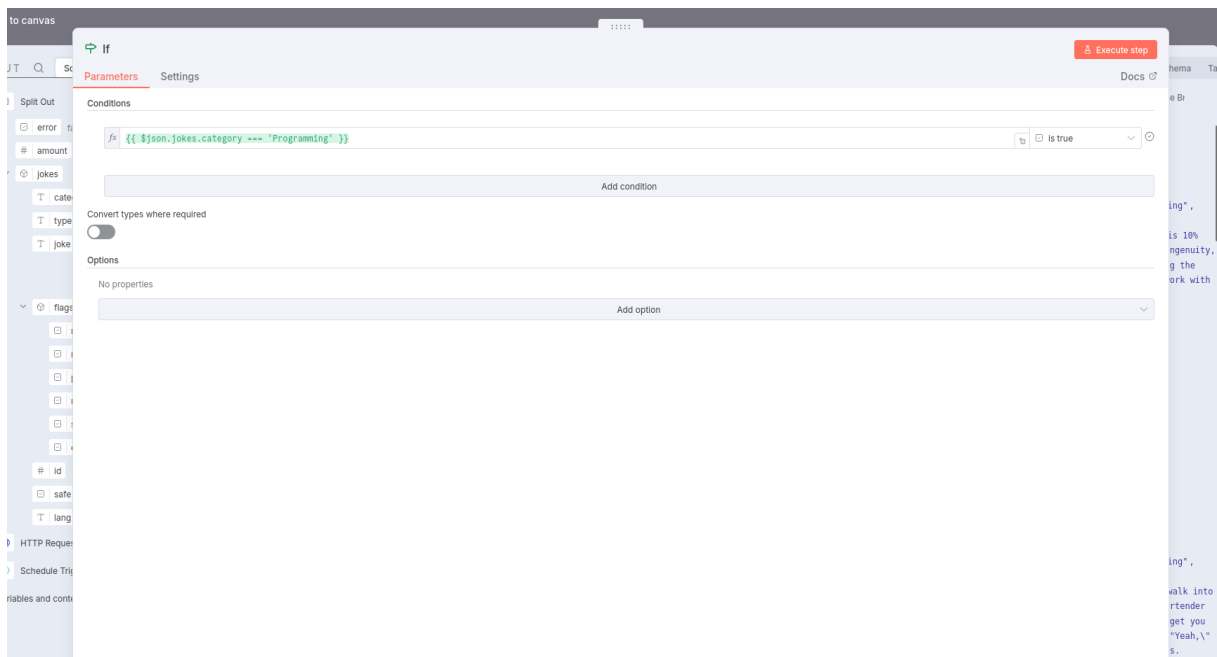
# Capítulo 2

## Ejercicio 1

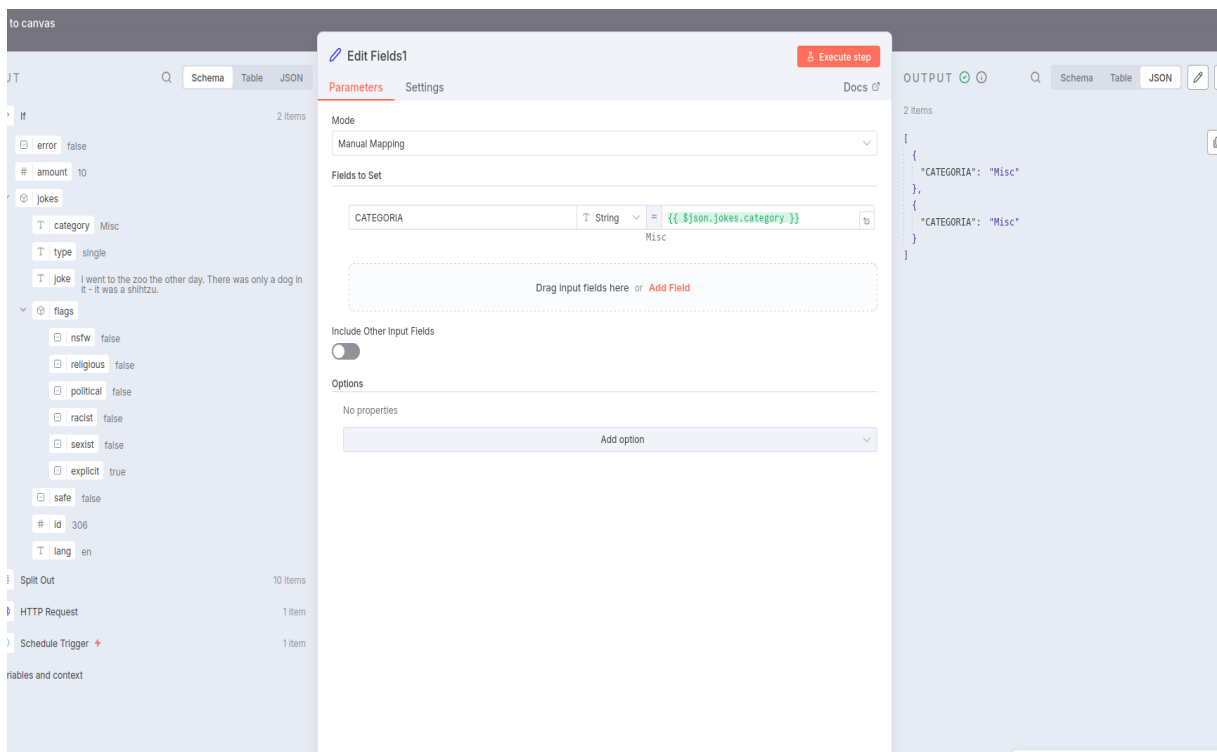
El objetivo del ejercicio 1 es clasificar los chistes por su categoría, diferenciando entre Programming y Misc. Para comenzar el flujo de trabajo es necesario añadir un nodo inicial Schedule Trigger. Este nodo se ha configurado para ejecutar el nodo automáticamente cada minuto. A continuación, añadimos un nodo HTTP Request para solicitar chistes de las dos categorías mediante la URL.



El siguiente paso es añadir un nodo IF para diferenciar los chistes por categorías. Por lo que deberemos configurar el nodo para que si la categoria del chiste es programming se vaya a la rama true y si es misc se vaya a la rama false.



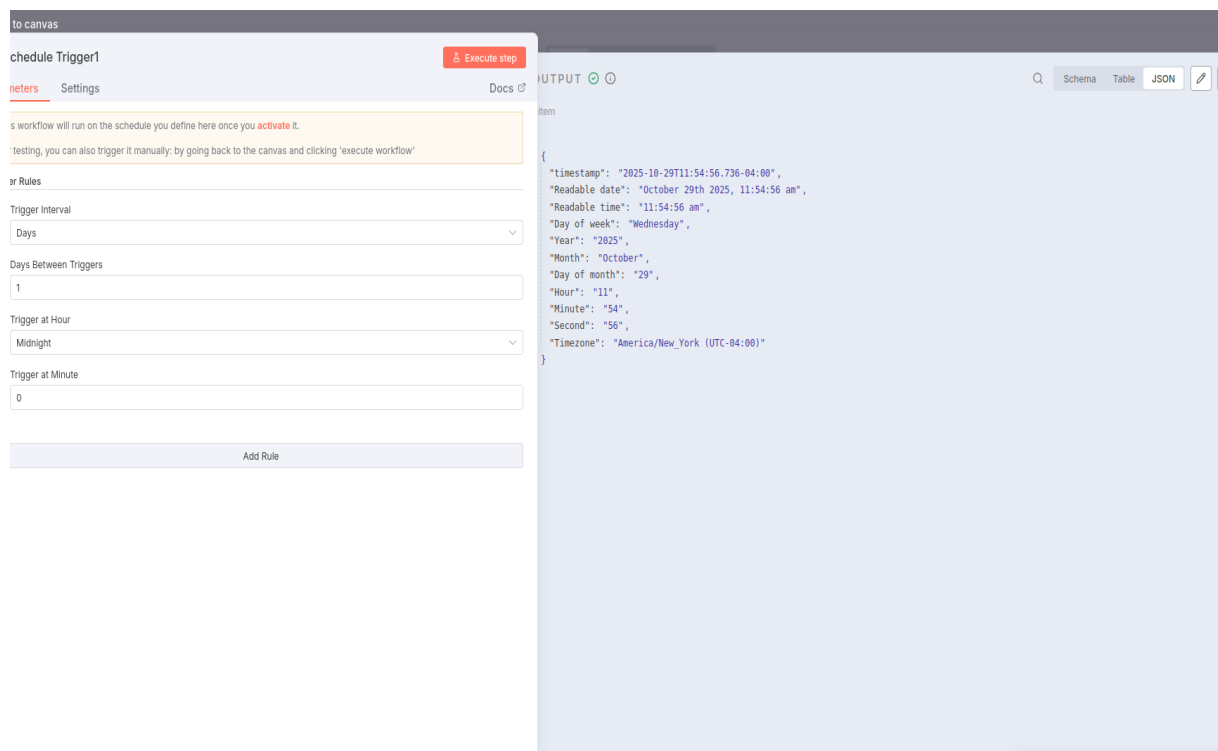
A continuación del nodo IF se añadirán un nodo Edit Fields (set) al final de cada rama para mostrar la categoria de cada chiste. En la siguiente foto se podrá ver los chistes de categoría Misc.



# Capítulo 3

## Ejercicio 2

El objetivo del ejercicio 2 es crear un flujo de trabajo que analice los productos de un comercio. Por lo que comenzaremos poniendo un nodo Schedule Trigger que se ejecute una vez al día.





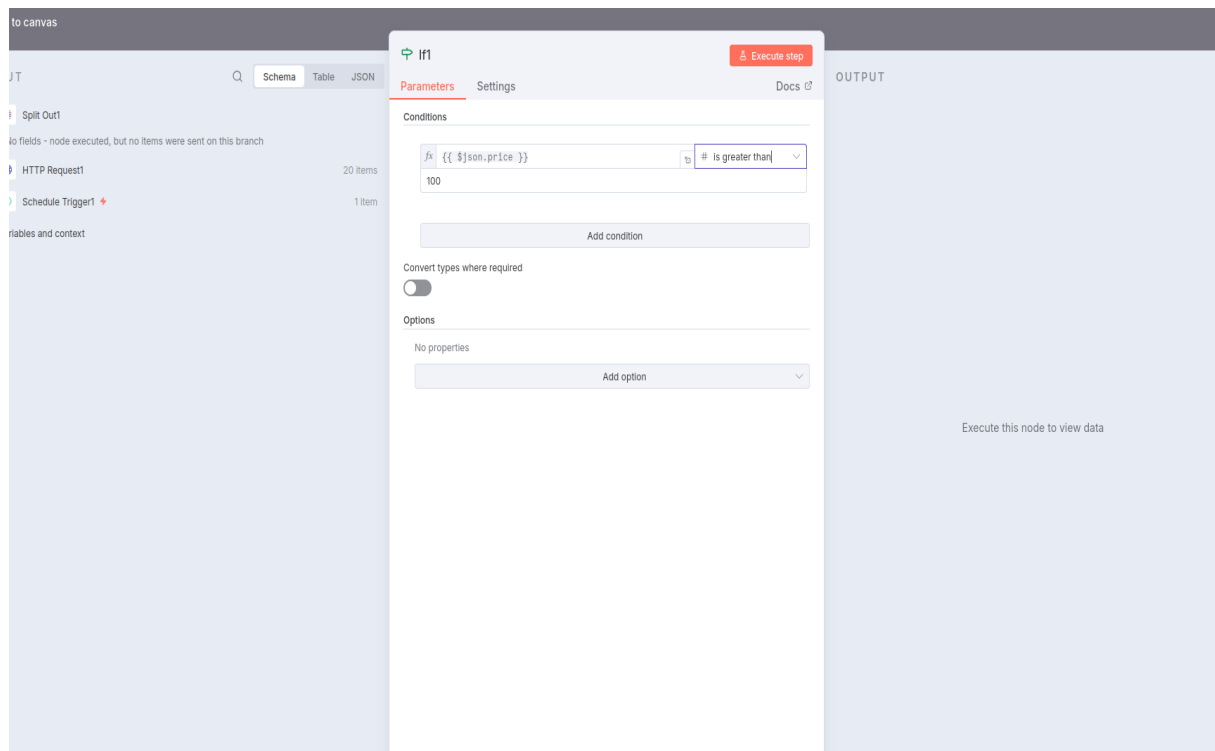
A continuacion, se añadió un nodo HTTP Request para realizar la solicitud a la API para obtener los datos mediante la URL.

The screenshot displays a workflow editor interface. On the left, a 'Schedule Trigger1' node is configured with a timestamp of '2025-10-29T11:53:24.001-04:00'. The main panel shows the 'HTTP Request1' node configuration. The 'Method' is set to 'GET' and the 'URL' is 'https://fakestoreapi.com/products'. The 'Authentication' is set to 'None'. The 'Send Query Parameters', 'Send Headers', and 'Send Body' options are all disabled. The 'Options' section is empty. On the right, the 'OUTPUT' panel shows a JSON array of 20 items. The first two items are visible:

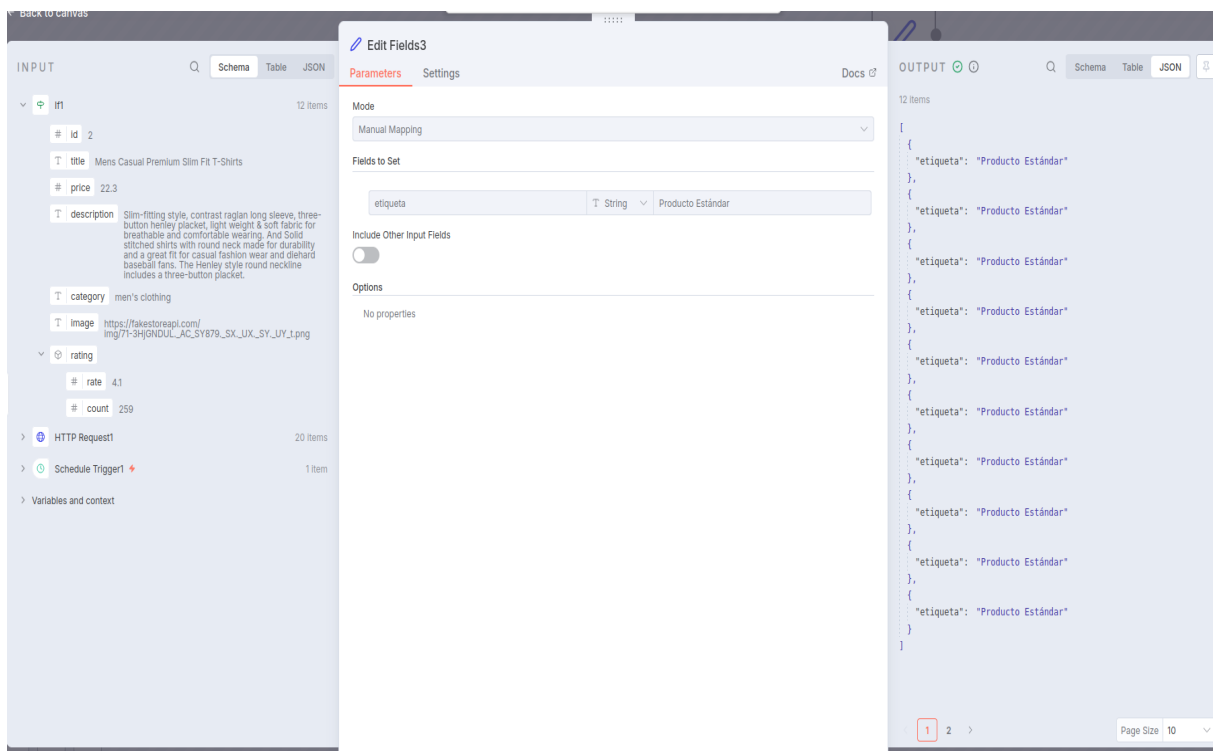
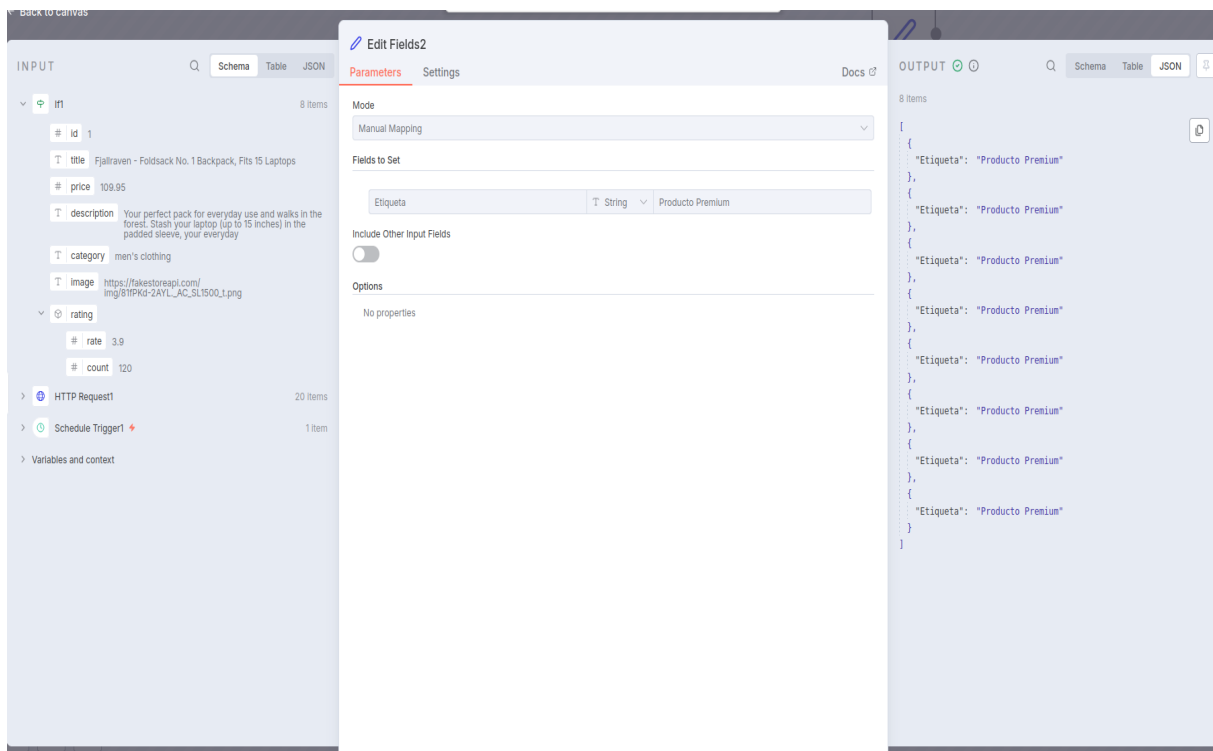
```
{
  "id": 1,
  "title": "Fjallraven - Foldsack No. 1 Backpack, Fits 15 Laptops",
  "price": 109.95,
  "description": "Your perfect pack for everyday use and walks in the forest. Stash your laptop (up to 15 inches) in the padded sleeve, your everyday",
  "category": "men's clothing",
  "image": "https://fakestoreapi.com/img/81fPKd-2AYL._AC_SL1500_t.png",
  "rating": {
    "rate": 3.9,
    "count": 120
  }
},
{
  "id": 2,
  "title": "Mens Casual Premium Slim Fit T-Shirts ",
  "price": 22.3,
  "description": "Slim-fitting style, contrast raglan long sleeve, three-button henley placket, light weight & soft fabric for breathable and comfortable wearing. And Solid stitched shirts with round neck made for durability and a good fit for casual fashion wear and diehard baseball fans. The Henley style round neck includes a three-button placket.",
  "category": "men's clothing",
  "image": "https://fakestoreapi.com/img/71-3HjGNDUL._AC_SY879._SX._UX._SY._UY_t.png",
  "rating": {
    "rate": 4.1,
    "count": 259
  }
}
```

The output panel shows a pagination bar with '1' selected, '2' next to it, and a 'Page Size' of '10'.

El siguiente paso consistió en añadir un nodo IF para realizar la comparación de cada producto. El objetivo es separar los productos en los que tienen un precio superior a 100 y los que tienen un precio igual o inferior.



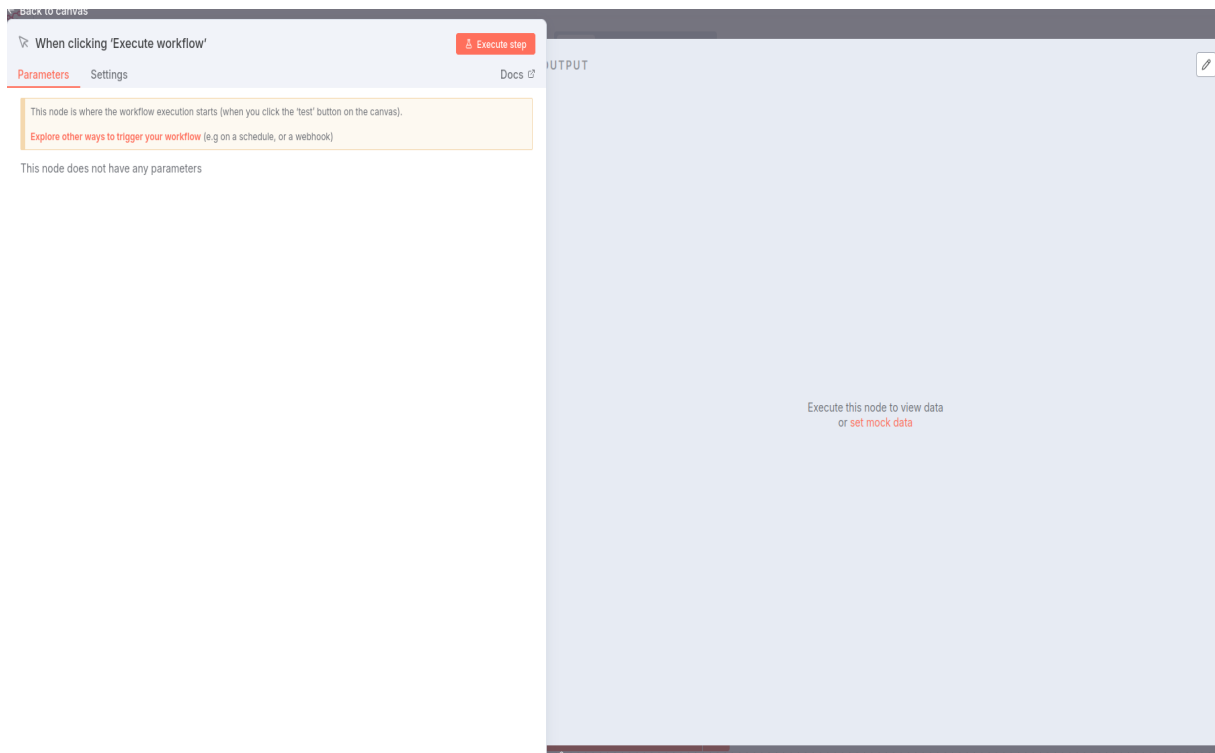
Una vez separados los productos por precio se añadirá dos nodos Edit Fields(Set) al final de cada rama, donde en la rama true tendremos los productos de más de 100 de valor y en la rama false los de menos de 100.



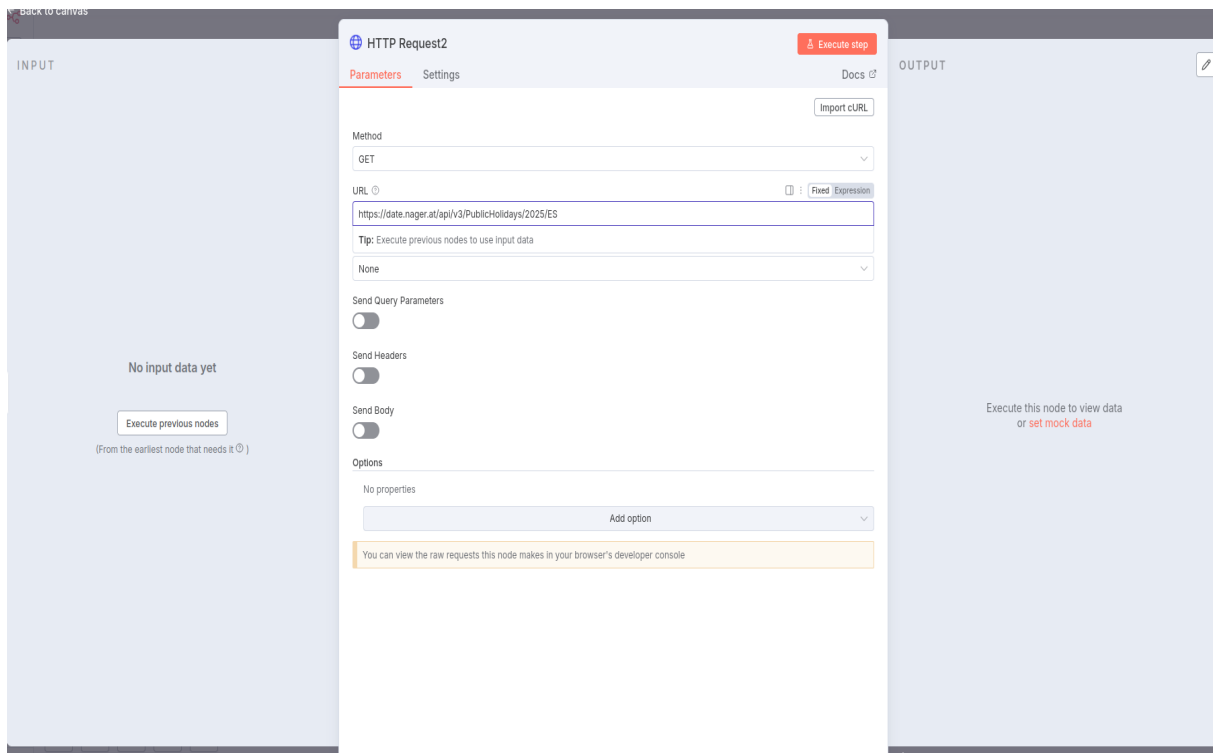
# Capítulo 4

## Ejercicio 3

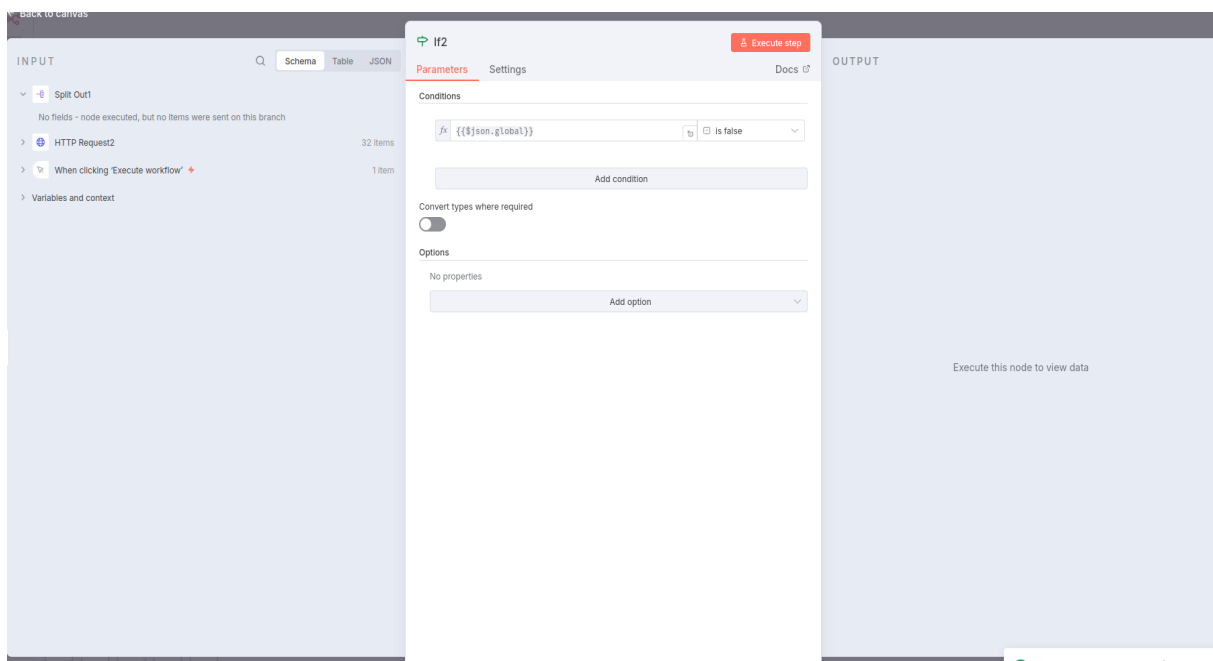
El objetivo del ejercicio 3 ha sido crear un flujo de trabajo que obtuviera los festivos nacionales y, únicamente a esos días, consultara un API para sugerir un actividad. Para ello se añadió un nodo Manual Trigger, que permite al usuario iniciar la ejecución del flujo manualmente.



El siguiente paso fue agregar un nodo HTTP Request para obtener los festivos oficiales en España.



Una vez obtenidos los festivos se añadirá un nodo IF para comprobar si el festivo es nacional o no. Para ello, se comprobará si el booleano global es false. Esto quiere decir que el festivo es local.



En la rama true se añadió un nodo HTTP Request para obtener las actividades aleatorias. Además, ha sido necesario usar un nodo Merge para poder conectar el HTTP Request con los festivos locales y el HTTP Request con las actividades aleatorias.

The screenshot shows the n8n workflow editor with a Merge node configured. The left pane shows the workflow structure with three inputs: HTTP Request3 (22 items), HTTP Request2 (32 items), and a trigger 'When clicking "Execute workflow"' (1 item). The Merge node is set to 'Combine' mode, 'Position' combine by, and '2' number of inputs. The right pane shows the output JSON, which is a list of two objects. The first object represents an activity with details like 'activity', 'availability', 'type', 'participants', 'price', 'accessibility', 'duration', 'kidFriendly', 'link', 'key', 'date', 'localName', 'name', 'countryCode', 'fixed', 'global', and 'counties'. The second object represents a social activity with details like 'activity', 'availability', 'type', 'participants', 'price', 'accessibility', 'duration', and 'kidFriendly'.

A continuación se creó un nodo Edit Fields(Set) para estructurar la salida del flujo donde se podrá ver el nombre del festivo y la actividad asignada a él.

The screenshot shows the n8n workflow editor with an Edit Fields4 node configured. The left pane shows the workflow structure with the Merge node followed by the Edit Fields4 node. The Edit Fields4 node is set to 'Manual Mapping' mode. The right pane shows the output JSON, which is a list of objects. Each object contains 'name' and 'activity' fields. The 'name' field is mapped from the 'localName' field of the input, and the 'activity' field is mapped from the 'activity' field of the input. The output shows a list of activities with their corresponding names, such as 'Día de Andalucía', 'Día de les Illes Balears', 'Jueves Santo', 'Lunes de Pascua', 'Día de Castilla y León', 'San Jorge (Día de Aragón)', 'Fiesta de la Comunidad de Madrid', 'Día das Letras Galegas', and 'Día de Canarias'.