Práctica 4: Robustez y Modularidad - Manejo de Errores y Sub-Flujos de Trabajo

Informe de Práctica

Elaborado por: Anass El Jabiry Kaddir

Fecha: 30 de noviembre de 2025

4. Ejercicios Prácticos

En esta sección pondremos a prueba los conocimientos adquiridos mejorando la funcionalidad de nuestros flujos de trabajo existentes.

4.1. Ejercicio 1: Notificación de Errores Mejorada

Dificultad: Baja

Objetivo

El objetivo de este ejercicio es evolucionar nuestro "Manejador de Errores" actual. Queremos que el sistema sea proactivo y, además de guardar un registro pasivo en Google Sheets, nos envíe una notificación inmediata por correo electrónico cuando algo falle.

Requisitos de Implementación

Para completar el ejercicio, se deben seguir los siguientes pasos en el flujo de trabajo:

- **Modificación del Flujo:** Abra el flujo de trabajo "Manejador de Errores" existente.
- **Nuevo Nodo:** Añada un nodo **Send Email** (o alternativamente Slack/Telegram) conectado al flujo.
- **Configuración del Asunto:** Establezca un asunto claro y alarmante, por ejemplo: "*¡Alerta de Error en n8n!*".
- **Cuerpo del Mensaje:** Incluya los detalles técnicos del error utilizando las expresiones del Error Trigger.

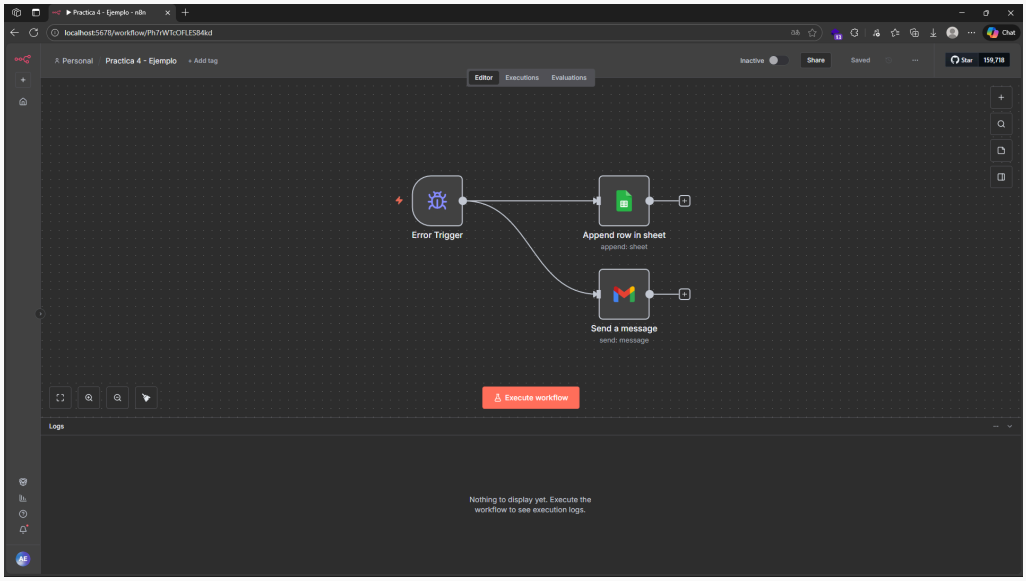

Figura 1: Diagrama del flujo integrando el nodo de notificación por correo.

Detalle de la Configuración del Correo

Para que la notificación sea útil, asegúrese de incluir las siguientes expresiones en el campo "Text" o "HTML" del nodo de correo:

Dato a Incluir	Expresión Recomendada
Nombre del Flujo	`Workflow: {{ $json.workflow.name }}`
Nodo que falló	`Nodo: {{ $json.execution.lastNodeExecuted }}`
Mensaje de Error	`Error: {{ $json.execution.error.message }}`
Link de revisión	`Link: {{ $json.execution.url }}`

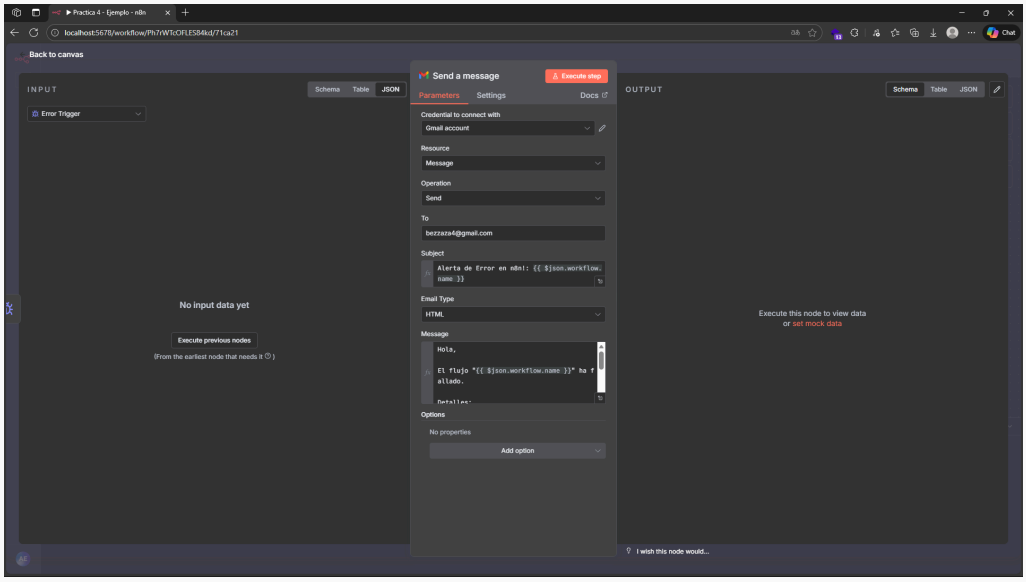

Figura 2: Vista de configuración del nodo Send Email con las variables dinámicas.

4.2. Ejercicio 2: Sub-Flujo de Validación de Datos

Dificultad: Media

Objetivo

Crear un componente modular reutilizable (sub-flujo) encargado de asegurar la integridad de los datos. Validaremos que la estructura de un producto contenga los campos esenciales antes de procesarlo.

API a utilizar: Fake Store API (Producto 1)

Requisitos de Implementación

Parte A: El Sub-Flujo "Validador de Productos"

1. Cree un nuevo flujo llamado **"Sub-flujo: Validador de Productos"**.
2. Inicie con un nodo **Execute Workflow Trigger** para permitir llamadas externas.
3. Implemente un nodo **IF** con lógica `AND` para verificar que existan y no estén vacíos los campos:
   - `title`
   - `price`
4. En la rama *False* (si falta algún campo), conecte un nodo **Stop and Error**.
5. Configure el mensaje de error: "*Validación fallida: Faltan campos de producto obligatorios*".

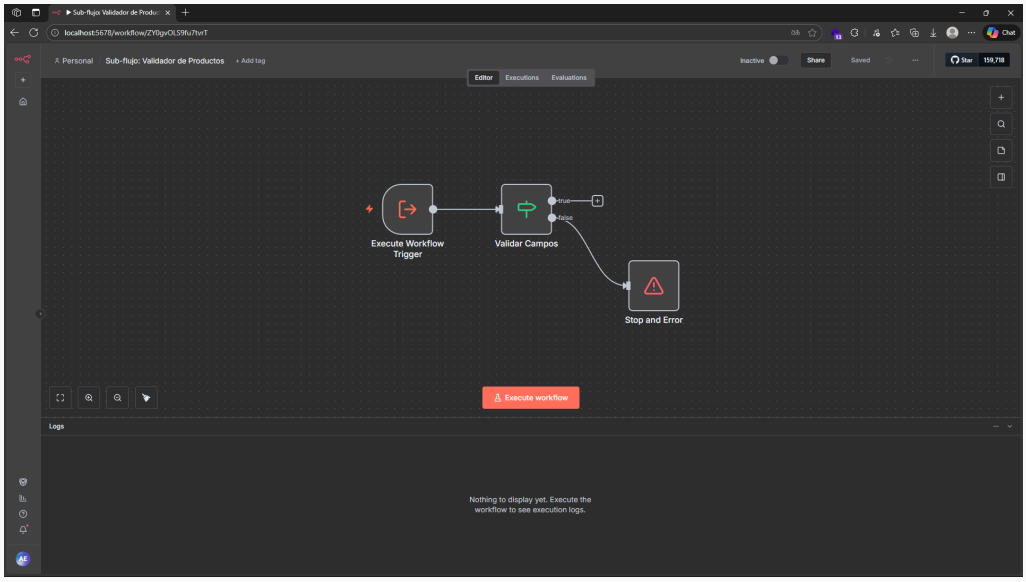

Figura 3: Lógica del sub-flujo verificando campos con el nodo IF y Stop and Error.

Parte B: El Flujo Principal y Prueba de Estrés

1. En el flujo principal, realice una petición HTTP a la Fake Store API para obtener un producto.
2. **Simulación de Error:** Inserte un nodo **Edit Fields (Set)** para *eliminar* intencionalmente el campo `price` del JSON recibido.
3. Llame al sub-flujo usando el nodo **Execute Workflow**.
4. Verifique que el "Manejador de Errores" (configurado previamente) capture este fallo personalizado y lo registre en Google Sheets.

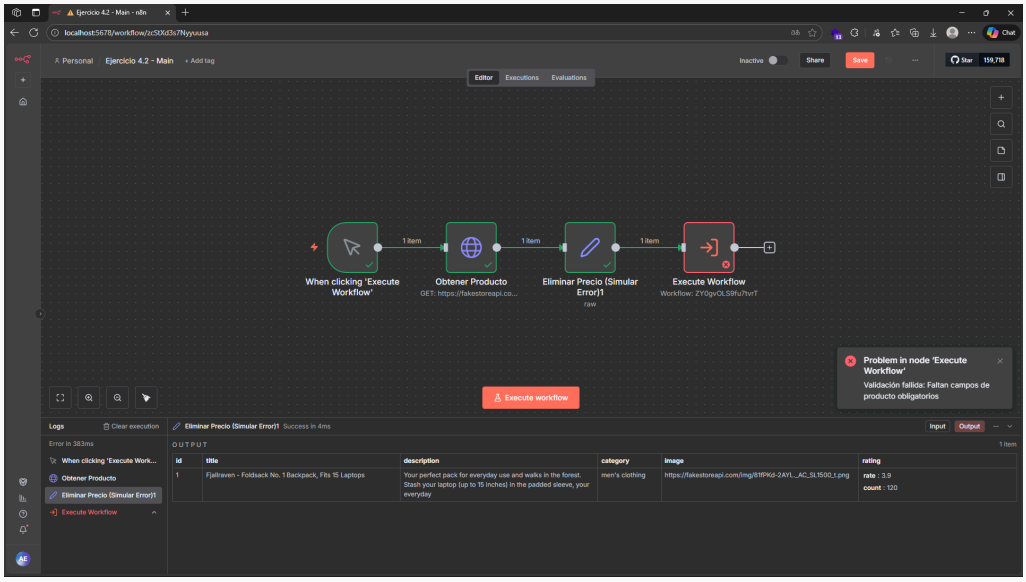

Figura 4: Flujo principal modificando los datos antes de invocar al validador.

4.3. Ejercicio 3: Orquestador de Tareas Dinámicas

Dificultad: Alta

Objetivo

Implementar un patrón de orquestación donde un flujo principal actúa como "controlador de tráfico", decidiendo dinámicamente qué sub-flujo ejecutar basándose en una variable de control.

APIs a utilizar:

- Bored API (Actividades aleatorias)
- Public Holidays API (Festivos España 2025)

Requisitos de Implementación

1. Preparación de los Sub-Flujos (Workers)

Cree dos flujos de trabajo sencillos que servirán como trabajadores especializados:

- **"Sub-flujo: Obtener Actividad":** Realiza una petición HTTP GET a la Bored API y devuelve la respuesta JSON.
- **"Sub-flujo: Obtener Festivos":** Realiza una petición HTTP GET a la API de festivos y devuelve la lista.

2. Configuración del Orquestador (Main Workflow)

El flujo principal debe contener la lógica de decisión:

1. Inicie con un **Manual Trigger**.
2. Utilice un nodo **Edit Fields (Set)** para definir la variable de control. *Ejemplo:* Cree un campo String llamado `tarea` con el valor `"actividad"`.
3. Añada un nodo **Switch** conectado al Set. Configure la lógica para evaluar el valor de `tarea`:
    - **Ruta 0 (Actividad):** Si `tarea` es igual a "actividad", conecte un nodo **Execute Workflow** que llame al "Sub-flujo: Obtener Actividad".
    - **Ruta 1 (Festivos):** Si `tarea` es igual a "festivos", conecte un nodo **Execute Workflow** que llame al "Sub-flujo: Obtener

Festivos".

4. Conecte ambas salidas de los nodos Execute Workflow a un único nodo **Merge** para consolidar el resultado final en una sola rama.
5. Finalice con un nodo **Edit Fields (Set)** para visualizar la salida limpia.

Prueba de Concepto: Para verificar el funcionamiento, cambie manualmente el valor de la variable `tarea` a "actividad" y vuelva a ejecutar el flujo. El sistema debería activar automáticamente la ruta alternativa.

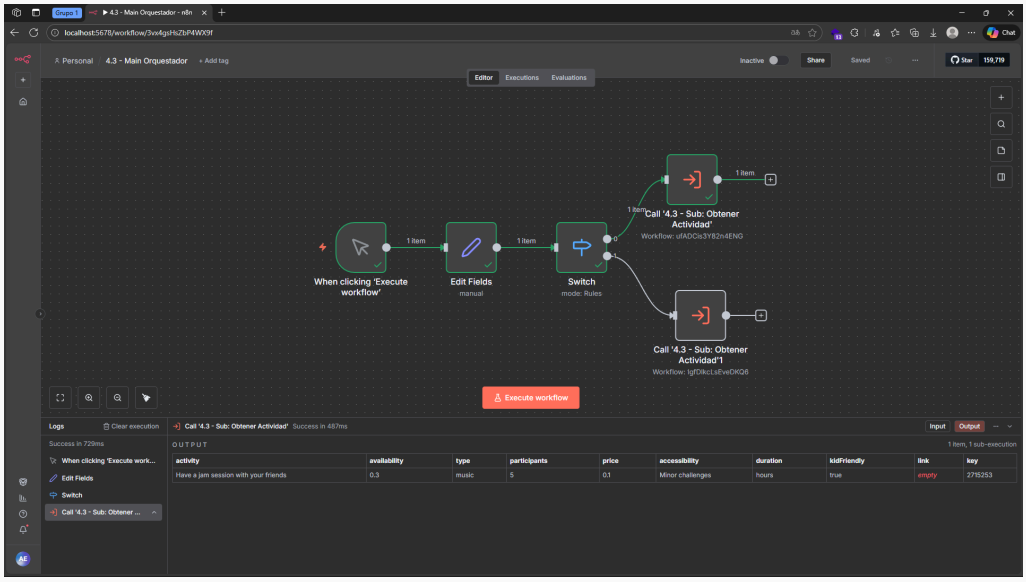

Figura 5: Flujo Orquestador utilizando un Switch para delegar tareas a sub-flujos específicos.