

INTELIGENCIA ARTIFICIAL GENERATIVA

# Orquestación de Modelos de IA

Práctica 7: Integración avanzada con Google Gemini Pro, gestión de contextos y flujos multimodales.

## TECNOLOGÍAS

-  Google Gemini API
-  n8n / LangChain

## REPOSITORIO

 [github.com/aeck676/itsi-2026](https://github.com/aeck676/itsi-2026)

## AUTOR

**Anass El Jabiry Kaddir**

## FECHA

8 de diciembre de 2025

# Preparación del Entorno

01

## Paso 1: Credenciales y API Key

### PASO 1: OBTENER CREDENCIALES

Configuración de la API Key obtenida en Google AI Studio dentro del gestor de credenciales de n8n (Tipo: Google Gemini API).

#### 1. GOOGLE AI STUDIO (API KEY)

The screenshot shows a table with one row of data. The columns are: Clave, Proyecto, Fecha de creación, and Nivel de cuota.

Clave	Proyecto	Fecha de creación	Nivel de cuota
...WvDI n8n	n8n-itsi gen-lang-client-0974591906	7 dic 2025	<a href="#">Configurar la facturación</a> Nivel gratuito

#### 2. CREDENCIALES EN N8N

The screenshot shows the n8n interface with a modal window open for a "Google Gemini (Tasks)" connection. The modal has a green success message: "Connection tested successfully". It includes fields for "Host" (https://generativelanguage.googleapis.com) and "API Key" (redacted). There is also a dropdown for "Allowed HTTP Request Domains" set to "All". A note at the bottom says "Enterprise plan users can pull in credentials from external vaults. More info".

# Ejercicio Guiado

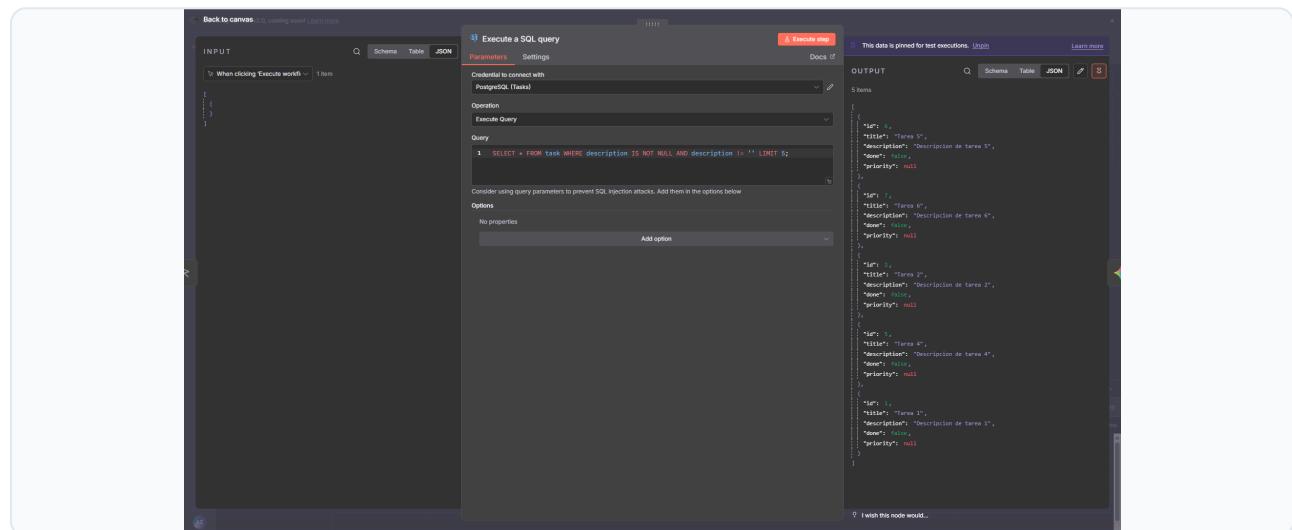
Parte 1: Clasificación Automática de Tareas

## OBJETIVO

Crear un flujo que lea tareas de PostgreSQL, utilice Gemini para determinar su prioridad (ALTA, MEDIA, BAJA) basándose en la descripción, y tome decisiones según esa clasificación.

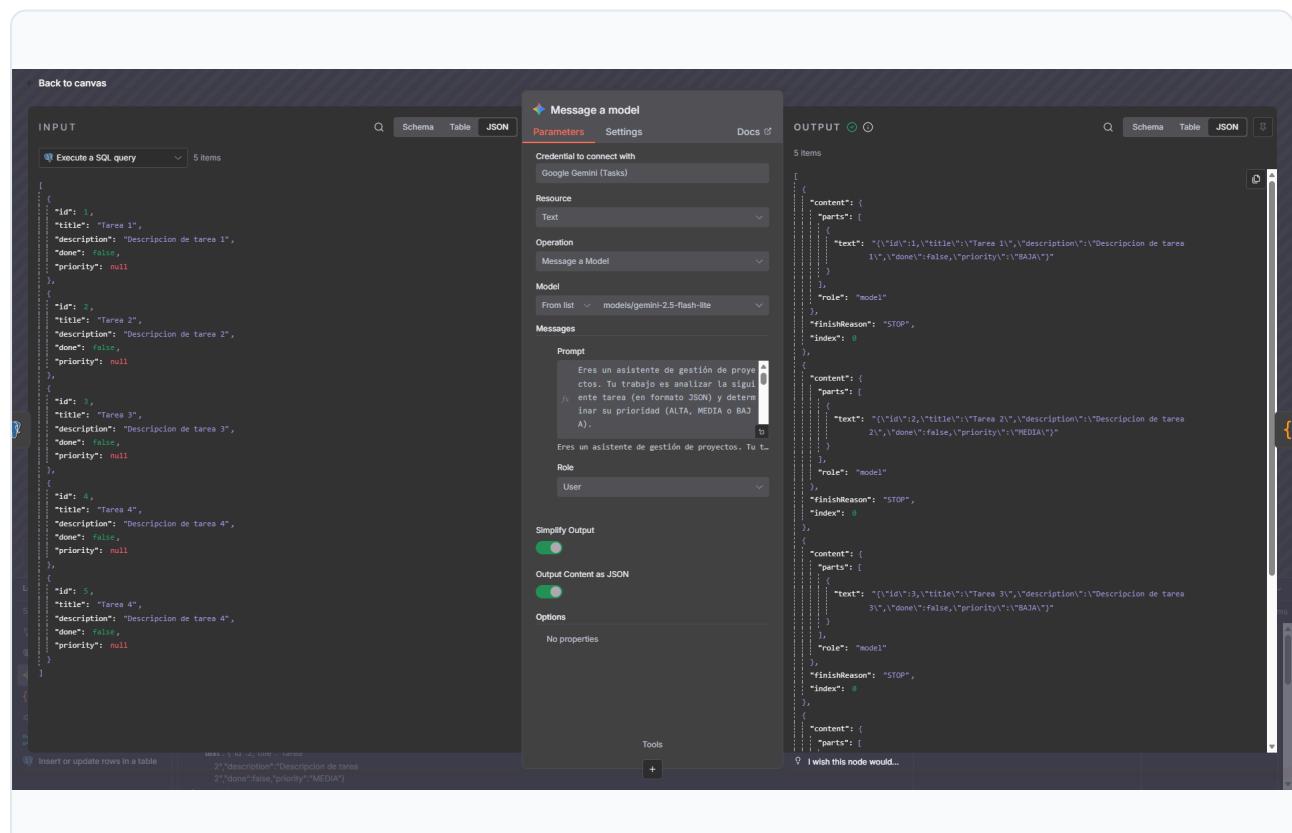
## PASO 2: LECTURA DE TAREAS (POSTGRESQL)

Consulta: SELECT \* FROM task ... LIMIT 5



### PASO 3: PROMPT EN GEMINI

## Configuración del Prompt para clasificar prioridad



# Ejercicio Guiado

02

## Parte 2: Decisión, Notificación y Actualización

### IMPLEMENTACIÓN DE LÓGICA

Enrutamiento basado en la respuesta de la IA y actualización posterior en base de datos.

### PASO 4: SWITCH (LÓGICA DE DECISIÓN)

The screenshot shows a workflow editor interface with a central node configuration panel titled "Switch". The "Parameters" tab is selected. The "Mode" dropdown is set to "Rules". There are two routing rules defined:

- Rule 1: Condition `{{{ $json.priority.trim() }}} T` is equal to ALTA. The output is labeled "ALTA".
- Rule 2: Condition `{{{ $json.priority.trim() }}} T` is equal to MEDIA. The output is labeled "MEDIA".

The "Output" tab on the right shows two outputs:

- Output 0: "ALTA" (2 items)
- Output 1: "MEDIA" (3 items)

The "Input" tab on the left shows a JSON array of task data:

```
[{"id": 1, "title": "Tarea 1", "description": "Descripción de tarea 1", "done": false, "priority": "BAJA"}, {"id": 2, "title": "Tarea 2", "description": "Descripción de tarea 2", "done": false, "priority": "MEDIA"}, {"id": 3, "title": "Tarea 3", "description": "Descripción de tarea 3", "done": false, "priority": "BAJA"}, {"id": 4, "title": "Tarea 4", "description": "Descripción de tarea 4", "done": false, "priority": "BAJA"}, {"id": 5, "title": "Tarea 4", "description": "Descripción de tarea 4", "done": false, "priority": "MEDIA"}]
```

### PASO 5: NOTIFICACIÓN Y UPDATE (AVANZADO)

The screenshot shows a workflow editor interface with a central node configuration panel titled "Send a message". The "Parameters" tab is selected. The "Resource" dropdown is set to "Message". The "Operation" dropdown is set to "Send". The "To" field contains the email address "aek676@inlumine.ul.es". The "Subject" field contains the message "¡Alerta de Tarea Prioridad ALTA!". The "Email Type" dropdown is set to "Text". The "Message" field contains the text "La tarea \"{{ \$json.title }}\" (ID: {{ \$json.i... }}) ha sido clasificada como ALTA priorida... d por la IA." The "Output" tab on the right shows a single output: "Execute this node to view data".

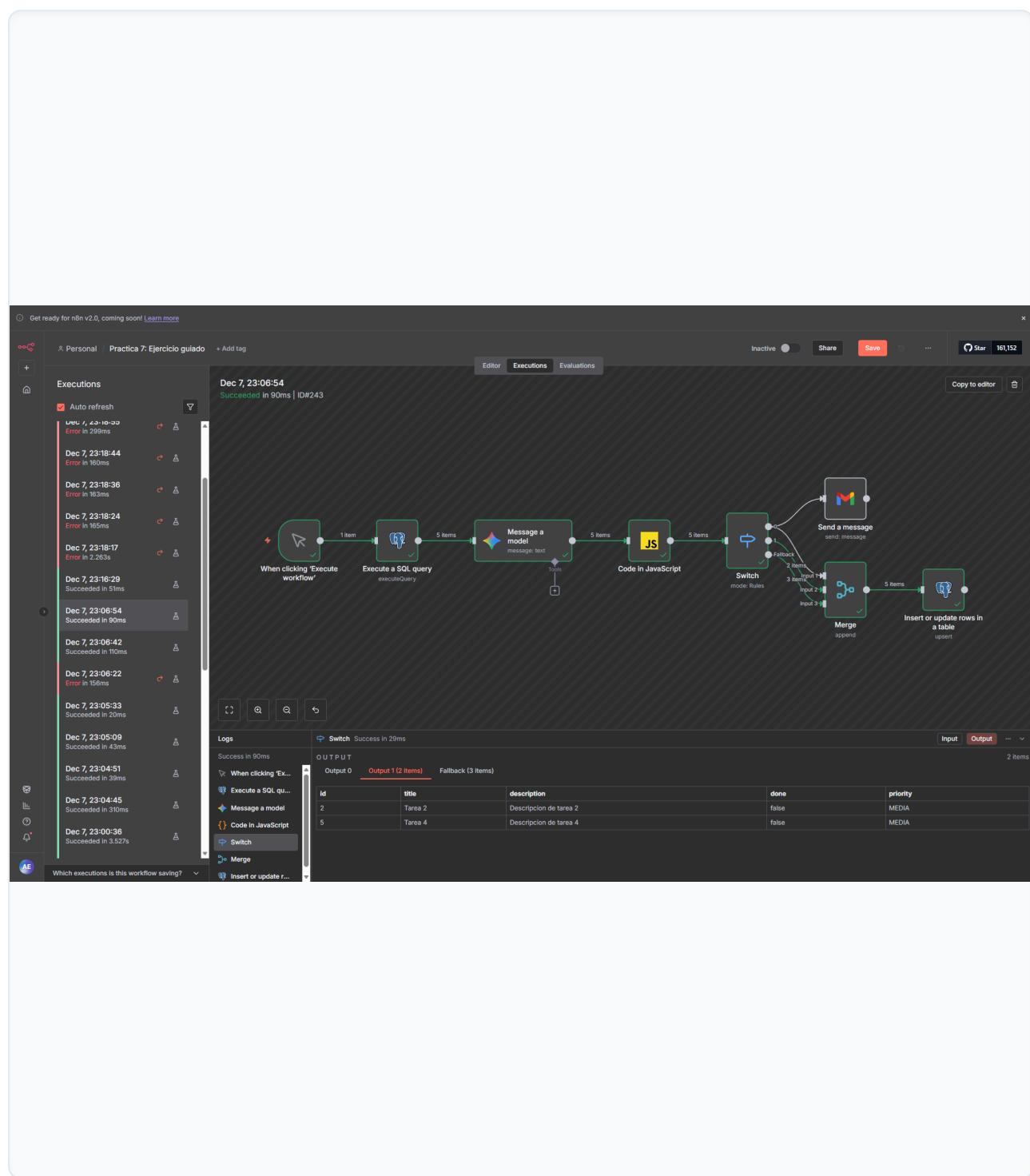
# Ejercicio Guiado

02

## Parte 3: Visión Global del Workflow

### DIAGRAMA COMPLETO

Diagrama completo que muestra la integración de lectura, análisis con IA, decisión lógica y acciones finales (email y actualización DB).



# Ejercicio 1: Resúmenes IA

03

Adaptación para Sumarización de Tareas

## OBJETIVO

Modificar el flujo guiado para solicitar a Gemini un **resumen conciso** (máx. 15 palabras) de la tarea. Se reemplaza el Switch por un nodo Set para guardar el resultado.

### 1. PROMPT DE RESUMEN

The screenshot shows the Streamlit interface with the 'Message a model' step selected. The input is a list of tasks (JSON array). The output is a JSON object with five items, each containing a summary generated by Gemini. The summaries are:

- "Descripción de la tarea 5: Resumen conciso de una sola frase."
- "Descripción de la tarea 6: Resumen conciso de una sola frase."
- "Descripción de tarea 2: Breve resumen de la segunda tarea."
- "Descripción de tarea 4: Breve descripción de la tarea solicitada."
- "Descripción de la tarea 1: Resumen conciso de la tarea 1."

### 2. NUEVO DISEÑO DEL FLUJO

The screenshot shows the Streamlit interface with the 'Edit Fields' step selected. The input is the same list of tasks. The output is a JSON object with five items, each containing a summary generated by Gemini. The summaries are identical to the ones in the previous screenshot. The 'Edit Fields' step is configured to map the 'resumen' field from the Gemini output to a new 'resumen' field in the final output.

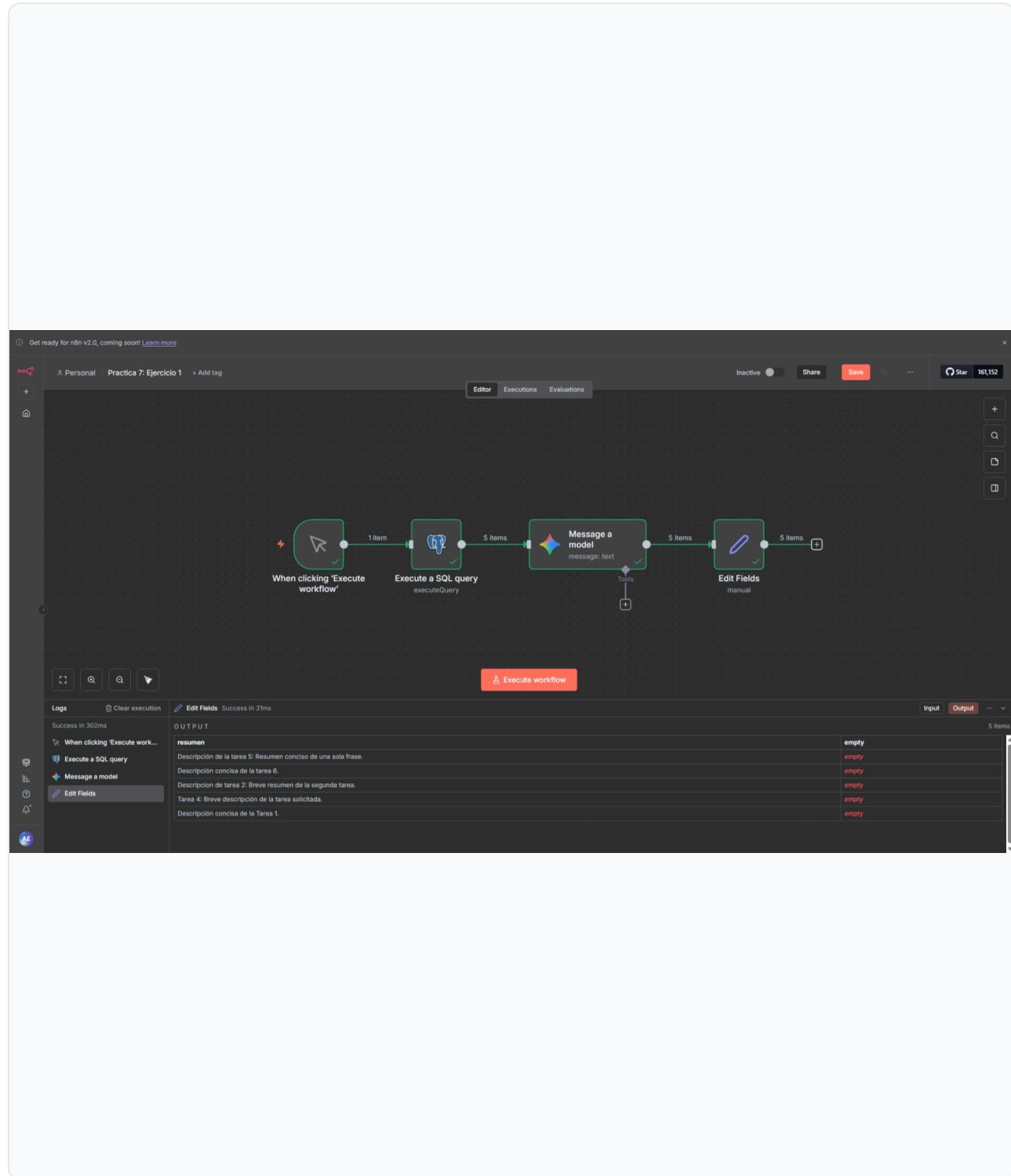
# Ejercicio 1: Resúmenes IA

03

## Resultado de la Ejecución

A continuación se muestra el resultado final, donde se observa el campo resumen generado por la IA en el JSON de salida.

### 3. EJECUCIÓN Y OUTPUT



# Ejercicio 2: Extracción

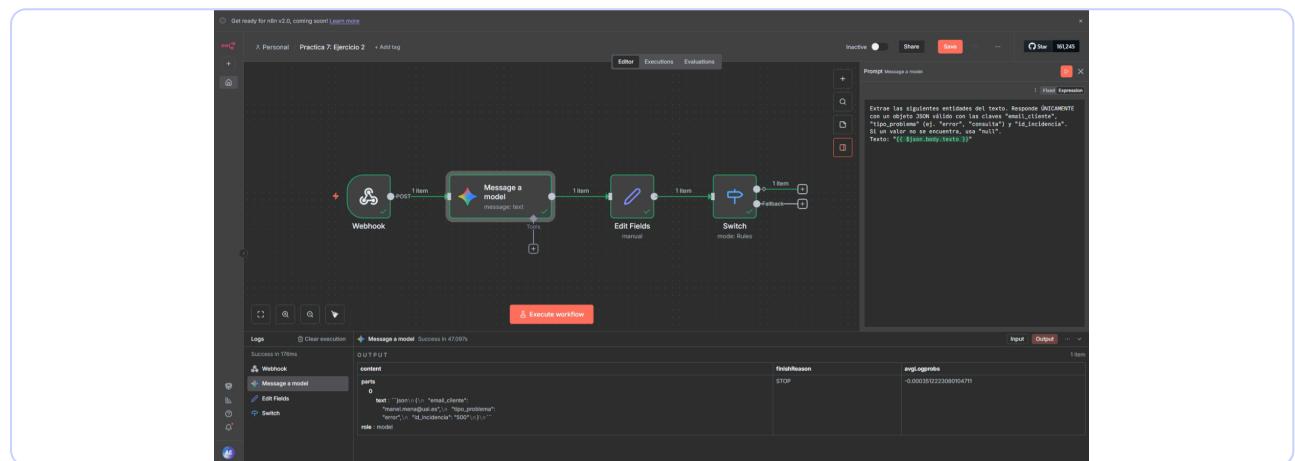
04

## Extracción de Entidades y Formato JSON

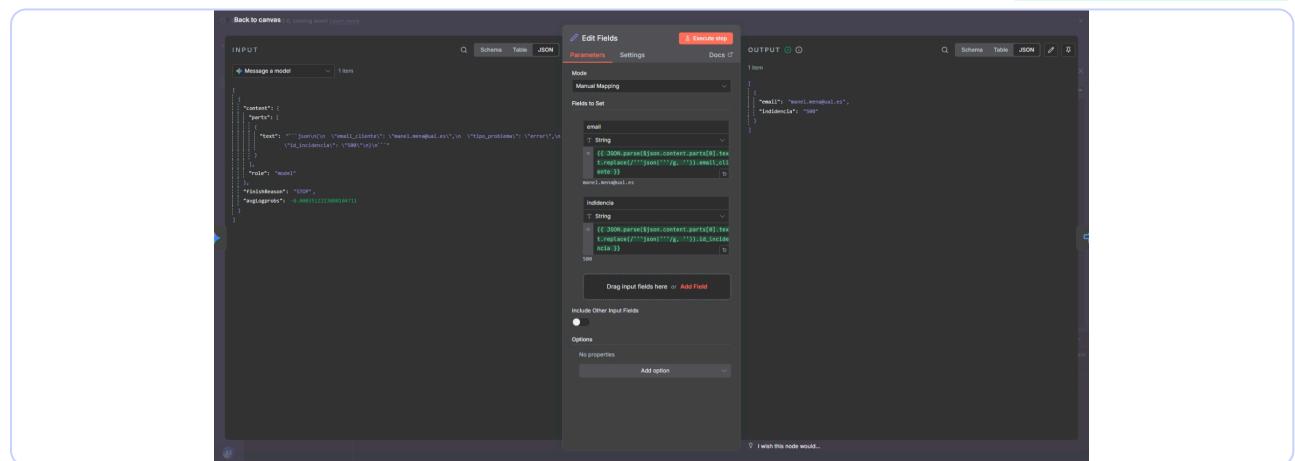
### OBJETIVO

Construir un flujo que reciba texto mediante un **Webhook**, use Gemini para extraer datos estructurados (email, tipo problema, id incidencia) en formato JSON, y enrute la lógica según el dominio del correo (@ual.es).

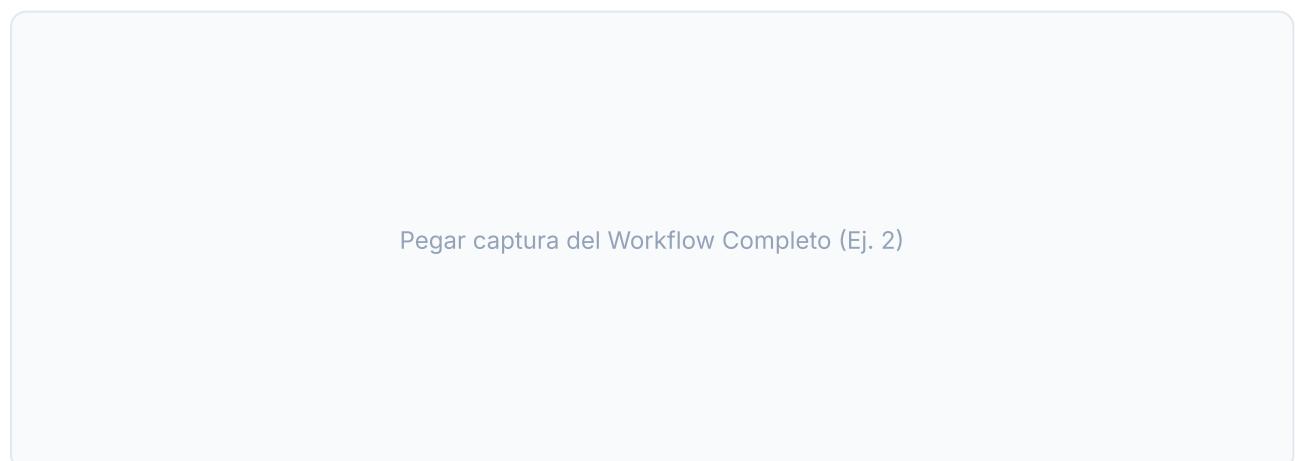
### 1. PROMPT DE EXTRACCIÓN JSON



### 2. PARSEO Y LIMPIEZA (SET NODE)



### 3. DISEÑO DEL WORKFLOW



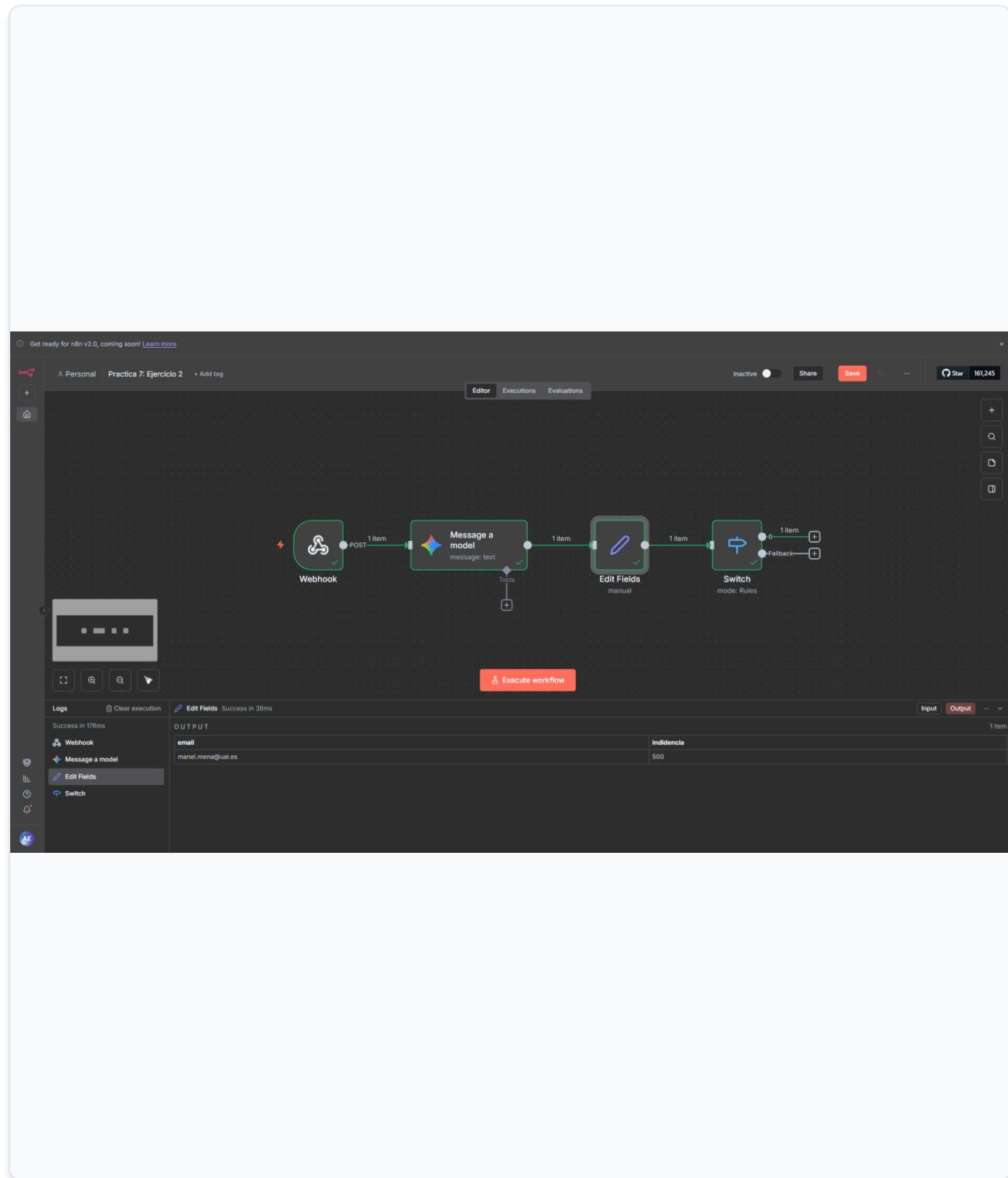
# Ejercicio 2: Extracción

04

Resultado de la Ejecución

Resultado de la ejecución del flujo: se observa cómo Gemini extrae los campos clave del texto desestructurado y el nodo Switch enruta correctamente la lógica.

## 3. EJECUCIÓN Y OUTPUT



# Ejercicio 3: AI Chain

05

Generación de Contenido en Cascada

## OBJETIVO (DIFICULTAD: ALTA)

Implementar una cadena de IA donde la salida de un modelo sirve de entrada para el siguiente.

1. Generar 3 ideas de títulos de blog (Lista).
2. Dividir la lista en ítems individuales (JavaScript).
3. Generar una introducción para cada título (Iteración).

## 1. DISEÑO DEL WORKFLOW COMPLETO

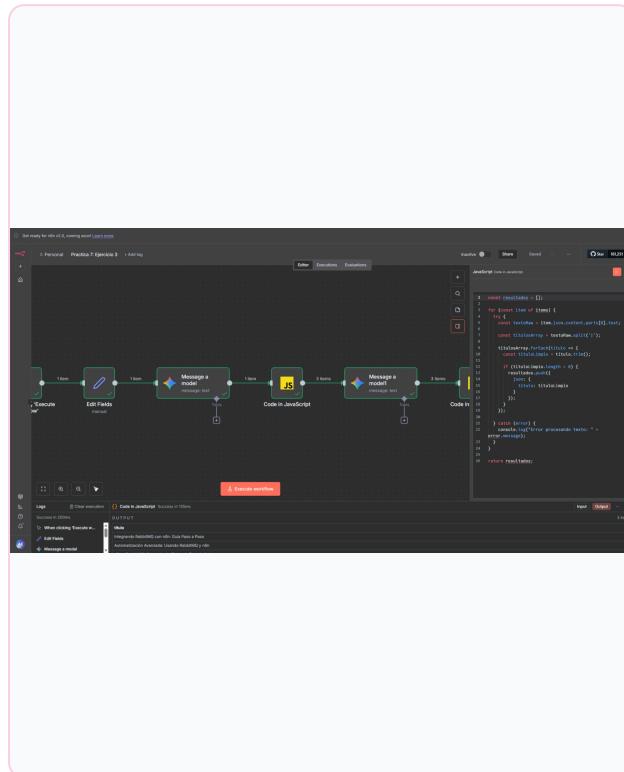
The screenshot shows a workflow editor interface with a central node configuration panel. The node is titled 'Message a model'. In the 'Parameters' tab, the 'Credential to connect with' is set to 'Google Gemini (Tasks)', 'Resource' is 'Text', 'Operation' is 'Message a Model', and 'Model' is 'From list' with 'models/gemini-2.0-flash-lite' selected. The 'Messages' section contains a prompt: 'Genera 3 ideas para un título de un artículo de blog sobre el tema: "[{{}}]"'. Below this, there is a 'Role' dropdown set to 'User'. The 'Output Content as JSON' toggle is turned on. The 'Simplify Output' toggle is turned on. The 'Options' section shows 'No properties'. On the right side of the node, under the 'OUTPUT' tab, the resulting JSON output is displayed:

```
[{"content": {"parts": [{"text": "Integrando RabbitMQ con n\u00f3n: Guia Paso a Paso | Automatizaci\u00f3n Avanzada: Usando RabbitMQ y\nn\u00f3n y RabbitMQ: Orquestando Flujos de Trabajo Asincronos\\n"}]}, "role": "model"}, {"finishReason": "STOP", "avgLogprob": -0.34646344843880086} ]
```

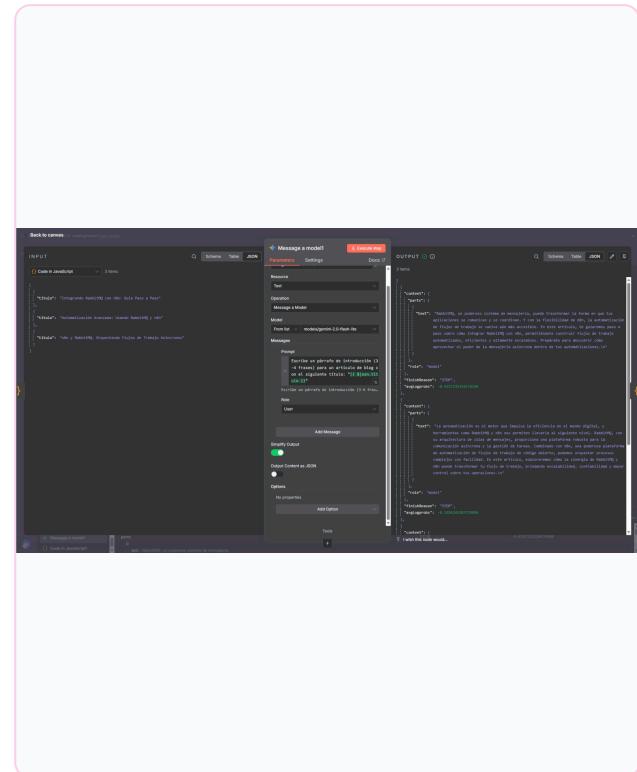
# Ejercicio 3: AI Chain

## Detalles de Implementación

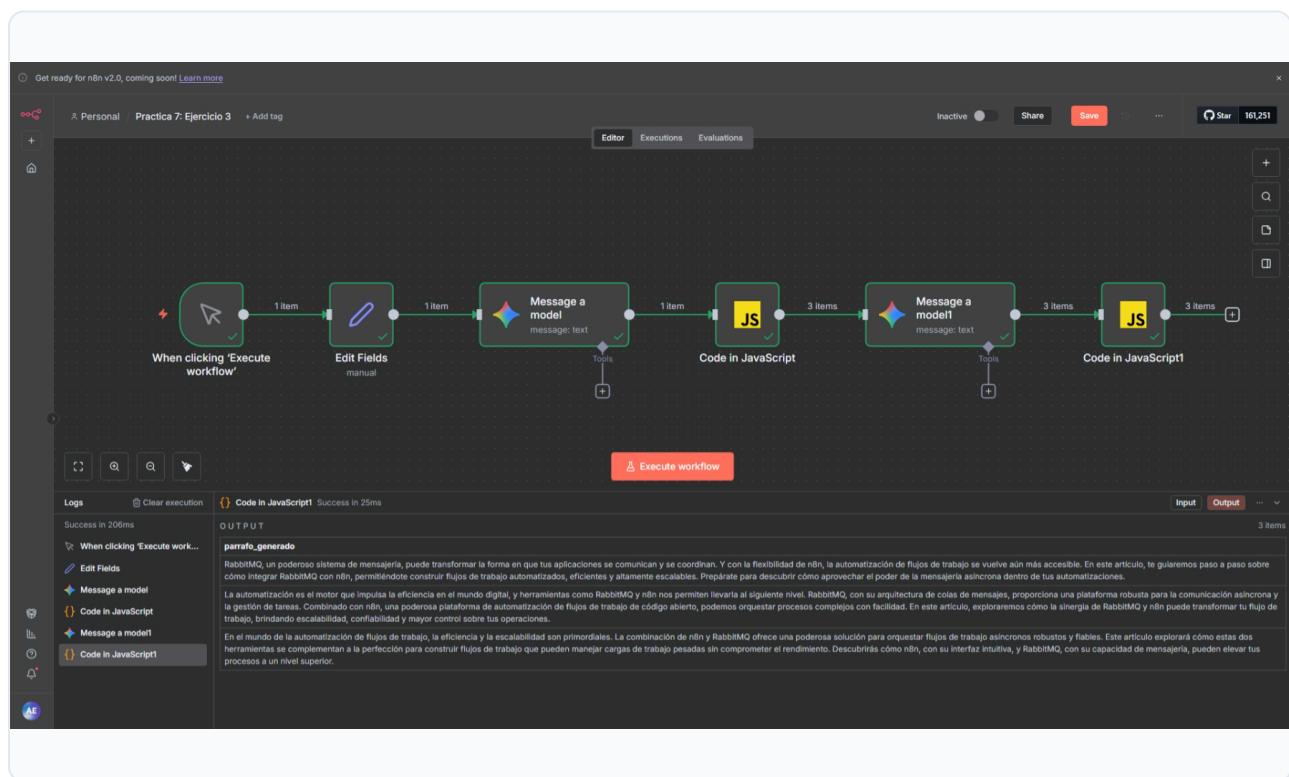
### 2. GENERACIÓN IDEAS + SPLIT (CODE)



### 3. PROMPT ITERATIVO (INTRO)



### 4. RESULTADO FINAL (3 INTROS GENERADAS)



# Conclusiones

Resumen de la Práctica 7

06

## ✓ Logros Alcanzados

- Integración exitosa de Modelos de Lenguaje (LLMs) en flujos de trabajo automatizados.
- Capacidad para procesar datos no estructurados (texto) y convertirlos en datos estructurados (JSON).
- Implementación de lógica de decisión basada en análisis semántico de IA.
- Creación de cadenas de razonamiento complejas (AI Chains) para generación de contenido.