

# DevOps y Versionado


Práctica 8: Gestión del ciclo de vida de workflows, control de versiones con Git y despliegue automatizado en n8n.

## TECNOLOGÍAS

 Git / GitHub

 n8n CLI / Docker

## REPOSITORIO

 [github.com/aek676/itsi-2026](https://github.com/aek676/itsi-2026)

# Ejercicio Guiado

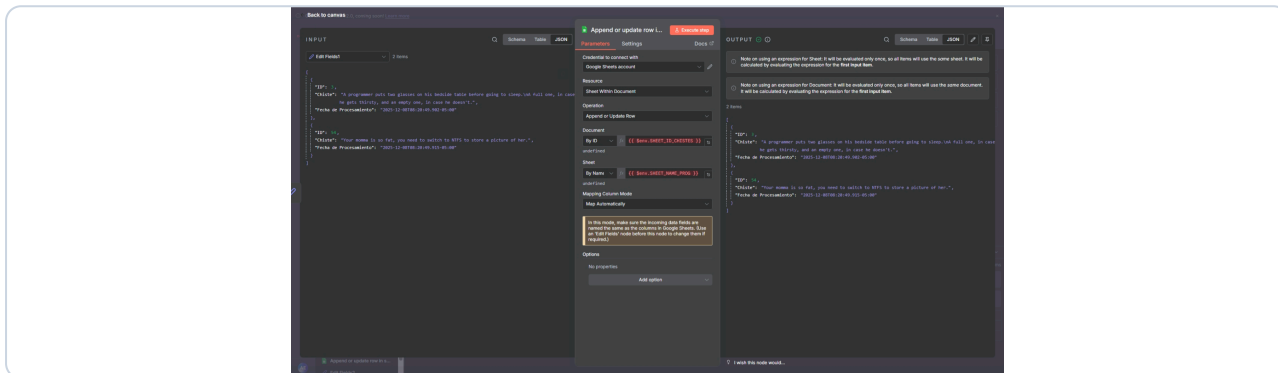
## Paso 1 y 2: Refactorización y Configuración

01

### DESACOPLAMIENTO CON VARIABLES DE ENTORNO

Se modificaron los nodos de Google Sheets para sustituir los valores fijos por variables `{{ $env.SHEET_NAME_... }}`. Estas variables se inyectaron mediante `docker-compose.yml` y un archivo `.env`.

### REFACTORIZACIÓN DEL NODO SHEETS



### CONFIGURACIÓN DOCKER COMPOSE (VARIABLES Y VOLÚMENES)

```
services:
  n8n:
    container_name: n8n
    image: docker.n8n.io/n8nio/n8n:latest
    ports: ["5678:5678"]
    env_file: [.env]
    environment:
      - N8N_RUNNERS_ENABLED=true
      - N8N_GIT_NODE_DISABLE_BARE_REPOS=true
      - N8N_ENFORCE_SETTINGS_FILE_PERMISSIONS=true
      - SHEET_NAME_PROG=Programming # Variable Inyectada
      - SHEET_NAME_MISC=Misc # Variable Inyectada
      - SHEET_NAME_DARK=Dark # Variable Inyectada
    volumes:
      - ./local-files:/files
      - n8n_data:/home/node/.n8n
      - ./workflows:/home/node/workflows # Bind Mount para GitOps
    depends_on: [mq, db]

# ... Resto de servicios (web, worker, notifier, etc)
```

# Ejercicio Guiado (Cont.)

01

## Paso 3 y 4: Importación y Despliegue

### ESTRATEGIA DE DESPLIEGUE

Para simular un despliegue en producción, se utilizó un **bind mount** de la carpeta local `./workflows` al contenedor. Posteriormente, se accedió a la terminal del contenedor `n8n` para ejecutar el comando de importación de la CLI.

### LOGS DE IMPORTACIÓN EXITOSA (N8N CLI)

```
~ $ n8n import:workflow --input=./workflows/workflow.json

Permissions 0644 for n8n settings file /home/node/.n8n/config are too wide.
This is ignored for now...

Importing 1 workflows...
Could not find workflow
Error: Could not find workflow
    at ActiveWorkflowManager.clearWebhooks...

Successfully imported 1 workflow.
```

### WORKFLOW IMPORTADO Y EJECUTADO CORRECTAMENTE

The screenshot shows the n8n workflow editor interface. The workflow is titled "Practica 3 - Ejercicio 1" and is in the "Editor" tab. The workflow consists of the following steps:

- When clicking 'Execute workflow' (Trigger)
- HTTP Request (GET: https://v2.jokeapi.dev/...)
- Split Out (10 items)
- Switch (mode: Rules)
- Edit Fields1 (raw, 2 items)
- Append or update row in sheet (appendOrUpdate: sheet, 2 items)
- Edit Fields2 (raw, 4 items)
- Append or update row in sheet1 (appendOrUpdate: sheet, 4 items)
- Append or update row in sheet2 (appendOrUpdate: sheet, 4 items)

The workflow is executed successfully, and the output is displayed in the bottom panel. The output shows a table with 2 columns: "Chiste" and "Fecha de Procesamiento".

ID	Chiste	Fecha de Procesamiento
3	A programmer puts two glasses on his bedside table before going to sleep. \n A full one, in case he gets thirsty, and an empty one, in case he doesn't.	2025-12-08T06:20:49.902-05:00
54	Your momma is so fat, you need to switch to NFTS to store a picture of her.	2025-12-08T06:20:49.915-05:00

# Ejercicio 1: Refactorización

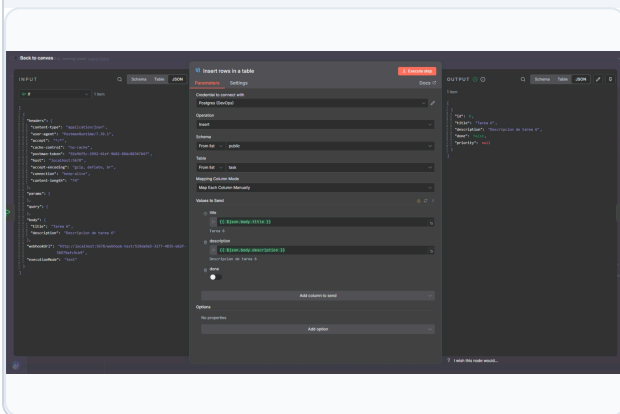
## Desacoplamiento Completo de Credenciales

02

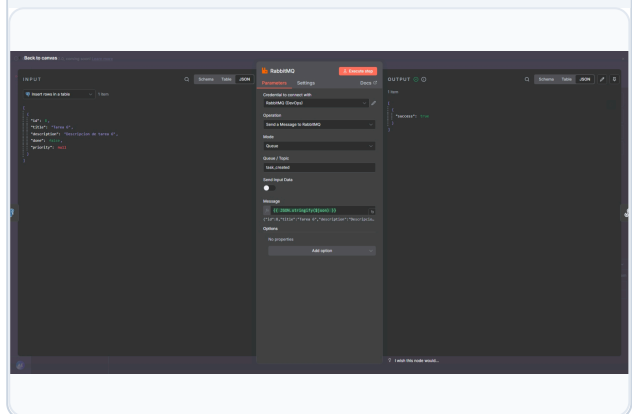
### IMPLEMENTACIÓN DE VARIABLES (.ENV)

Se creó un archivo .env para almacenar las credenciales de PostgreSQL y RabbitMQ. En n8n, se configuraron nuevas credenciales utilizando expresiones {{ \$env.VAR }} en lugar de valores fijos.

#### NODO POSTGRESQL (VARIABLE)



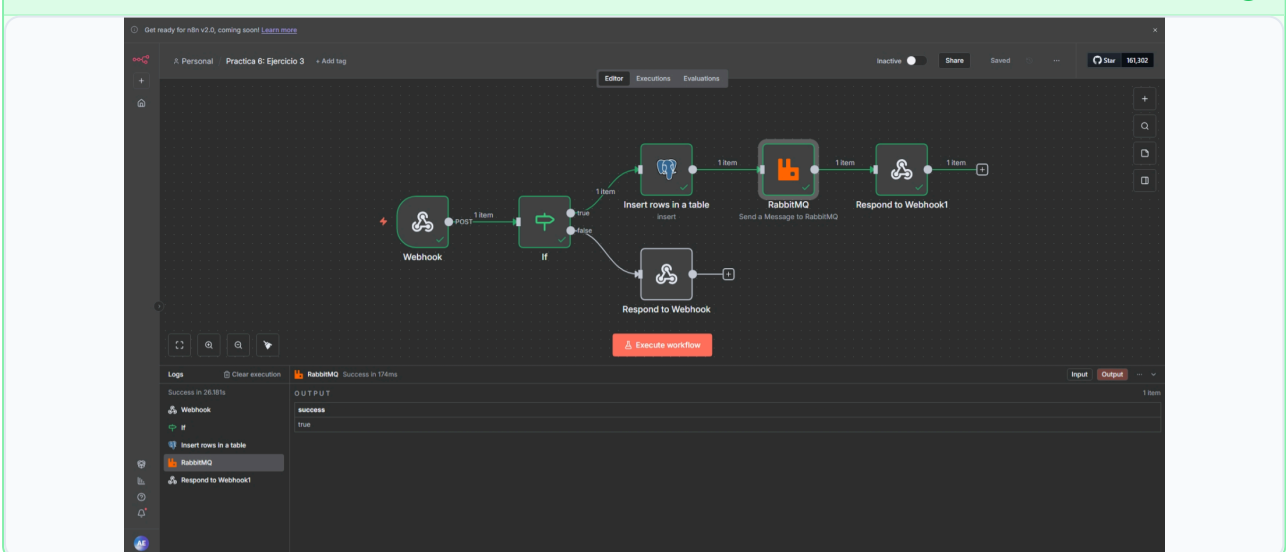
#### NODO RABBITMQ (VARIABLE)



### ARCHIVO .ENV (CONFIGURACIÓN)

```
DB_HOST=db
DB_NAME=taskdb
DB_USER=user
DB_PASS=password
RABBITMQ_HOST=mq
RABBITMQ_USER=guest
RABBITMQ_PASS=guest
```

### EJECUCIÓN EXITOSA CON VARIABLES



# Ejercicio 3: Pipeline CI/CD

Automatización con GitHub Actions

03

## CONCEPTO GITOPS

Implementación de un flujo de CI/CD que detecta cambios en la rama main y despliega automáticamente el workflow actualizado en el entorno de producción usando n8n-cli y GitHub Actions.

## DEFINICIÓN DEL PIPELINE (.GITHUB/WORKFLOWS/DEPLOY.YML)

```
name: Deploy n8n Workflow

on:
  push:
    branches: [ "main" ]
    paths: [ "workflows/workflow.json" ]

jobs:
  deploy:
    runs-on: ubuntu-latest
    steps:
      - name: Checkout repository
        uses: actions/checkout@v3

      - name: Setup Node.js
        uses: actions/setup-node@v3
        with:
          node-version: '24'

      - name: Install n8n-cli
        run: npm install -g n8n-cli

      - name: Deploy to Production
        env:
          N8N_HOST: ${ secrets.N8N_PROD_HOST }
          N8N_API_KEY: ${ secrets.N8N_PROD_API_KEY }
        run: |
          n8n-cli workflows import workflows/workflow.json \
            --name "Practica8 - Trabajo Guiado" \
            --skip-validation
```

## EJECUCIÓN DEL ACTION (SIMULACIÓN)

Error Esperado (Sin entorno Prod)

The screenshot shows the GitHub Actions interface for a workflow named 'Deploy n8n Workflow'. The workflow is triggered by a push to the 'main' branch. The 'deploy' job is running on 'ubuntu-latest'. The steps are: 'Checkout repository', 'Setup Node.js', 'Install n8n-cli', and 'Deploy to Production'. The 'Deploy to Production' step has failed with an error: 'Unable to connect to n8n. Check N8N\_HOST and ensure n8n is running. [CONNECTION\_ERROR]'. The error message also includes a list of possible causes: 'n8n instance is not running', 'Network connectivity issues', 'Firewall blocking the connection', and 'N8N\_HOST URL is incorrect'. The workflow run is labeled 'Practica8 - Trabajo Guiado'.

### Competencias Adquiridas

En esta práctica se han abordado los fundamentos de DevOps aplicados a la orquestación de flujos de trabajo con n8n. Se han logrado los siguientes objetivos:

- **Gestión de Configuración:** Se ha aprendido a desacoplar la lógica de los datos sensibles mediante el uso de variables de entorno, inyectándolas a través de Docker Compose.
- **Control de Versiones:** Se ha implementado un sistema de versionado utilizando Git, tratando los flujos de trabajo como código (JSON).
- **Despliegue Automatizado:** Se ha utilizado la herramienta de línea de comandos (n8n CLI) para importar y desplegar flujos de trabajo en un entorno simulado de producción.
- **Automatización CI/CD:** Se ha diseñado conceptualmente un pipeline de integración y despliegue continuo con GitHub Actions.

Estas prácticas son esenciales para mantener entornos de producción robustos, escalables y seguros.