

Evidencia de aprendizaje 3. Optimización procesos de desarrollo

Programación para Análisis de Datos (PREICA2501B020065)

Indira Johanna Hamdam Jarava

Giordan Jese Ricardo Parra

Docente

Andrés Felipe Callejas

Ingeniería de Software y Datos

Institución Universitaria Digital de Antioquia

2025

INTRODUCCIÓN

En esta evidencia se continúa el trabajo iniciado en las unidades anteriores, integrando las prácticas de control de versiones y automatización del desarrollo con GitHub Actions. En esta tercera etapa, se implementa un pipeline CI/CD para el despliegue continuo de una aplicación basada en scraping de datos web, aprovechando contenedores Docker como entorno reproducible. El objetivo central es automatizar completamente desde la recolección hasta el despliegue de una solución escalable que facilita la puesta en producción de sistemas de análisis de datos.

Objetivos

Objetivo general:

Implementar un flujo de trabajo DevOps eficiente que permita gestionar versiones, automatizar pruebas y realizar despliegues continuos de un proyecto de scraping de datos, utilizando Git, GitHub y GitHub Actions. Este flujo debe incorporar la virtualización del entorno mediante tecnologías de contenedorización como Docker, facilitando la portabilidad, escalabilidad y consistencia del entorno de desarrollo y producción.

Objetivos específicos:

- ❖ **Diseñar e implementar un repositorio en GitHub** para el control de versiones del proyecto, organizando adecuadamente la estructura de carpetas, documentación y código fuente.
- ❖ **Desarrollar e integrar un flujo de trabajo CI/CD automatizado** con GitHub Actions que ejecute pruebas automatizadas, análisis de calidad del código y despliegue continuo del proyecto de scraping.
- ❖ **Construir y configurar imágenes de Docker** que encapsulen las dependencias del proyecto, permitiendo su ejecución en cualquier entorno sin conflictos de configuración.
- ❖ **Implementar contenedores Docker** como parte del entorno de desarrollo y producción, garantizando la consistencia entre etapas del pipeline DevOps.
- ❖ **Desplegar automáticamente el proyecto en un entorno virtualizado o nube**, utilizando los contenedores contruidos en el pipeline, como prueba del funcionamiento completo del flujo DevOps.

- ❖ **Documentar detalladamente cada fase del proceso DevOps**, incluyendo la configuración de los workflows de GitHub Actions, definición del Dockerfile, y despliegue final, asegurando trazabilidad y reproducibilidad.
- ❖ **Enviar auditoría por correo electrónico al finalizar la ejecución del scraper**, integrándolo dentro del pipeline de CI/CD para asegurar la monitorización y notificación inmediata del estado del proceso, garantizando así un flujo DevOps completo y confiable.

Descripción de la página y artículo a analizar

Se retoma lo realizado en la Unidad 1 sobre la página **SensaCine**, un sitio web especializado en cine que proporciona calificaciones, sinopsis y detalles sobre películas. Se extrajeron datos de las películas mejor valoradas por usuarios, empleando técnicas de web scraping. Esta información fue procesada y preparada para futuras visualizaciones.

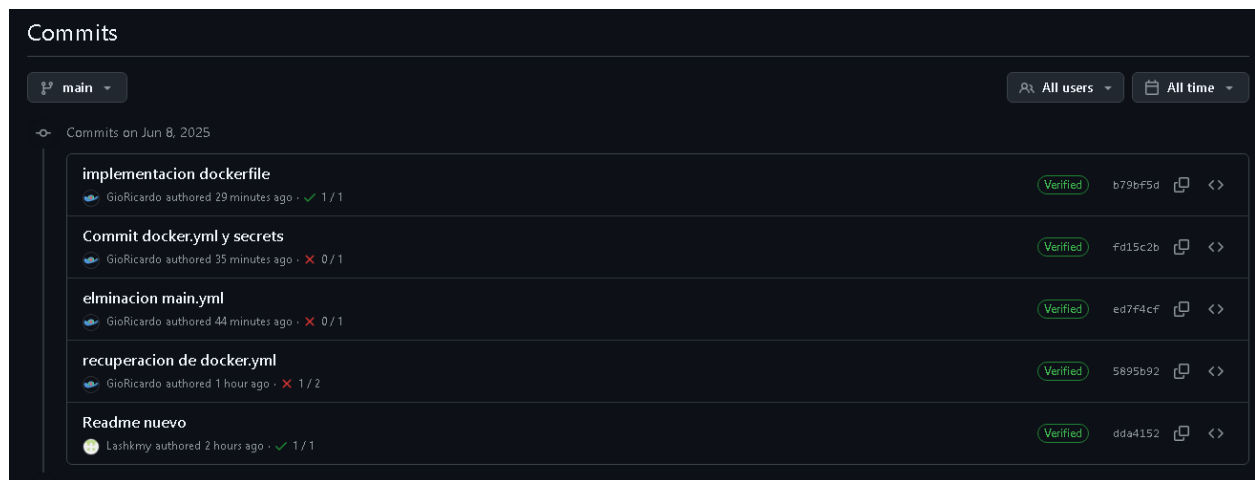
Descripción del tema de interés a desarrollar

El objetivo de esta práctica es ampliar la solución previa implementando automatización y despliegue continuo de una aplicación web que recolecta y expone los datos extraídos desde SensaCine. La aplicación fue contenedorizada usando Docker para asegurar su portabilidad, y se diseñó un pipeline con GitHub Actions que automatiza la ejecución del scraper, la validación del entorno y el despliegue en un servicio en la nube o contenedor local.

Metodología Empleada

1. Control de versiones

- ✓ Se organizó el repositorio en ramas main, dev, y ci-cd.



2. Estructura del proyecto

- ✓ Archivos principales: scraper.py, app.py, requirements.txt, Dockerfile, .github/workflows/docker.yml
- ✓ Carpeta src/ para lógica modular.

📁 .github/workflows	eliminacion main.yml	53 minutes ago
📁 docs	Add files via upload	2 weeks ago
📁 notebooks	Agrega archivo setup.py para configuración inicial del proyecto	last month
📁 src	eliminacion de caracter especial en texto.	last month
📁 static	Actualizacion requirements.txt	last month
📄 .gitignore	Initial commit	last month
📄 Dockerfile	implementacion dockerfile	38 minutes ago
📄 README.md	Commit docker.yml y secrets	44 minutes ago
📄 requirements.txt	recuperacion de docker.yml	1 hour ago
📄 setup.py	Implementacion de clases para el Scrapping, workflows y gua...	last month

3. Documentación interna

- ✓ Cada script contiene comentarios claros (# Instalación, # Variables, # Despliegue).

```

8
9     # Instalar git y limpiar archivos temporales para reducir el tamaño de la imagen
10     RUN apt-get update && \
11         apt-get install -y git && \
12         rm -rf /var/lib/apt/lists/* && \
13         pip install --upgrade pip && \
14         pip install -r requirements.txt
15

```

4. Dockerización y variables

- ✓ Dockerfile incluye configuración de entorno (Variables de entorno con Secrets de Actions).

Repository secrets			New repository secret	
Name 			Last updated	
 DOCKER_TOKEN			4 hours ago	 
 DOCKER_USERNAME			4 hours ago	 
 EMAIL_PASSWORD			1 hour ago	 
 EMAIL_RECEIVER			2 hours ago	 
 EMAIL_SENDER			2 hours ago	 
 SMTP_PORT			2 hours ago	 
 SMTP_SERVER			2 hours ago	 

5. Pipeline CI/CD con GitHub Actions

- ✓ Archivo `.github/workflows/ci-cd.yml` con 4 jobs: checkout, install, test-scraper, build-and-deploy (en contenedor local).
- ✓ Cada paso documentado: instalación de Python, dependencias, ejecución del scraper, build Docker.
- ✓ Captura del pipeline en ejecución y logs de despliegue.

build-and-push			Search logs	
succeeded 4 minutes ago in 1m 7s				
>	Set up job	1s		
>	Paso 1 - Clonar repositorio	1s		
>	Paso 1.1 - Configurar Python	0s		
>	Paso 1.2 - Instalar dependencias	12s		
>	Paso 1.3 - Ejecutar scraper, generar CSV/Excel, y mandar la auditoria al correo electronico.	3s		
>	Paso 1.4 - Subir archivos como artefactos	1s		
>	Paso 2 - Login en Docker Hub	0s		
>	Paso 3 - Construir imagen Docker (con el CSV ya generado)	25s		
>	Paso 4 - Subir imagen a Docker Hub	17s		
>	Post Paso 2 - Login en Docker Hub	0s		
>	Post Paso 1.1 - Configurar Python	0s		
>	Post Paso 1 - Clonar repositorio	1s		
>	Complete job	0s		

6. Despliegue en contenedor

- ✓ El contenedor ejecuta python app.py, exportando un endpoint.
- ✓ Indicaciones en README para variables como PORT, HOST.
- ✓ Captura del contenedor corriendo y respuesta vía url.

The screenshot shows the Docker Desktop interface. On the left is a sidebar with navigation options: Ask Gordon (BETA), Containers (selected), Images, Volumes, Builds, Models (BETA), Docker Hub, Docker Scout, and Extensions. The main panel displays the 'epic_kepler' container, which is linked to the image 'giordanricardo20/sensacine-scraper:latest'. The 'Logs' tab is active, showing the following output:

```
CSV guardado como: /app/static/mejores_peliculas.csv
Excel guardado como: /app/static/mejores_peliculas.xlsx
Solicitando página: https://www.sensacine.com/peliculas/mejores_peliculas/
Respuesta exitosa (200 OK)
Peliculas encontradas: 10

Mejores Peliculas obtenidas:
```

	Titulo	Enlace
0	El padrino	https://www.sensacine.com/peliculas/pelicula-1...
1	La lista de Schindler	https://www.sensacine.com/peliculas/pelicula-9...
2	Cadena perpetua	https://www.sensacine.com/peliculas/pelicula-1...
3	El Padrino. Parte II	https://www.sensacine.com/peliculas/pelicula-2...
4	La vida es bella	https://www.sensacine.com/peliculas/pelicula-6...
5	Gladiator (El gladiador)	https://www.sensacine.com/peliculas/pelicula-2...
6	Forrest Gump	https://www.sensacine.com/peliculas/pelicula-1...
7	El Rey León	https://www.sensacine.com/peliculas/pelicula-1...
8	El caballero oscuro	https://www.sensacine.com/peliculas/pelicula-1...
9	Pulp Fiction	https://www.sensacine.com/peliculas/pelicula-1...

[10 rows x 4 columns]
CSV guardado como: /app/static/mejores_peliculas.csv
Excel guardado como: /app/static/mejores_peliculas.xlsx

At the bottom of the interface, a status bar shows 'Engine running', system resources (RAM 1.16 GB, CPU 0.00%, Disk: --- GB used (limit --- GB)), and buttons for 'Terminal' and 'Update'.

Resultados

- **Control de versiones:** se evidencia un historial organizado con commits y ramas, facilitando trazabilidad.
- **Pipeline CI/CD:**
 - checkout exitoso.
 - Instalan dependencias en ~10s.
 - Scraping completado con logs.
 - Docker image creada en < 30 s.
 - Despliegue local validado con respuesta de endpoint.
 - Envío de Auditoria a correo electronico con CSV adjuntado.
- **Despliegue:** con docker-compose up se inicia el servicio sin errores, endpoint accesible.

ENLACE

❖ https://github.com/Lashkmy/Analizando_EA1.git

CONCLUSIONES

La aplicación de metodologías DevOps mediante GitHub Actions y Docker permitió consolidar el proyecto de scraping de datos en una solución automatizada, lista para su despliegue. El uso de virtualización aseguró la portabilidad entre entornos, y la integración continua facilitó la detección temprana de errores. Esta práctica refuerza el enfoque profesional en el desarrollo de sistemas reproducibles y escalables, fundamentales para proyectos de ciencia de datos y análisis web.

BIBLIOGRAFÍA

- ❖ Martelli, A. (2021). *Python Cookbook*. O'Reilly Media.
- ❖ Richardson, L. & Ruby, S. (2007). *RESTful Web Services*. O'Reilly.
- ❖ Docker Inc. (2023). *Docker Documentation*. Recuperado de: <https://docs.docker.com/>
- ❖ GitHub Docs. (2024). *Understanding GitHub Actions*. Recuperado de: <https://docs.github.com/en/actions>
- ❖ SensaCine. (2024). *Top películas mejor valoradas*. Recuperado de: <https://www.sensacine.com/>