Ejercicios - Optimización de imágenes

Arturo Silvelo

Try New Roads

Introducción: app.js

La aplicación `app.js` incluida en esta carpeta es una API HTTP sencilla desarrollada en Node.js. Su objetivo es servir como base para practicar la creación de imágenes Docker avanzadas y la gestión de variables de entorno y secretos.

¿Cómo funciona?

- Al arrancar, expone un endpoint HTTP (por defecto en el puerto 3000) que responde con un JSON mostrando:
  - El entorno de ejecución (`NODE_ENV`)
  - El puerto (`PORT`)
  - El estado del secreto (`DB_PASSWORD`), que nunca se muestra en claro

Ejercicio 1: Multi-stage build básico

Parte de la aplicación `app.js` incluida en esta carpeta. Crea una imagen multi-stage para esta app Node.js:

- El primer stage debe instalar dependencias y preparar la app.
- El segundo stage solo copiará `app.js` y `package.json` necesarios para ejecutar la app.
- El resultado debe ser una imagen pequeña y segura que ejecute correctamente `app.js`.

Ejercicio 2: Optimización de capas

Mejora el Dockerfile multi-stage anterior para optimizar el uso de capas y aprovechar la cache de Docker.

Ejercicio 3: Uso de ARG y ENV

Modifica el Dockerfile para que la app (`app.js`) pueda recibir argumentos de build (`ARG`) y variables de entorno (`ENV`) que permitan personalizar su comportamiento:

- Permite cambiar el puerto (`PORT`) y el entorno (`NODE_ENV`) en tiempo de build y ejecución.
- Comprueba que los cambios se reflejan en la respuesta de la app (consulta el endpoint y revisa el JSON devuelto).

Ejercicio 4: Gestión de secretos

Implementa una estrategia segura para manejar un secreto (por ejemplo, una contraseña de base de datos) en el contenedor usando la variable de entorno `DB_PASSWORD` que utiliza `app.js`.

- No incluyas el secreto real en el `Dockerfile` ni en la imagen.
- Comprueba que la app nunca expone el valor real del secreto en la respuesta.