**Guía para Crear y Empaquetar una API en una Imagen Docker**

**1. Herramientas necesarias:**

1. **Docker** - Para construir y ejecutar contenedores.
2. **Editor de Código.**
3. **PHP**

**2. Creación de la API:**

Primero, se debe crear una carpeta en su computadora y se tiene que agregar un archivo index.php.

**Estructura de Archivos:**

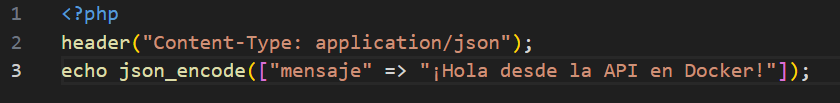
mi\_api/

│-- index.php

│-- Dockerfile

│-- .dockerignore

**Código en index.php**



(API básica que devuelve un mensaje en formato JSON)

**3. Creación del Dockerfile**

Cree un archivo llamado Dockerfile en la misma carpeta y agregue lo siguiente:

Pantalla de computadora con letras

El contenido generado por IA puede ser incorrecto.

**4. Creación del Archivo “.dockerignore”**

Este archivo evita que archivos innecesarios se copien al contenedor.

**Contenido de “.dockerignore”**

.git

node\_modules

vendor

Dockerfile

.dockerignore

**5. Construcción y Ejecución de la Imagen Docker**

Ahora no tenemos que dirigir a la carpeta donde está la API; abrir una terminal y ejecute los siguientes comandos:

**Construcción de la Imagen:**

docker build -t prueba-api-php

Esto crea una imagen llamada mi-api-php.

**Ejecutar el Contenedor:**

docker run -d -p 8080:8080 --name api prueba-api-php

* -d: Ejecuta en segundo plano.
* -p 8080:8080: Mapea el puerto 8080 del contenedor a la PC.
* --name api: Asigna el nombre api al contenedor.

**Para verificar que el contenedor esta corriendo ejecutamos:**

docker ps

**Para probar nuestra API debemos hacer lo siguiente:**

Abra un navegador y accedemos a nuestro “localhost”

O usamos el **url** en la terminal:

curl http://localhost:8080

Y nos va a devolver:

{"mensaje":"¡Hola desde la API en Docker!"}

Si se trabaja con más miembros es recomendable subirlo a Docker Hub. Para un trabajo más profesional y mejor gestión por el control de versiones

**Guía para Empaquetar una Aplicación en Docker**

Para empaquetar una aplicación en Docker y hacerla ejecutable desde un contenedor, sigue esta guía paso a paso. En este caso, vamos a crear una imagen Docker que sirva una aplicación web desde la URL proporcionada: <https://www.abank.com.sv/> .

Paso 1: Instalar Docker

Asegúrate de tener Docker instalado en tu máquina. Puedes descargarlo desde Docker Hub.

Paso 2: Crear un Dockerfile

El Dockerfile es un archivo de texto que contiene las instrucciones para construir la imagen Docker. Crea un nuevo directorio para tu proyecto y dentro de él, crea un archivo llamado Dockerfile.

bash

mkdir mi-aplicacion-docker

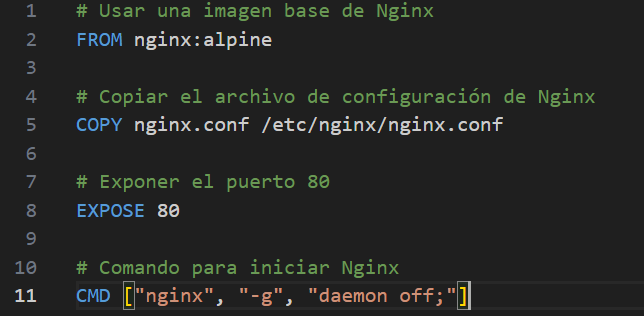
cd mi-aplicacion-docker

touch Dockerfile

Paso 3: Escribir el Dockerfile

Abre el Dockerfile en un editor de texto y agrega el siguiente contenido:

dockerfile



Paso 4: Crear el archivo de configuración de Nginx

Crea un archivo llamado nginx.conf en el mismo directorio y agrega el siguiente contenido:

nginx

server {

listen 80;

location / {

proxy\_pass https://www.abank.com.sv;

proxy\_set\_header Host $host;

proxy\_set\_header X-Real-IP $remote\_addr;

proxy\_set\_header X-Forwarded-For $proxy\_add\_x\_forwarded\_for;

proxy\_set\_header X-Forwarded-Proto $scheme;

}

}

Paso 5: Construir la imagen Docker

Desde el directorio donde se encuentra tu Dockerfile, ejecuta el siguiente comando para construir la imagen:

bash

docker build -t mi-aplicacion-nginx

Paso 6: Ejecutar el contenedor

Una vez que la imagen se haya construido correctamente, puedes ejecutar un contenedor basado en esa imagen con el siguiente comando:

bash

docker run -d -p 8080:80 mi-aplicacion-nginx

Este comando ejecuta el contenedor en segundo plano (-d) y mapea el puerto 80 del contenedor al puerto 8080 de tu máquina local.

Paso 7: Acceder a la aplicación

Abre tu navegador web y dirígete a <http://localhost:8080> . Deberías ver la aplicación que se está sirviendo desde <https://www.abank.com.sv> .

Paso 8: Detener y eliminar el contenedor

Si deseas detener el contenedor, primero necesitas encontrar el ID del contenedor ejecutando:

bash

docker ps

Luego, usa el siguiente comando para detenerlo:

bash

docker stop <container\_id>

Y para eliminarlo:

bash

docker rm <container\_id>

Consideraciones Finales:

Seguridad: Asegúrate de que el proxy inverso esté configurado correctamente para manejar la seguridad y las cabeceras adecuadas.

Escalabilidad: Puedes escalar tu aplicación utilizando Docker Compose si necesitas manejar múltiples contenedores o servicios.

Pruebas: Realiza pruebas para asegurarte de que la aplicación funciona como se espera en el contenedor.

Siguiendo estos pasos, habrás creado una imagen Docker que contiene una aplicación web que se puede desplegar en cualquier entorno compatible con Docker.