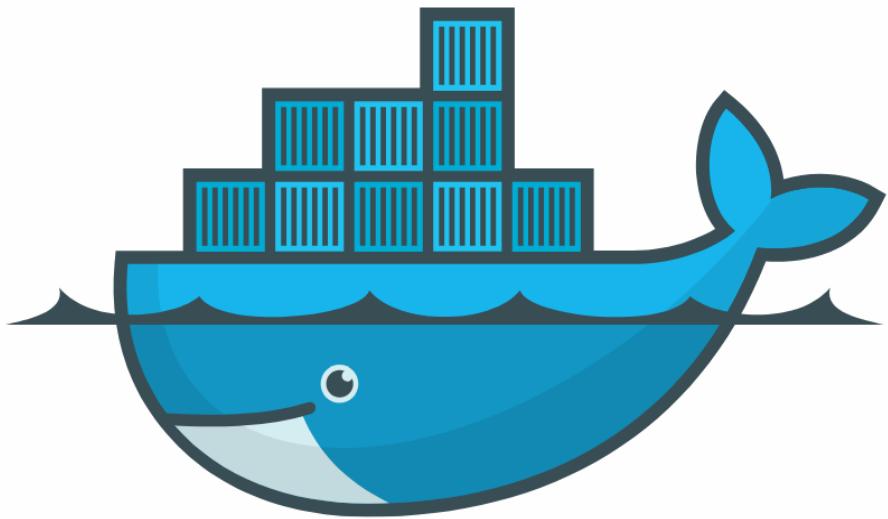


CREANDO DOCKERFILE



Ivana Sánchez Pérez

INTRODUCCIÓN

Docker es una plataforma de código abierto que permite desarrollar, enviar y ejecutar aplicaciones en contenedores. Un contenedor es una unidad estándar de software que empaqueta el código y todas sus dependencias, de modo que la aplicación se ejecute de manera rápida y confiable en diferentes entornos informáticos.

¿Por qué usar Docker?

1. **Consistencia:** Los contenedores garantizan que las aplicaciones se ejecuten de la misma manera en cualquier entorno, ya sea en desarrollo, pruebas o producción.
2. **Aislamiento:** Cada contenedor es independiente, lo que significa que las aplicaciones pueden ejecutarse sin interferir entre sí, incluso si usan versiones diferentes de las mismas bibliotecas.
3. **Escalabilidad:** Docker simplifica el escalado de aplicaciones al permitir el despliegue rápido de contenedores en múltiples instancias.
4. **Eficiencia:** Los contenedores comparten el mismo núcleo del sistema operativo, lo que los hace más ligeros en comparación con las máquinas virtuales, que requieren un sistema operativo completo.

Componentes Clave

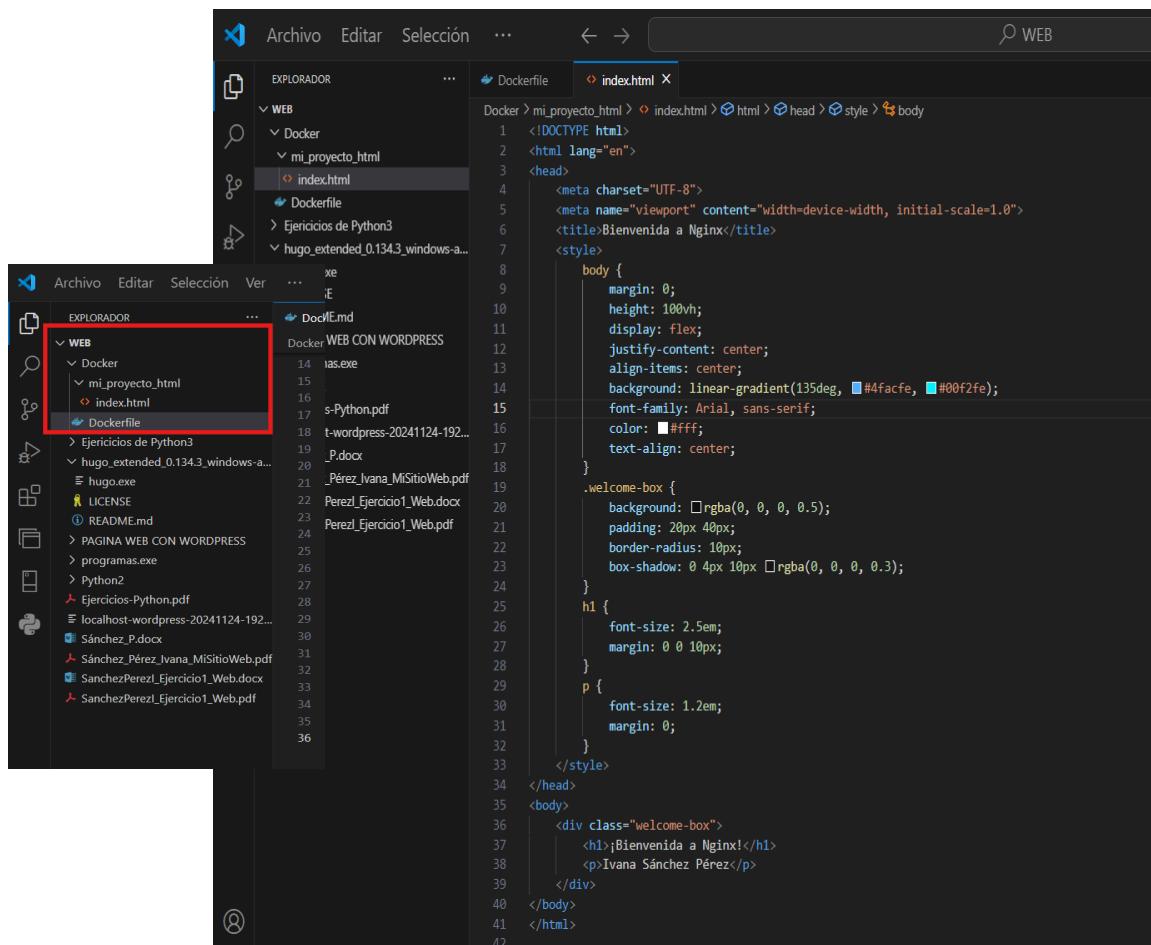
- **Docker Engine:** El núcleo de Docker que permite la creación y gestión de contenedores.
- **Docker Hub:** Un registro en la nube donde se pueden almacenar y compartir imágenes de contenedores.
- **Docker Compose:** Una herramienta para definir y ejecutar aplicaciones multicontenedor mediante un archivo de configuración YAML.

Conclusión

Docker ha revolucionado la forma en que se desarrollan y despliegan aplicaciones, proporcionando una solución eficiente, portátil y escalable para la gestión de software. Con su creciente popularidad, aprender a usar Docker se ha vuelto una habilidad esencial para desarrolladores y administradores de sistemas.

TAREA: Creando Dockerfile completo

Lo primero que he hecho ha sido crear la estructura de carpetas y archivos, que he necesitado para realizar la tarea. He comenzado por la carpeta principal a la que he llamado Docker, y dentro de ella el archivo Dockerfile y la carpeta mi_proyecto_html; en esta última es dónde he ubicado mi archivo index.html, donde he creado un html para el inicio de la página de Nginx.



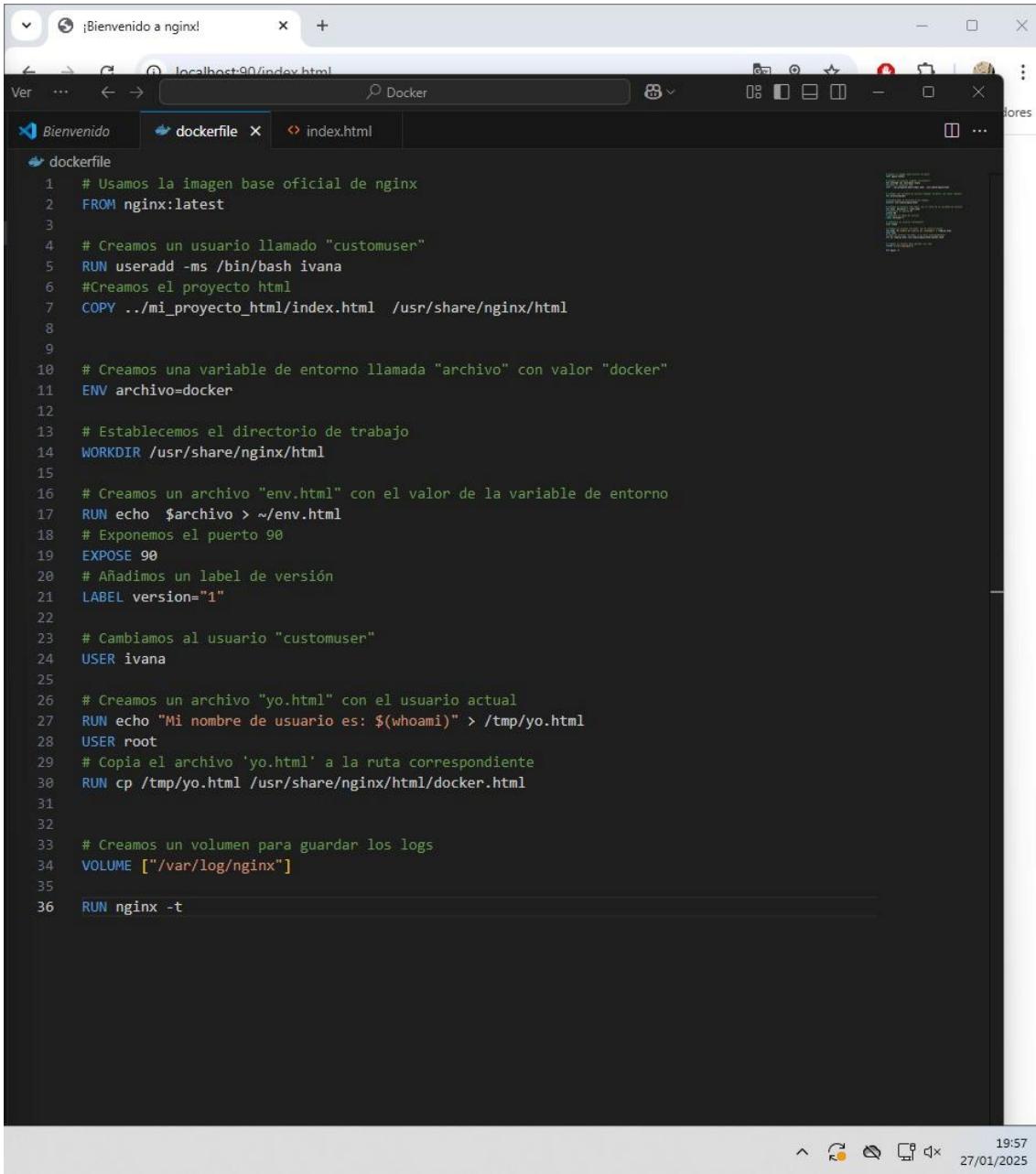
The screenshot shows a file explorer interface with two panes. The left pane, titled 'EXPLORADOR', displays a directory structure under 'WEB'. It includes a folder 'Docker' containing 'Dockerfile', 'index.html', and 'Dockerfile'. Below 'Docker' are 'Ejercicios de Python3', 'hugo_extended_0.134.3_windows-a...', 'hugo.exe', 'LICENSE', 'README.md', 'PAGINA WEB CON WORDPRESS', 'programas.exe', 'Python2', 'Ejercicios-Python.pdf', 'localhost-wordpress-20241124-192...', 'Sánchez_P.docx', 'Sánchez_Pérez_Ivana_MiSitioWeb.pdf', 'SanchezPerez_Ejercicio1.Web.docx', and 'SanchezPerez_Ejercicio1_Web.pdf'. The right pane shows the contents of 'index.html' and 'Dockerfile'. The 'index.html' content is as follows:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Bienvenida a Nginx!</title>
    <style>
        body {
            margin: 0;
            height: 100vh;
            display: flex;
            justify-content: center;
            align-items: center;
            background: linear-gradient(135deg, #4facfe, #00f2fe);
            font-family: Arial, sans-serif;
            color: #fff;
            text-align: center;
        }
        .welcome-box {
            background: rgba(0, 0, 0, 0.5);
            padding: 20px 40px;
            border-radius: 10px;
            box-shadow: 0 4px 10px rgba(0, 0, 0, 0.3);
        }
        h1 {
            font-size: 2.5em;
            margin: 0 0 10px;
        }
        p {
            font-size: 1.2em;
            margin: 0;
        }
    </style>
</head>
<body>
    <div class="welcome-box">
        <h1>Bienvenida a Nginx!</h1>
        <p>Ivana Sánchez Pérez</p>
    </div>
</body>
</html>
```

The 'Dockerfile' content is as follows:

```
Dockerfile
FROM nginx:latest
COPY index.html /usr/share/nginx/html
```

Y ahora procedo a configurar el archivo Dockerfile tal y como se muestra en la imagen.



The screenshot shows a browser window with the title '¡Bienvenido a nginx!' and the URL 'localhost:90/index.html'. The browser's address bar also displays 'Docker'. The main content area contains a code editor with a dark theme, showing a Dockerfile. The Dockerfile content is as follows:

```
1 # Usamos la imagen base oficial de nginx
2 FROM nginx:latest
3
4 # Creamos un usuario llamado "customuser"
5 RUN useradd -ms /bin/bash ivana
6 #Creamos el proyecto html
7 COPY ../mi_proyecto_html/index.html /usr/share/nginx/html
8
9
10 # Creamos una variable de entorno llamada "archivo" con valor "docker"
11 ENV archivo=docker
12
13 # Establecemos el directorio de trabajo
14 WORKDIR /usr/share/nginx/html
15
16 # Creamos un archivo "env.html" con el valor de la variable de entorno
17 RUN echo $archivo > ~/env.html
18 # Exponemos el puerto 90
19 EXPOSE 90
20 # Añadimos un label de versión
21 LABEL version="1"
22
23 # Cambiamos al usuario "customuser"
24 USER ivana
25
26 # Creamos un archivo "yo.html" con el usuario actual
27 RUN echo "Mi nombre de usuario es: $(whoami)" > /tmp/yo.html
28 USER root
29 # Copia el archivo 'yo.html' a la ruta correspondiente
30 RUN cp /tmp/yo.html /usr/share/nginx/html/docker.html
31
32
33 # Creamos un volumen para guardar los logs
34 VOLUME ["/var/log/nginx"]
35
36 RUN nginx -t
```

EXPLICACIÓN DE LAS INSTRUCCIONES

1. **FROM nginx:latest**

- Especifica la imagen base que se utilizará para construir la imagen Docker. En este caso, se usa la imagen oficial de Nginx en su versión más reciente.

2. **RUN useradd -ms /bin/bash ivana**
 - Crea un usuario llamado "ivana" con un shell Bash como predeterminado.
3. **COPY ./mi_proyecto_html/index.html /usr/share/nginx/html**
 - Copia un archivo HTML desde el sistema anfitrión a la carpeta de Nginx donde se almacenan los archivos estáticos.
4. **ENV archivo docker**
 - Define una variable de entorno llamada "archivo" con el valor "docker".
5. **WORKDIR /usr/share/nginx/html**
 - Cambia el directorio de trabajo dentro de la imagen a "/usr/share/nginx/html".
6. **RUN echo \$archivo > ./env.html**
 - Crea un archivo llamado "env.html" en el directorio actual y escribe el valor de la variable de entorno "archivo" en este archivo.
7. **EXPOSE 90**
 - Expone el puerto 90 para permitir el acceso al servidor Nginx desde el exterior del contenedor.
8. **LABEL version="1"**
 - Añade metadatos a la imagen, indicando que esta tiene la versión "1".
9. **USER ivana**
 - Cambia el usuario que ejecutará las siguientes instrucciones al usuario "ivana" creado previamente.
10. **RUN echo "Mi nombre de usuario es: \$(whoami)" > /tmp/yo.html**
 - Crea un archivo temporal "yo.html" que contiene el nombre del usuario actual ("ivana").
11. **USER root**
 - Cambia de nuevo al usuario "root" para realizar operaciones que requieren permisos elevados.
12. **RUN cp /tmp/yo.html /usr/share/nginx/html/docker.html**

- Copia el archivo "yo.html" a la carpeta pública de Nginx y lo renombra como "docker.html".

13. VOLUME ["/var/log/nginx"]

- Crea un volumen para almacenar los registros de Nginx fuera del contenedor.

14. RUN nginx -t

- Verifica la configuración de Nginx para asegurarse de que no haya errores.

Demostración del funcionamiento

1. Construcción de la imagen Docker

- Abrimos la PowerShell como administrador
- Hacemos un cd para ubicarnos en la carpeta principal “Docker”
- Ejecutamos el comando docker build -t mi_contenedor_nginx .

```

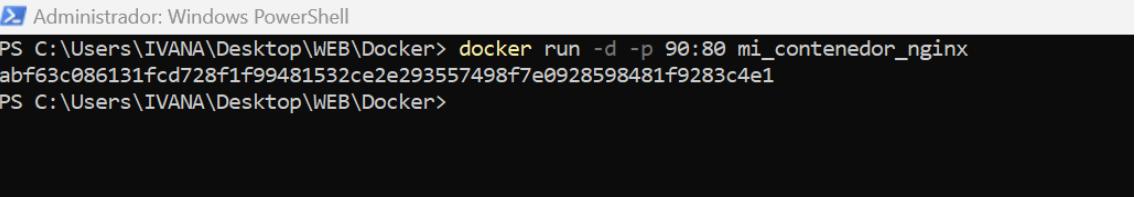
PS C:\Users\2asirn\Desktop\WEB\Docker> docker build -t mi_contenedor_nginx .
[+] Building 2.3s (11/11) FINISHED
=> [internal] load build definition from dockerfile
=> => transferring dockerfile: 1.16kB
=> [internal] load metadata for docker.io/library/nginx:latest
=> [internal] load .dockerignore
=> => transferring context: 2B
=> [1/7] FROM docker.io/library/nginx:latest@sha256:0a399eb16751829e1af26fea27b20c3ec28d7ab1fb72182879dcae1cc
=> => resolve docker.io/library/nginx:latest@sha256:0a399eb16751829e1af26fea27b20c3ec28d7ab1fb72182879dcae1cc
=> CACHED [2/7] RUN useradd -ms /bin/bash usuario1
=> [3/7] RUN mv /usr/share/nginx/html/index.html /usr/share/nginx/html/index.html.old && echo "<h1>Ivana </h1>">> index.html
=> [4/7] WORKDIR /usr/share/nginx/html
=> [5/7] RUN echo "Archivo: docker" > ~/env.html
=> [6/7] RUN echo "Mi nombre de usuario es: $(whoami)" > /tmp/yo.html
=> [7/7] RUN cp /tmp/yo.html /usr/share/nginx/html/docker.html
=> exporting to image
=> => exporting layers
=> => exporting manifest sha256:630a94f66deb9c204d68f1ad660726c2365f6a8c3b2e70fc283ebe7cef7a6dc9
=> => exporting config sha256:cfdff1bd3d45c1bd34a65aad785a6f9db5fd869bf3a94a9b5c2185872fdec7c8
=> => exporting attestation manifest sha256:697421572b693089c7dc7942370efdc8136ef3f175aa27324f088842880c84c8
=> => exporting manifest list sha256:a3055dda61a1e92c7b036b08e20a736ed07749c83534c6cf14abcc86bb651f8
=> => naming to docker.io/library/mi_contenedor_nginx:latest
=> => unpacking to docker.io/library/mi_contenedor_nginx:latest

View build details: docker-desktop://dashboard/build/desktop-linux/desktop-linux/uuuh6b2eq0el4kvvp1qxy5j5m
PS C:\Users\2asirn\Desktop\WEB\Docker>

```

2. Ejecución del contenedor

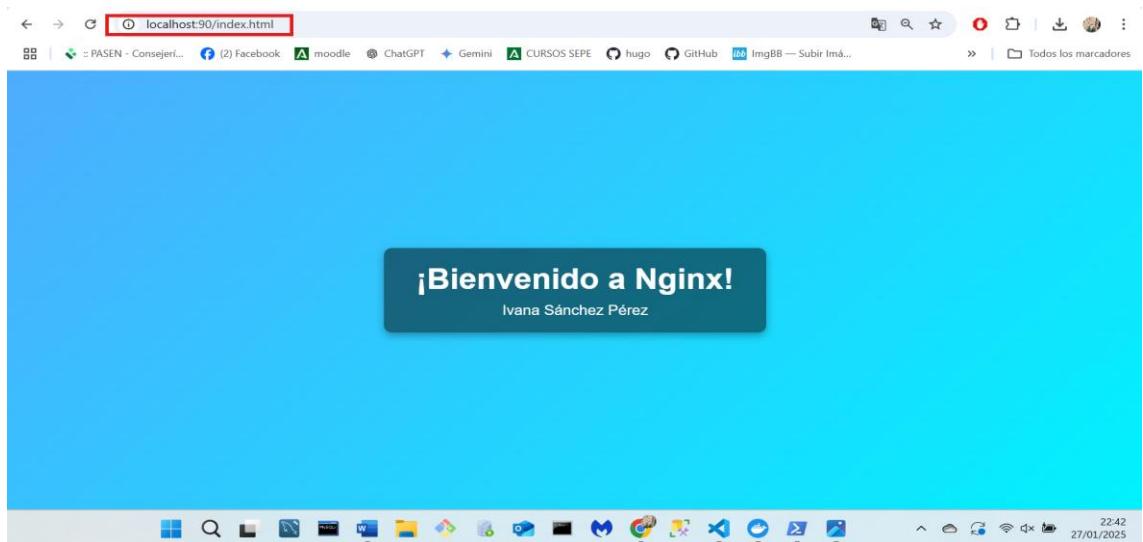
- Una vez construida la imagen, inicia un contenedor basado en ella:
- docker run -d -p 90:80 mi_contenedor_nginx
- Esto iniciará el servidor Nginx y lo expondrá en el puerto 90.



```
Administrator: Windows PowerShell
PS C:\Users\IVANA\Desktop\WEB\Docker> docker run -d -p 90:80 mi_contenedor_nginx
abf63c086131fc728f1f99481532ce2e293557498f7e0928598481f9283c4e1
PS C:\Users\IVANA\Desktop\WEB\Docker>
```

3. Verificación de los archivos creados

- Abrimos un navegador web y accedemos a las siguientes URLs:
 - <http://localhost:90/index.html>: Debería mostrar el contenido del archivo "index.html" copiado al contenedor.



- <http://localhost:90/docker.html>: Mostrará el texto "Mi nombre de usuario es: ivana".

