

## Laboratorio 8: Docker Apache

Ejercicio Práctico: Implementación de contenedores Docker

### Objetivo del Ejercicio:

Implementar un sistema de contenedores para despliegue de servicios.

### Pasos del Ejercicio:

#### Configuración del Entorno:

Descargue e instale Docker Desktop.

##### Paso 01: Instalar .

- <https://www.docker.com/products/docker-desktop/>
- Luego de instalar Docker instala una imagen por defecto

Desde powershell:

>Docker images

```
PS C:\Users\IngAR> docker images
REPOSITORY          TAG         IMAGE ID      CREATED        SIZE
docker/welcome-to-docker latest      c1f619b6477e 5 months ago  18.6MB
```

Para ver que imágenes están activas

>Docker ps

```
PS C:\Users\IngAR> docker ps
CONTAINER ID   IMAGE                                COMMAND                  CREATED
5066dbef3d57   docker/welcome-to-docker:latest    "/docker-entrypoint..." 32 minutes ago
```

##### Paso 02: Instalación de Apache HTTP Server.

servidor web de código abierto desarrollado y mantenido por la Apache Software Foundation. Es uno de los servidores web más populares y ampliamente utilizados en Internet. Apache HTTP Server es conocido por su flexibilidad, potencia y extensibilidad, lo que lo convierte en una opción favorable tanto para pequeños sitios web personales como para grandes portales empresariales.

- Desde el powershell  
>docker run -dit -name mayapache -p8080:80 httpd

##### docker run

docker: es el comando básico para iniciar la interfaz de línea de comandos de Docker.

run: le dice a Docker que cree e inicie un nuevo contenedor basado en la imagen que se especifica después de este comando.

##### -dit

Este es un conjunto de tres banderas que configura cómo se ejecutará el contenedor:

-d: "detached mode" (modo desapegado). Esto significa que el contenedor se ejecuta en el fondo y no bloquea la terminal desde la que se lanzó el comando.

-i: "interactive" (interactivo). Esto mantiene abierta la entrada estándar (STDIN) del contenedor incluso si no está adjunto a la terminal (útil para la interacción).

-t: "tty" (terminal virtual). Esto asigna una terminal virtual al contenedor, lo que es útil para hacer que el contenedor se comporte como una terminal completa.

##### --name myapache

--name: Especifica un nombre personalizado para el contenedor, que en este caso es myapache.

Esto es útil para referirse al contenedor en comandos futuros en lugar de usar el ID del contenedor generado automáticamente.

##### -p 8080:80

-p: Esta opción publica un puerto del contenedor a un puerto del host. Aquí, el puerto 80 del contenedor (puerto por defecto para un servidor web Apache) se está mapeando al puerto 8080 del

host. Esto significa que puedes acceder al servidor Apache en este contenedor visitando localhost:8080 en tu navegador mientras se ejecuta en la misma máquina host.

### httpd

**httpd:** Es el nombre de la imagen de Docker desde la cual se crea el contenedor. httpd es la imagen oficial de Docker para el servidor Apache HTTP. Está preconfigurada para funcionar como un servidor web

```

NAMES
5066dbef3d57 docker/welcome-to-docker:latest "/docker-entrypoint..." 32 min
>80/tcp welcome-to-docker
PS C:\Users\IngAR> docker run -dit --name myapache -p8080:80 httpd
Unable to find image 'httpd:latest' locally
latest: Pulling from library/httpd
13808c22b207: Pull complete
6e9a8835eae4: Pull complete
4f4fb700ef54: Pull complete
b927d001db70: Pull complete
559cc51378ed: Pull complete
d2b091e65160: Pull complete
Digest: sha256:10758fe1fe13980e0d7bbdf8f0bbb40cf04e7c996248aac4ea390670b2420bd1
Status: Downloaded newer image for httpd:latest
48a01441050dae7dbc771202f7914a63cee9b941cb25b1dc5fefcfed78f02bcb
PS C:\Users\IngAR>

```







>Docker images

```

PS C:\Users\IngAR> docker images
REPOSITORY          TAG          IMAGE ID          CREATED          SIZE
httpd                latest       fa0099f1c09d     6 days ago      148MB
docker/welcome-to-docker latest      c1f619b6477e     5 months ago    18.6MB
PS C:\Users\IngAR>

```

En Docker desktop se puede ver que el servidor esta instalado y activo:

<input type="checkbox"/>	Name	Image	Status	CPU (%)	Port(s)	Last started	Act
<input type="checkbox"/>	 <a href="#">welcome-to-docker</a> 5066dbef3d57	<a href="#">docker/welcome-to-docker</a>	Running	0%	<a href="#">8088:80</a> 	2 hours ago	
<input type="checkbox"/>	 <a href="#">myapache</a> 48a01441050d	<a href="#">httpd</a>	Running	0%	<a href="#">8080:80</a> 	1 hour ago	

Si se accede desde el navegador al localhost con puerto 8080 se obtiene la pagina por defecto de apache.

- Ingresar al contenedor httpd:  
> Docker exec -it myapache bash

### docker exec

**docker:** Es el comando básico para interactuar con Docker.

**exec:** Este subcomando se utiliza para ejecutar un comando dentro de un contenedor que ya está en funcionamiento.

### -it

Esta opción combina dos flags:

**-i o --interactive:** Mantiene la entrada estándar abierta. Esto significa que el contenedor puede recibir entradas desde tu terminal.

**-t o --tty:** Asigna un terminal virtual al contenedor. Esto es útil para hacer que la interacción con el contenedor sea más familiar, como si estuvieras usando una terminal normal en la máquina.

### myapache

**myapache:** Es el nombre del contenedor con el se quiere interactuar.

**bash**

**bash:** Es el comando que se ejecutara dentro del contenedor. Iniciando una sesión de shell bash dentro del contenedor. Esto te permite interactuar con el sistema de archivos y los procesos del contenedor como si se trabajara en un entorno de línea de comandos directamente en ese sistema.

```
PS C:\Users\IngAR> docker exec -it myapache bash
root@48a01441050d:/usr/local/apache2#
```

Se ingresa a un contenedor Linux

```
#pwd
/usr/local/apache2
root@48a01441050d:/usr/local/apache2#
```

# ls

```
root@48a01441050d:/usr/local/apache2# ls
bin build cgi-bin conf error htdocs icons include logs modules
```

# cd htdocs

Es el directorio donde se guardan los archivos html públicos.

```
root@48a01441050d:/usr/local/apache2# cd htdocs
root@48a01441050d:/usr/local/apache2/htdocs# ls
index.html
```

# cat index.html \

Para ver el contenido del archivo

```
root@48a01441050d:/usr/local/apache2/htdocs# cat index.html
<html><body><h1>It works!</h1></body></html>
root@48a01441050d:/usr/local/apache2/htdocs#
```

### Paso 03: Crear un nuevo contenedor apache con una WEB específica.

- Descargue un Template HTML CSS










Por ejemplo: <https://www.free-css.com/free-css-templates/page295/makaan>

- Desde el powershell, ingrese a la carpeta del template.

```
> docker run --name myweb -dit -p 3600:80 -v ${PWD}:/usr/local/apache2/htdocs/ httpd
```

```
PS C:\Users\IngAR\Downloads\real-estate-html-template> docker run --name myweb -dit -p 3600:80 -v ${PWD}:/usr/local/apache2/htdocs/ httpd
aa56927c3029b38e814899af833935d6247ba20abb2f65303b588b26a80ebfb9
```

En Docker Desktop se ve la nueva instancia activa y de nombre myweb

<input type="checkbox"/>	Name	Image	Status	CPU (%)	Port(s)	Last started	Actions
<input type="checkbox"/>	welcome-to-5066dbef3d57	<a href="#">docker/welcome-to</a>	Running	0%	<a href="#">8088:80</a>	11 hours ago	  
<input type="checkbox"/>	myapache-48a01441050d	<a href="#">httpd</a>	Running	0%	<a href="#">8080:80</a>	10 hours ago	  
<input type="checkbox"/>	myweb-aa56927c3029	<a href="#">httpd</a>	Running	0.01%	<a href="#">3600:80</a>	2 minutes ago	  

Lo que se corrobora con el localhost:



#### Paso 04: Crear un archivo DockerFile.

Los archivos Dockerfile son scripts de configuración utilizados por Docker para automatizar el proceso de creación de imágenes de contenedores. Esencialmente, un Dockerfile contiene una serie de instrucciones que Docker ejecuta paso a paso para construir una imagen que puede ser utilizada para crear contenedores. Cada instrucción en un Dockerfile añade una nueva capa a la imagen, y cada capa se guarda en caché, lo que permite una reconstrucción rápida si el Dockerfile se modifica y se vuelve a construir.

- Detener todas las imágenes activas, desde el powershell

```
> docker stop $(docker ps -aq)
```

```
PS C:\Users\IngAR> docker stop $(docker ps -aq)
aa56927c3029
48a01441050d
5066dbef3d57
```

```
> docker ps
```

```
PS C:\Users\IngAR> docker ps
CONTAINER ID   IMAGE      COMMAND                  CREATED        STATUS        PORTS      NAMES
```

```
> docker ps -a
```

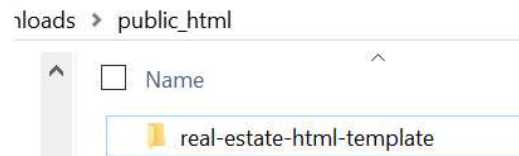
Muestra los contenedores que fueron detenidos, para eliminarlos:

```
> docker rm $(docker ps -aq)
```

```
PS C:\Users\IngAR> docker rm $(docker ps -aq)
aa56927c3029
48a01441050d
5066dbef3d57
```

- Creación del Dockerfile:

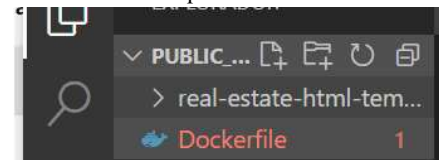
Crear una carpeta con el nombre public\_html y guardar allí el template que se descargó.



- Abra el editor de código de su preferencia, en este caso Visual Studio Code con el complemento de Docker, y arrastre la carpeta allí:



- Cree un archivo que se llama Dockerfile



Escriba el siguiente código en el archivo y guárdelo:

```
FROM httpd
COPY ./real-estate-html-template/ /usr/local/apache2/htdocs/
```



```
Dockerfile > FROM
1 FROM httpd
2 COPY ./real-estate-html-template/ /usr/local/apache2/htdocs/
```

#### Paso 05: Hacer uso del archivo Dockerfile

- Desde el powershell y estando dentro de la carpeta public\_html  
> docker build -t docker-apache .

```
PS C:\Users\IngAR\Downloads\public_html> docker build -t docker-apache .
[+] Building 1.0s (7/7) FINISHED
=> [internal] load build definition from Dockerfile
=> => transferring dockerfile: 109B
=> [internal] load metadata for docker.io/library/httpd:latest
=> [internal] load .dockerignore
=> => transferring context: 2B
=> [internal] load build context
=> => transferring context: 1.58MB
=> CACHED [1/2] FROM docker.io/library/httpd:latest
=> [2/2] COPY ./real-estate-html-template/ /usr/local/apache2/htdocs/
=> exporting to image
=> => exporting layers
=> => writing image sha256:677c2212a0b378ec82f506ed520b831c95296f41ae1d8dd6cda6352252f2853d
=> => naming to docker.io/library/docker-apache

View build details: docker-desktop://dashboard/build/default/default/0tv99mriuxbr77aalm1d33qzf

What's Next?
View a summary of image vulnerabilities and recommendations → docker scout quickview
PS C:\Users\IngAR\Downloads\public_html>
```

Sea creado una nueva imagen:

>Docker images

```
PS C:\Users\IngAR\Downloads\public_html> docker images
REPOSITORY          TAG             IMAGE ID        CREATED         SIZE
docker-apache       latest         677c2212a0b3   About a minute ago  150MB
httpd               latest         fa0099f1c09d   7 days ago     148MB
docker/welcome-to-docker latest        c1f619b6477e   5 months ago    18.6MB
PS C:\Users\IngAR\Downloads\public_html>
```

Desde Docker Desktop

<input type="checkbox"/>	Name	Tag	Status	Created	Size	Actions
<input type="checkbox"/>	<a href="#">docker-apache</a>	latest	Unused	0 seconds ago	149.7 MB	<a href="#">▶</a> <a href="#">⋮</a> <a href="#">🗑</a>

Crear un contenedor para ejecutar la imagen:

> docker run -p 3900:80 --name myapach2 -d docker-apache

```
PS C:\Users\IngAR\Downloads\public_html> docker run -p 3500:80 --name myapach -d docker-apache
8f2e1b4e696e0a1a166f629f4a74a9cf4d182f5f323f47673fd316e7f4471a87
```

Ingresando desde el navegador al localhost y Puerto 3500:



# Find A Perfect Home To Live With Your

#### Paso 06: Crear más contenedores con la misma imagen:

- Desde el powershell y la carpeta public\_html

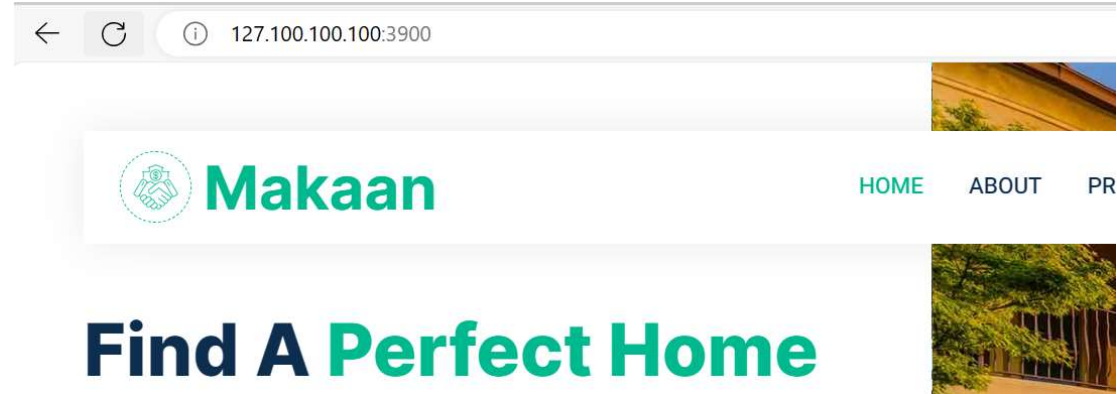
```
> docker run -p 3900:80 --name myapach2 -d docker-apache
```

```
PS C:\Users\IngAR\Downloads\public_html> docker run -p 3900:80 --name myapach2 -d docker-apache
fd34486466b373d4ca69ba817adc4777d32e342fa3e775fc7a13aa0251d13b08
PS C:\Users\IngAR\Downloads\public_html>
```

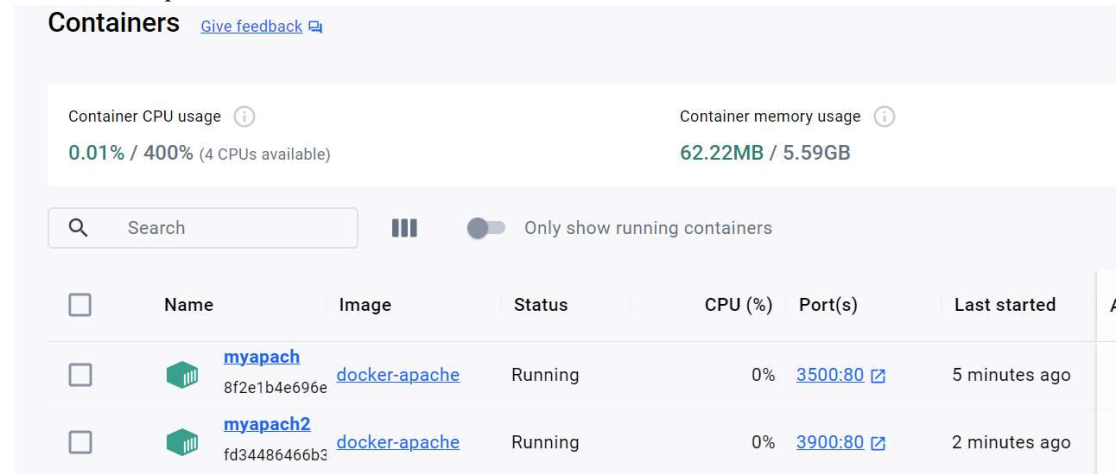
```
>docker ps
```

```
PS C:\Users\IngAR\Downloads\public_html> docker ps
CONTAINER ID   IMAGE          COMMAND                  CREATED        STATUS        PORTS                               NAMES
fd34486466b3   docker-apache  "httpd-foreground"      About a minute ago    Up About a minute    0.0.0.0:3900->80/tcp              myapach2
8f2e1b4e696e   docker-apache  "httpd-foreground"      4 minutes ago        Up 4 minutes        0.0.0.0:3500->80/tcp              myapach
```

Ingresando al nuevo contenedor con el puerto 3900:



Docker Desktop muestra:



### Discusión Posterior:

Después de completar el ejercicio práctico, responda las siguientes preguntas:

¿Cómo puedes asegurar un contenedor Docker en un entorno de producción, y qué prácticas se deben implementar para minimizar la superficie de ataque en el contenedor y en la imagen de Docker?

Explica cómo podrías configurar Docker para usar almacenamiento persistente en un clúster Kubernetes, incluyendo detalles sobre cómo gestionar el almacenamiento dinámico mediante Persistent Volumes y Persistent Volume Claims.

Detalla el proceso y las consideraciones necesarias para migrar una aplicación monolítica a contenedores Docker, incluyendo la descomposición de componentes, manejo de dependencias y estrategias para la gestión del estado.

¿Cómo se pueden implementar y gestionar redes seguras en Docker, específicamente para contenedores que deben comunicarse entre múltiples hosts Docker en diferentes redes? Incluye una explicación sobre el uso de Docker Swarm o Kubernetes para la gestión de redes.

Describe un flujo de trabajo de CI/CD utilizando Docker, Jenkins y GitHub, incluyendo cómo se pueden construir imágenes Docker automáticamente en respuesta a cambios de código en un repositorio de GitHub y cómo estas imágenes pueden ser desplegadas automáticamente a un entorno de prueba o producción.

Analiza las diferencias entre CMD y ENTRYPOINT en un Dockerfile, y proporciona ejemplos de cuándo sería más apropiado usar uno sobre el otro. Además, explica cómo estos comandos interactúan con los argumentos pasados al iniciar un contenedor.

Explora las implicaciones de seguridad de ejecutar aplicaciones en contenedores Docker como usuario root dentro del contenedor, y cómo el uso de User Namespaces en Docker puede ayudar a mitigar los riesgos asociados.

¿Cómo se pueden utilizar las capacidades de autoescala de Docker en un entorno de producción, específicamente en relación con Docker Swarm o Kubernetes, para manejar variaciones en la carga de trabajo? Incluye una discusión sobre las métricas y los parámetros que se deberían monitorizar y utilizar para tomar decisiones de escalado.

**Realice un informe donde documenta todo el proceso sin olvidar conclusiones y bibliografía.**