TC4031.10

Cómputo en la nube (Gpo 10)

Tarea 3. Crear un contenedor Docker

Jaime Alejandro Mendivil A. A01253316

8 de Febrero del 2026

Introducción

Esta actividad es continuación de la práctica anterior pero implementando la creación de un contenedor mediante Docker con nuestro servidor web. Deberemos poder integrar el contenido de nuestro proyecto anterior en una imagen oficial de Apache Web Server,

Instalación de Docker

Empezamos con la descarga de Docker desktop para Windows.

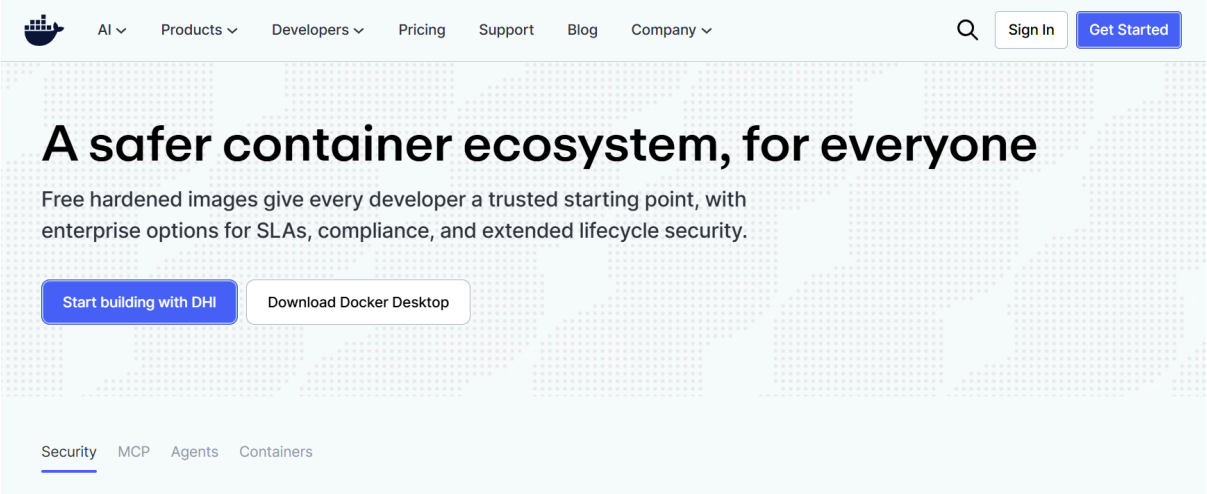

Figura 1. Página de descarga de Docker Desktop.

Descargamos Docker desktop ya que facilita visualizar que imágenes y contenedores tenemos corriendo mediante la interfaz gráfica de Docker desktop.

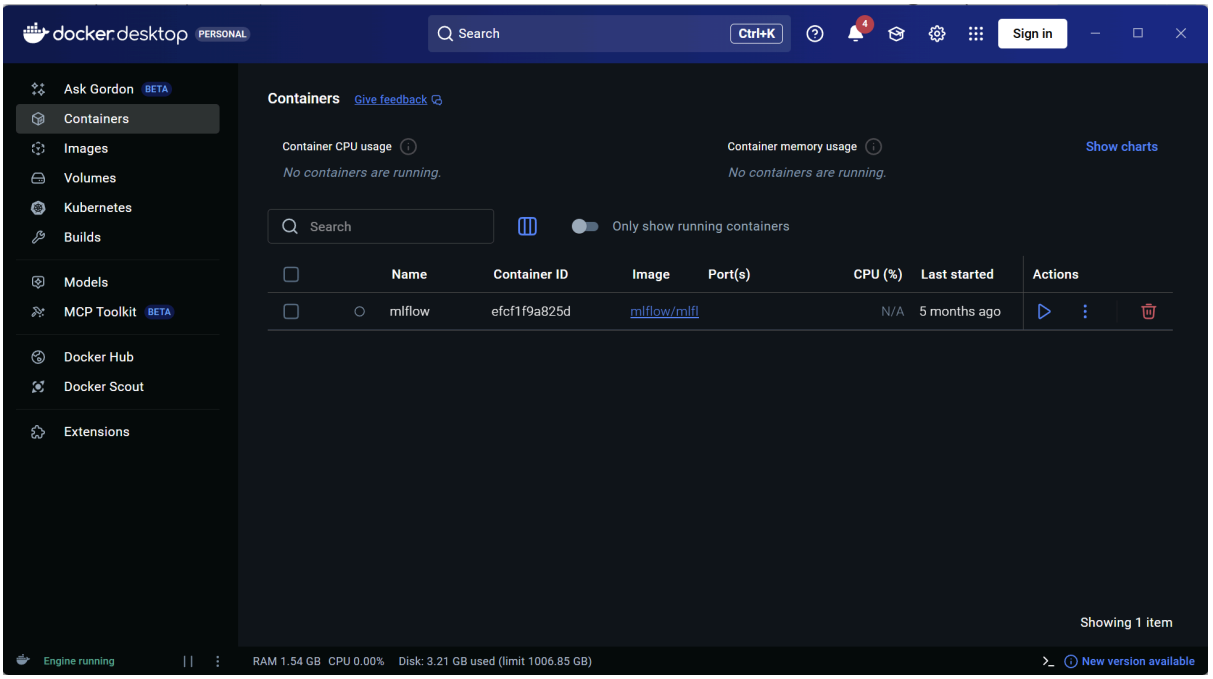

Figura 2. Pantalla de Docker Desktop

Una vez instalado podemos visualizar si tenemos algún contenedor corriendo dentro de nuestro equipo.

Preparación del contenedor

Como este proyecto es muy similar a la entrega anterior solo hay que copiar los archivos a una carpeta “public_html”

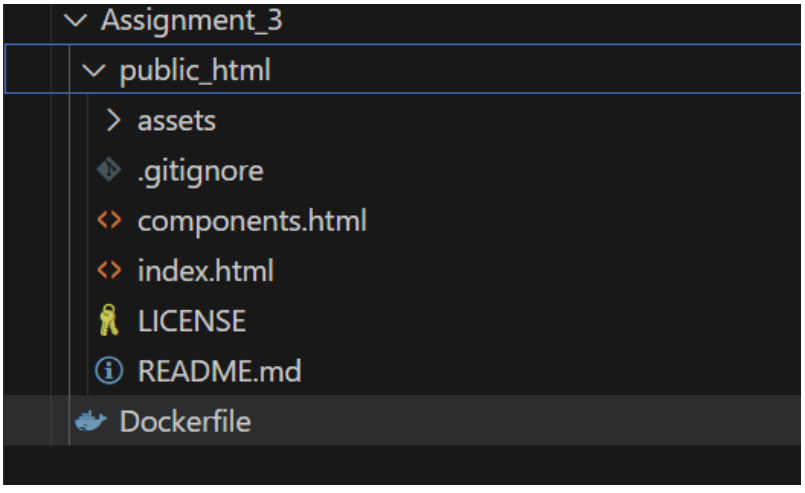

Figura 3. Archivos del sitio web

Adicionalmente creamos un archivo Dockerfile para empezar a empaquetar nuestro contenedor.

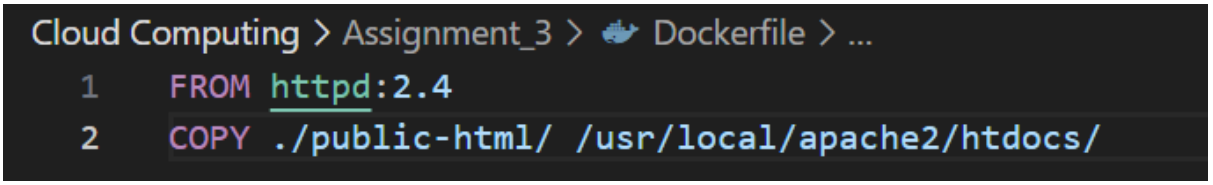

Figura 4. Contenido del Dockerfile

Basándonos en la documentación de la imagen de Docker para el servidor web de Apache, tenemos que crear un archivo dockerfile con los comandos anteriores, cambiar al directorio donde está nuestro proyecto y ejecutar los siguientes comandos:

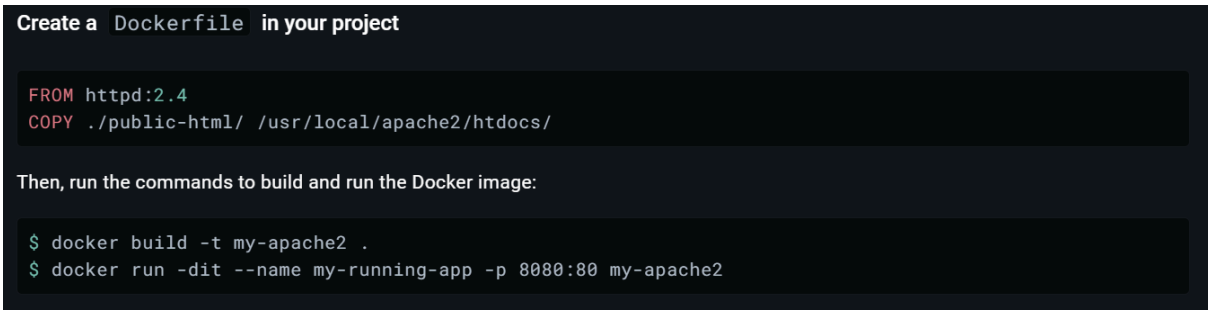

Figura 5. Comandos para crear contenedor de Docker.

Nos movemos de directorio en nuestro IDE y ejecutamos los comandos.

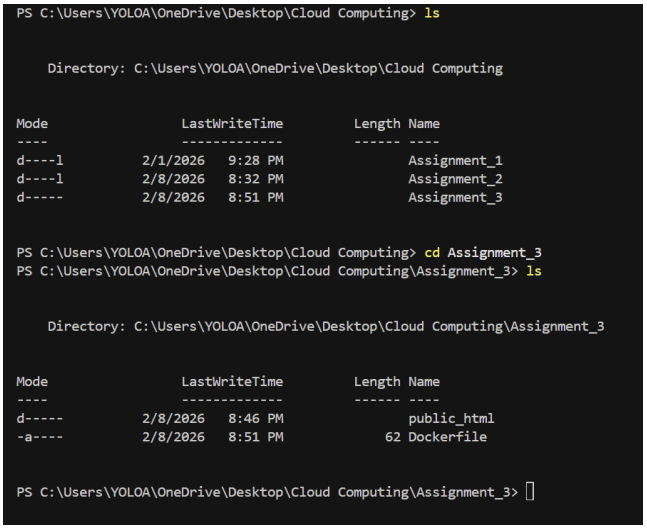

Figura 6. Cambio de directorio

Creación del contenedor

```
PS C:\Users\YOLOA\OneDrive\Desktop\Cloud Computing\Assignment_3> docker build -t my-apache2 .
[+] Building 1.9s (7/7) FINISHED                                                                    docker:desktop-linux
 => [internal] load build definition from Dockerfile                                                              0.0s
 => => transferring dockerfile: 99B                                                                               0.0s
 => [internal] load metadata for docker.io/library/httpd:2.4                                                      0.4s
 => [internal] load .dockerignore                                                                                 0.0s
 => => transferring context: 2B                                                                                   0.0s
 => [internal] load build context                                                                                 0.6s
 => => transferring context: 9.41MB                                                                               0.6s
 => [1/2] FROM docker.io/library/httpd:2.4@sha256:b89c19a390514d6767e8c62f29375d0577190be448f63b24f5f11d6b03f7bf18  0.2s
 => => resolve docker.io/library/httpd:2.4@sha256:b89c19a390514d6767e8c62f29375d0577190be448f63b24f5f11d6b03f7bf18  0.0s
 => [2/2] COPY ./public-html/ /usr/local/apache2/htdocs/                                                          0.1s
 => exporting to image                                                                                            0.6s
 => => exporting layers                                                                                           0.4s
 => => exporting manifest sha256:3c977a8e00e4224f4741e39e2eac667e8c2c0ef502873514eee054ea02363f95                 0.0s
 => => exporting config sha256:697ba3e628bbc147b1ff16d5cdecf2c11d53086110e395d77bcf8da9a7894628                   0.0s
 => => exporting attestation manifest sha256:267d5924de2e9c559814c8b08358522012290dc0a48ad7b9537af3cf70ed6223     0.0s
 => => exporting manifest list sha256:ad9e85ba117f41c618c1efd70f017d2bcefe9c10b888f5129bf5656748cdb3b5            0.0s
 => => naming to docker.io/library/my-apache2:latest                                                              0.0s
 => => unpacking to docker.io/library/my-apache2:latest                                                           0.1s
PS C:\Users\YOLOA\OneDrive\Desktop\Cloud Computing\Assignment_3>
```

Figura 7. Resultados del primer comando

Corrimos el comando anterior y provocó errores pero se soluciono al cambiar el nombre de la carpeta “public_html” por “public-html”.

```
PS C:\Users\YOLOA\OneDrive\Desktop\Cloud Computing\Assignment_3> docker run -dit --name my-running-app -p 8080:80 my-apache2
781a771a1ba5ba8b6fff80f8e85ad8fb0eecf17e35212236f8d9ec71df8a2060
```

Figura 8. Resultados del segundo comando

Ahora ya estamos corriendo nuestro contenedor personalizado de docker y lo podemos observar en Docker Desktop.

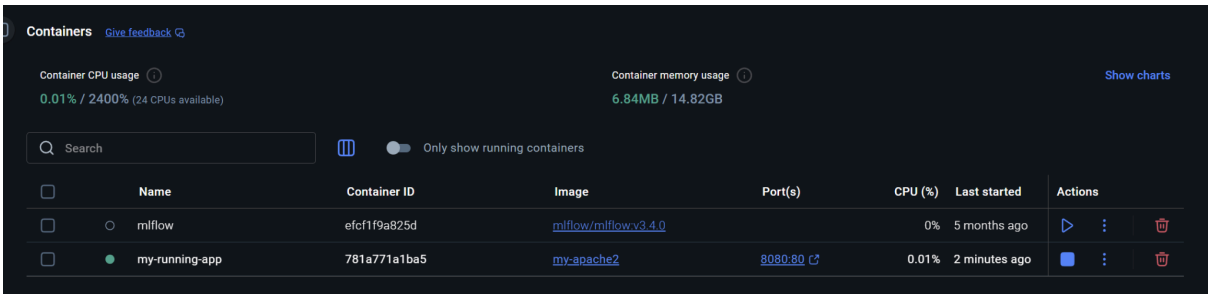

Figura 9. Contenedor corriendo

De esta forma podemos acceder al puerto expuesto y visualizar nuestra app.

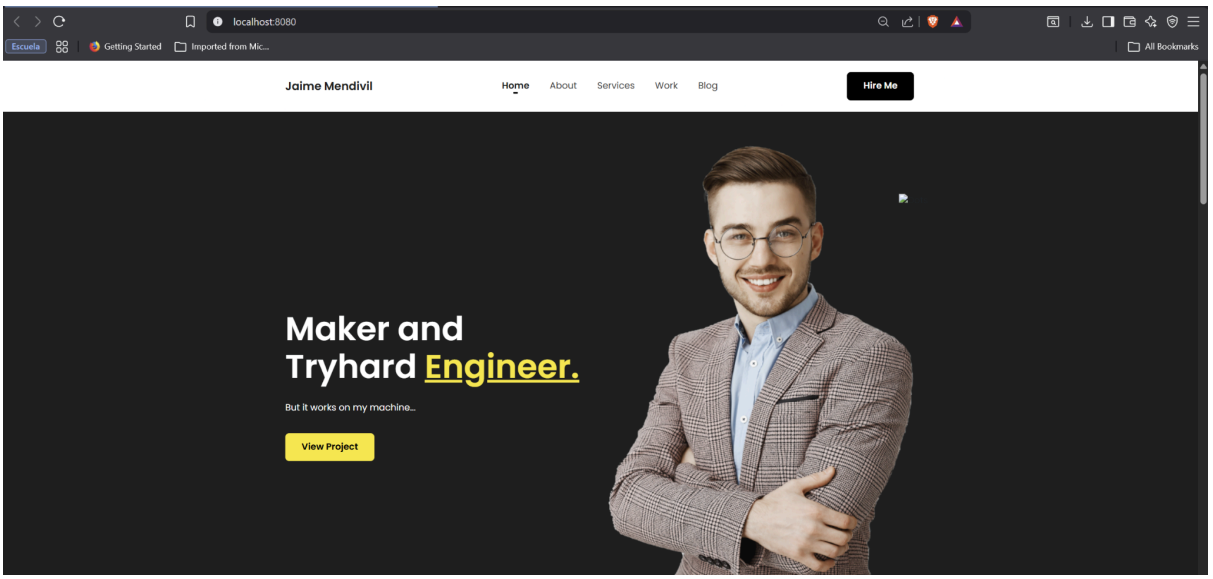

Figura 10. Contenedor corriendo exitosamente

Y como podemos observar fue un éxito.

Personalización de la página web

A continuación se descarga el archivo proporcionado con el template de página web, y se edita el archivo de Index.html para ajustar los datos a nuestro caso. Como esta parte es idéntica a la entrega anterior solamente se copiaron los archivos ya existentes y se copiaron a la carpeta “public-html”.

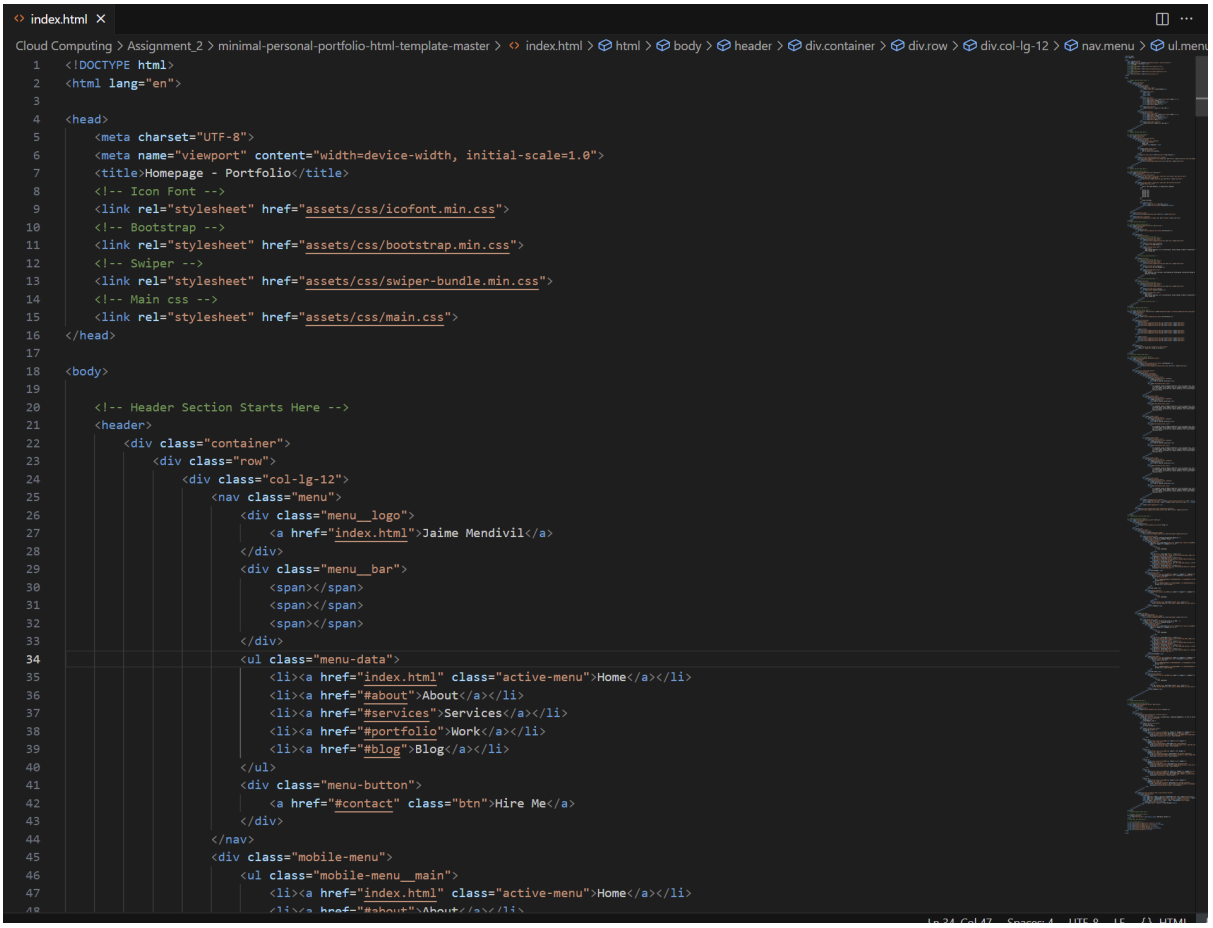

Figura 11. Datos del sitio web.

Carga del sitio web al contenedor

Para poder hacer uso de nuestra página web en un contenedor de Docker, hacemos Pull de la versión más actual de la imagen HTTPD (Apache Web Server) y al leer la documentación hacemos los cambios necesarios para integrar esta imagen preexistente con nuestro proyecto anterior.

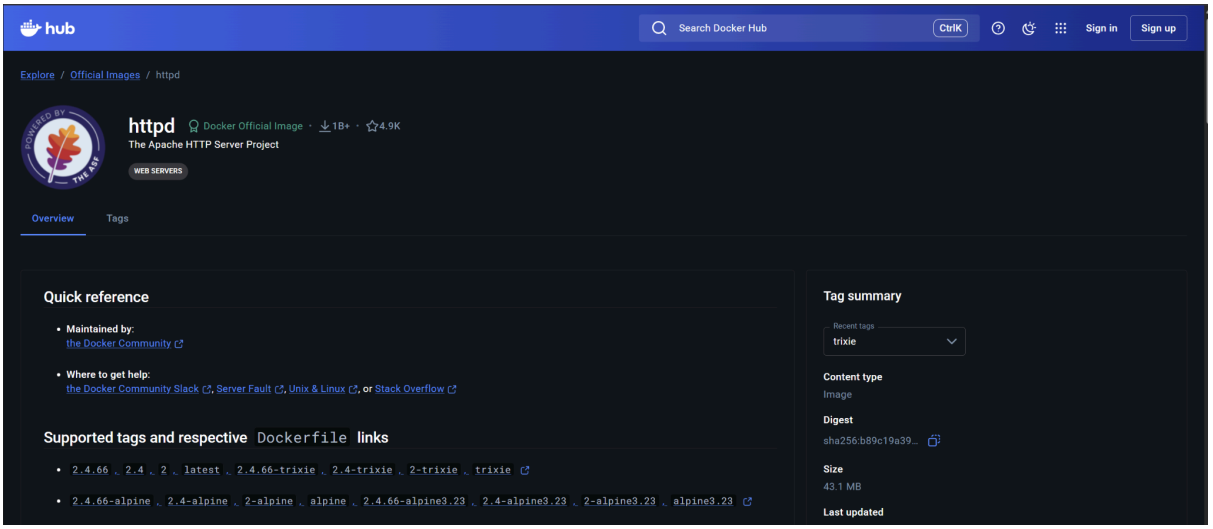

Figura 12. Imagen oficial de Apache Web Server en Docker Hub

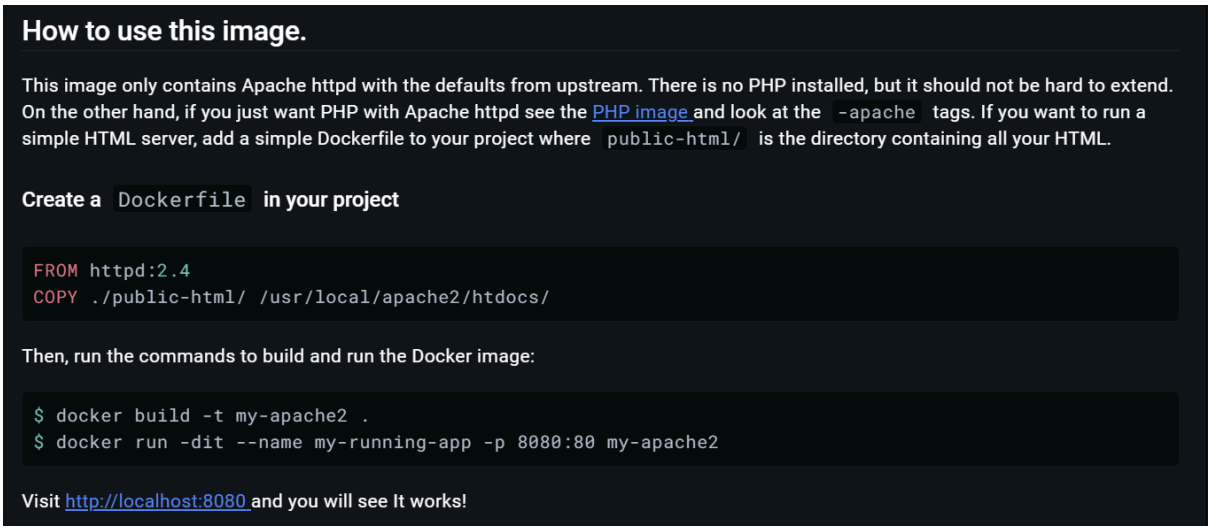

Figura 13. Comandos a seguir

Siguiendo los pasos generamos fácilmente nuestro contenedor personalizado.

Resultados obtenidos

Pudimos generar de forma exitosa nuestro contenedor de Docker y correrlo para visualizar nuestra página web. Los siguientes pasos serían publicar nuestra imagen de Docker para que se pueda hacer Pull y correr nuestra aplicación en cualquier dispositivo que cuente con Docker, paso fundamental para hacer aplicaciones listas para la nube.

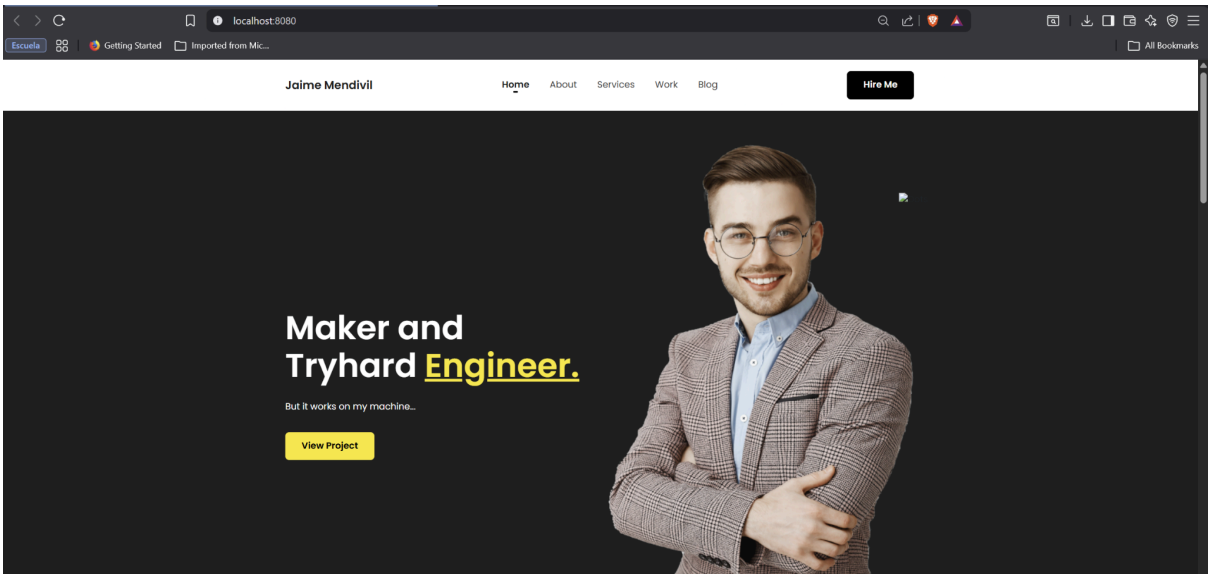

Figura 14. Contenedor corriendo exitosamente

Conclusión

A lo largo de esta práctica logramos realizar exitosamente un contenedor personalizado de Docker, cosa que nos será muy importante para prácticas futuras en donde empaquetamos y despleguemos nuestras aplicaciones en la nube. El hacer uso de Docker de forma correcta nos ayudará a poder escalar de mejor manera cualquier aplicación en la nube ya que podemos hacer uso de orquestación de contenedor mediante y Docker Compose y Kubernetes.

Bibliografía

Debian Project. (2024). Debian GNU/Linux documentation. https://www.debian.org/doc/

Apache Software Foundation. (2024). Apache HTTP Server documentation. https://httpd.apache.org/docs/

Apache Software Foundation. (2024). Apache HTTP Server project. https://httpd.apache.org/

Docker. (2024). Docker documentation. https://docs.docker.com/

Docker Hub. (2024). Official Apache HTTP Server (httpd) Docker image. https://hub.docker.com/_/httpd

WinSCP Team. (2024). WinSCP documentation. https://winscp.net/eng/docs.php

Linux Foundation. (2023). iproute2 – Linux networking utilities. https://wiki.linuxfoundation.org/networking/iproute2