



**Tecnológico
de Monterrey**

Escuela de Ingeniería y Ciencias

Cómputo en la Nube

Creación de un contenedor Docker

Armando Bringas Corpus

A01200230

1 Introducción

La siguiente práctica tiene como objetivo la implementación de un contenedor que representan el siguiente paso de la evolución de la virtualización para poder crear servicios dentro de máquinas virtuales que sean más pequeñas y puedan consumir menos recursos lo cual representa una ventaja económica con respecto a implementar máquinas virtuales tradicionales. En esta práctica se estará implementando un contenedor en el cual se estará levantando un pequeño servidor web utilizando la tecnología para creación de contenedores llamada Docker, de esta manera podremos analizar las ventajas y desventajas con respecto a implementar una Máquina Virtual usando plataformas como Oracle VirtualBox.

2 Instalación de Docker

El primer paso es instalar Docker que representa una de las aplicaciones más utilizadas y mejores para crear contenedores.

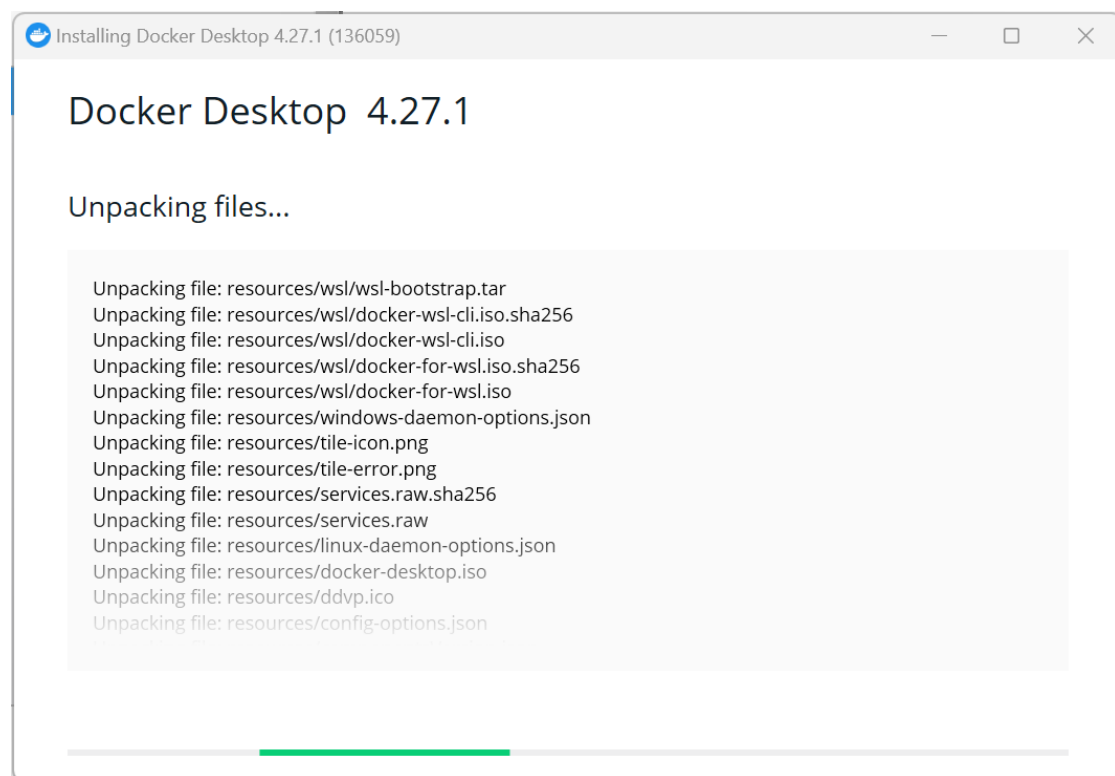


Figura 1: Instalación de Docker

La instalación de Docker a comparación con la de Oracle VirtualBox fue mucho

más sencillas, realizamos la instalación por defecto por lo que no fue requerida alguna configuración especial, como se muestra en la figura 1 observamos una interfaz bastante amigable en la que por el momento no contamos ni con alguna imagen o contenedor creados.

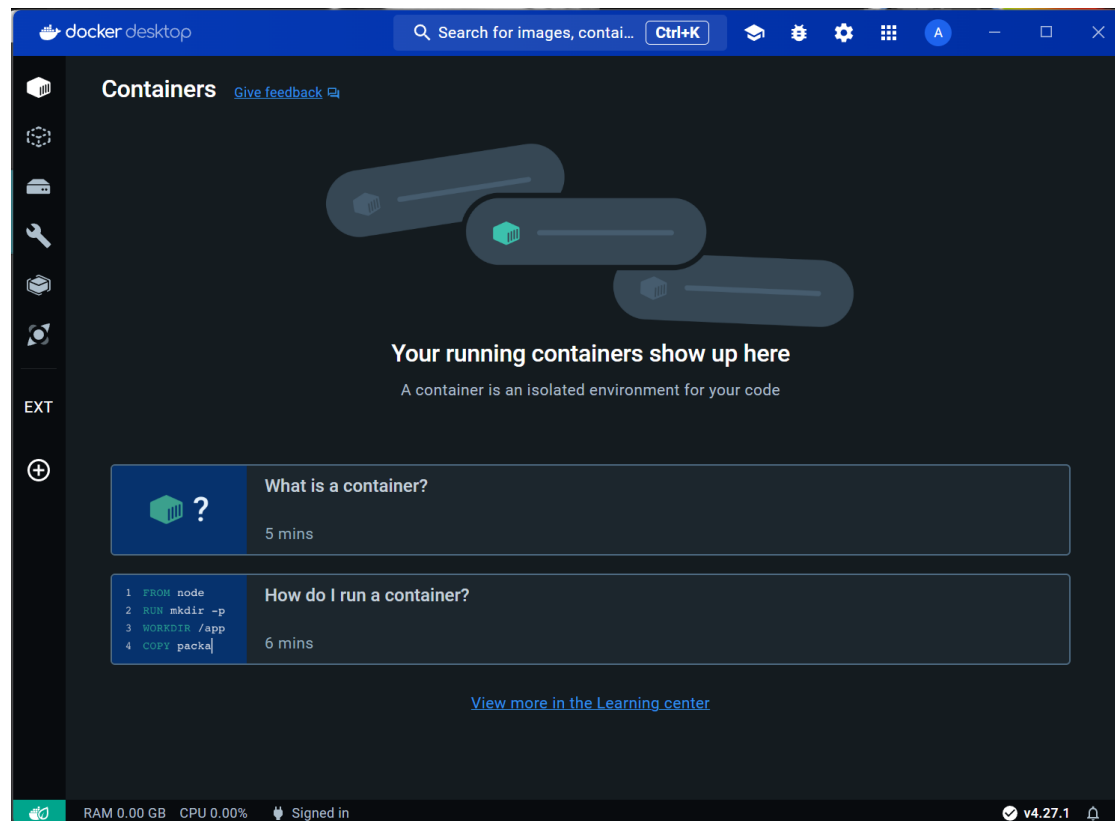


Figura 2: Instalación de Docker

3 Preparación del contenedor

El siguiente paso es la preparación del contenedor donde primero creamos un archivo llamado **Dockerfile** sin extensión, este archivo lo colocamos en la carpeta donde estarán los archivos de nuestro sitio web a mostrar cuya carpeta renombramos como **public-html** y dentro del archivo colocamos el siguiente contenido:

```
1 FROM httpd:2.4
2 COPY ./public-html/ /usr/local/apache2/htdocs/
```

Posteriormente procedemos a crear la imagen del servidor web a través de **Apache** utilizando el servicio **httpd** y procedes a través de la línea de comandos

```
1 docker build -t apache-portafolio-web:1.0 .
```

```
C:\Windows\System32\cmd.e × + ▾
```

```
Microsoft Windows [Version 10.0.22621.3007]
(c) Microsoft Corporation. All rights reserved.
```

```
C:\Users\bring\Documents\Github\Cloud_Computing\M3_Virtualización_y_Contenedores\tarea_3_Creación_Contenedor_Docker>docker build -t apache-portafolio-web:1.0 .
```

```
[+] Building 11.9s (5/7)
```

	docker:default
=> [internal] load build definition from Dockerfile	0.2s
=> => transferring dockerfile: 99B	0.0s
=> [internal] load metadata for docker.io/library/httpd:2.4	4.4s
=> [auth] library/httpd:pull token for registry-1.docker.io	0.0s
=> [internal] load .dockerignore	0.1s
=> => transferring context: 2B	0.0s
=> [internal] load build context	3.1s
=> => transferring context: 9.66MB	3.0s
=> [1/2] FROM docker.io/library/httpd:2.4@sha256:ad01c94aa4ee2a03eba75c84c1e485f62d29d119792c4	7.2s
=> => resolve docker.io/library/httpd:2.4@sha256:ad01c94aa4ee2a03eba75c84c1e485f62d29d119792c4	0.1s
=> sha256:ad01c94aa4ee2a03eba75c84c1e485f62d29d119792c4ed59e66c5b63ae5c91f 8.86kB / 8.86kB	0.0s
=> => sha256:4f4fb700ef54461cf02571ae0db9a0dc1e0cdb5577484a6d75e68dc38e8acc1 32B / 32B	0.2s
=> => sha256:ecf19d91df773a43b1df424b77dfe42e992129de74919ce40f764f57el1fe85 2.04kB / 2.04kB	0.0s
=> => sha256:59bcd61b45fd94a7f19314d259effc3da8bb42a0e157263af2246a22c07e96595 8.02kB / 8.02kB	0.0s
=> => sha256:ef22398cad3c99d77ded290f17cc9b6f84470b11cefcb4e8c9aab5b551980b059 145B / 145B	0.3s
=> => sha256:c57ee5000d61345aa3ee6684794a8110328e2274d9a5ae7855969d1a2639446 29.15MB / 29.15MB	5.3s
=> => sha256:f420b40fd7be14050fbcd6f48809e3d4177f8de6e33c9940ea96d590fd4bc20e 4.01MB / 4.01MB	1.5s
=> => sha256:ea4892b1a58d0b34a8d764c36795e1301a7275fc662e568370f5f457d69d0dd6 26.21MB / 31.20MB	7.0s
=> => sha256:1fe3871b50ff6024c1f7d4a2616126251845ca17fc9565d88d44eaf4d12da382 293B / 293B	1.8s
=> extracting sha256:c57ee5000d61345aa3ee6684794a8110328e2274d9a5ae7855969d1a26394463	1.5s

Finalmente regresamos a nuestra interfaz (GUI) de Docker desktop y observamos que la imagen creada está creada, en este caso con el nombre de **apache-portafolio-web**.

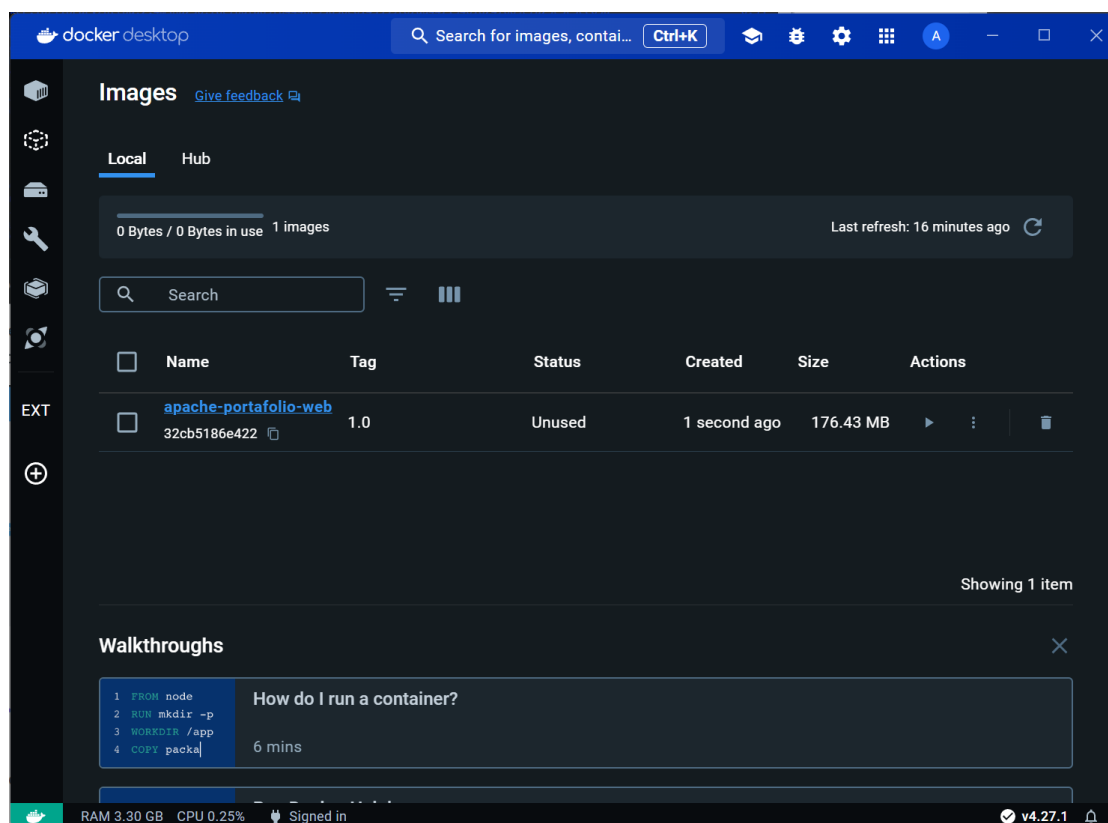


Figura 4: Preparación del Contenedor

4 Creación del contenedor

Posteriormente procedemos a la creación del contenedor con base a la imagen que contendrá nuestro servidor web y sus archivos, esto lo hacemos ejecutando el siguiente comando en la línea de comandos:

```
1 docker run -dit --name portafolio-web-container
2 -p 8080:80 apache-portafolio-web:1.0
```

```
C:\Windows\System32\cmd.e X + - □ X
=> => transferring context: 2B 0.0s
=> [internal] load build context 3.1s
=> => transferring context: 9.66MB 3.0s
=> [1/2] FROM docker.io/library/httpd:2.4@sha256:ad01c94aa4ee2a03eba75c84c1e485f62d29d119792c 16.4s
=> => resolve docker.io/library/httpd:2.4@sha256:ad01c94aa4ee2a03eba75c84c1e485f62d29d119792c4 0.1s
=> => sha256:ad01c94aa4ee2a03eba75c84c1e485f62d29d119792c4ed59e66c5b63ae5c91f 8.86kB / 8.86kB 0.0s
=> => sha256:4f4fb700ef54461cfa02571ae0db9a0dc1e0cdb5577484a6d75e68dc38e8acc1 32B / 32B 0.2s
=> => sha256:ecf19d91df773a43b1df424b77dfe42e992129de74919ce40f764f57e1f1fe85 2.04kB / 2.04kB 0.0s
=> => sha256:59bcd61b45fd94a7f19314d259effc3da8b42a0e157263af2246a22c07e96595 8.02kB / 8.02kB 0.0s
=> => sha256:ef22398cad3c99d77ded290f17cc9b6f84470b11cefb4e8c9aab5b551980b059 145B / 145B 0.3s
=> => sha256:c57ee5000d61345aa3ee6684794a8110328e2274d9a5ae7855969d1a2639446 29.15MB / 29.15MB 5.3s
=> => sha256:f420b40fd7be14050fbcdf648809e3d4177f8de6e33c9940ea96d590fd4bc20e 4.01MB / 4.01MB 1.5s
=> => sha256:ea4892b1a58d0b34a8d764c36795e1301a7275fc662e5683705f457d69d0dd6 31.20MB / 31.20MB 7.4s
=> => sha256:1fe3871b50ff6024c1f7daa2616126251845ca17fc9565d88d44eaf4d12da382 293B / 293B 1.8s
=> => extracting sha256:c57ee5000d61345aa3ee6684794a8110328e2274d9a5ae7855969d1a26394463 5.5s
=> => extracting sha256:ef22398cad3c99d77ded290f17cc9b6f84470b11cefb4e8c9aab5b551980b059 0.0s
=> => extracting sha256:4f4fb700ef54461cfa02571ae0db9a0dc1e0cdb5577484a6d75e68dc38e8acc1 0.0s
=> => extracting sha256:f420b40fd7be14050fbcdf648809e3d4177f8de6e33c9940ea96d590fd4bc20e 0.6s
=> => extracting sha256:ea4892b1a58d0b34a8d764c36795e1301a7275fc662e5683705f457d69d0dd68 3.4s
=> => extracting sha256:1fe3871b50ff6024c1f7daa2616126251845ca17fc9565d88d44eaf4d12da382 0.0s
=> [2/2] COPY ./public-html/ /usr/local/apache2/htdocs/ 0.4s
=> exporting to image 0.2s
=> => exporting layers 0.2s
=> => writing image sha256:32cb5186e422eb903e0d58cf74785ee447a6a7d422c23233885d2cbd894575ec 0.0s
=> => naming to docker.io/library/apache-portafolio-web:1.0 0.0s

What's Next?
View a summary of image vulnerabilities and recommendations → docker scout quickview

C:\Users\bring\Documents\Github\Cloud_Computing\M3_Virtualización_y_Contenedores\Tarea_3_Creación_Contenedor_Docker>docker run -dit --name portafolio-web-container -p 8080:80 apache-portafolio-web:1.0
1bfac00a36dce40170086b4d3794e1f1db6045f078e88d868f135447aa6da217

C:\Users\bring\Documents\Github\Cloud_Computing\M3_Virtualización_y_Contenedores\Tarea_3_Creación_Contenedor_Docker>
```

Figura 5: Creación del Contenedor

A continuación, se da una explicación de los comandos que se utilizaron:

- **docker run**: utilizado para la creación y ejecución del contenedor.
- **-dit**: las opciones que se utilizaron para el comando.
- **--name**: el nombre del contenedor, en este caso se le asigno el de *portafolio-web-container*.
- **-p**: para redireccionar el puerto, en este caso de 8080 a 80.
- **apache-portafolio-web:1.0**: corresponde al nombre de la imagen que es la base para la creación que se hizo del contenedor.

En la siguiente figura 6 podemos verificar en Docker Desktop que el contenedor fue creado exitosamente.

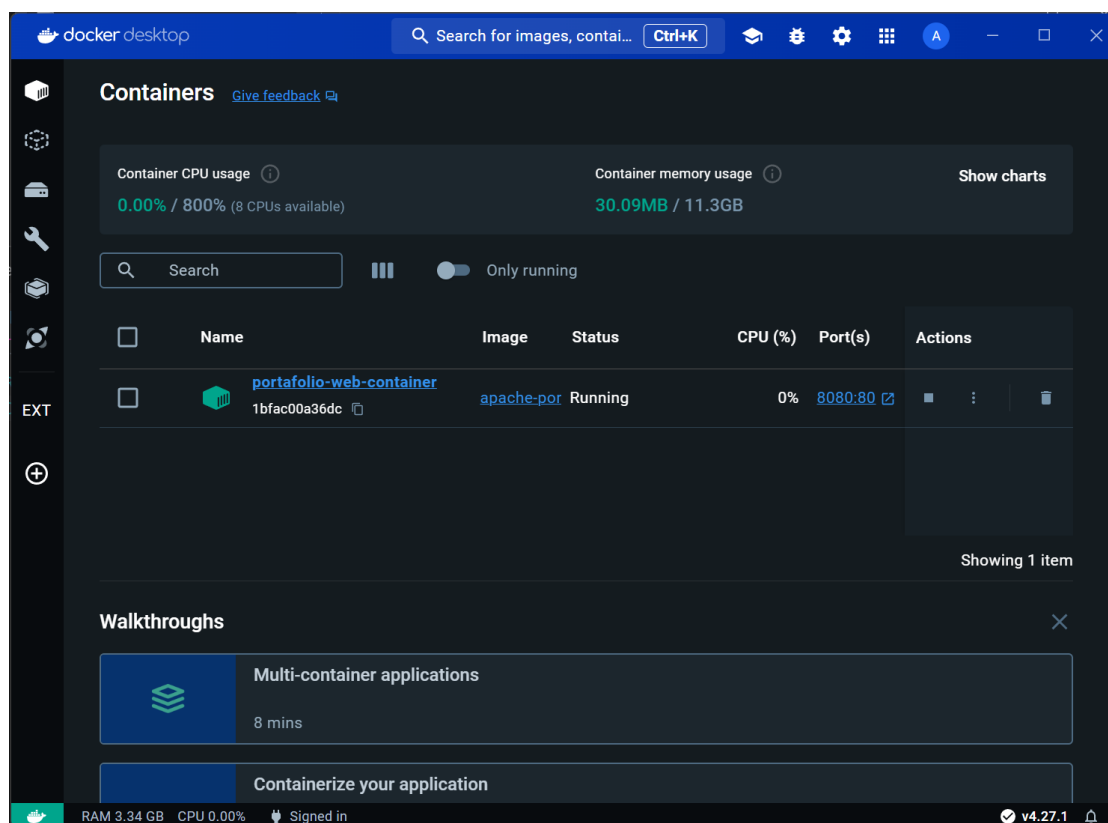
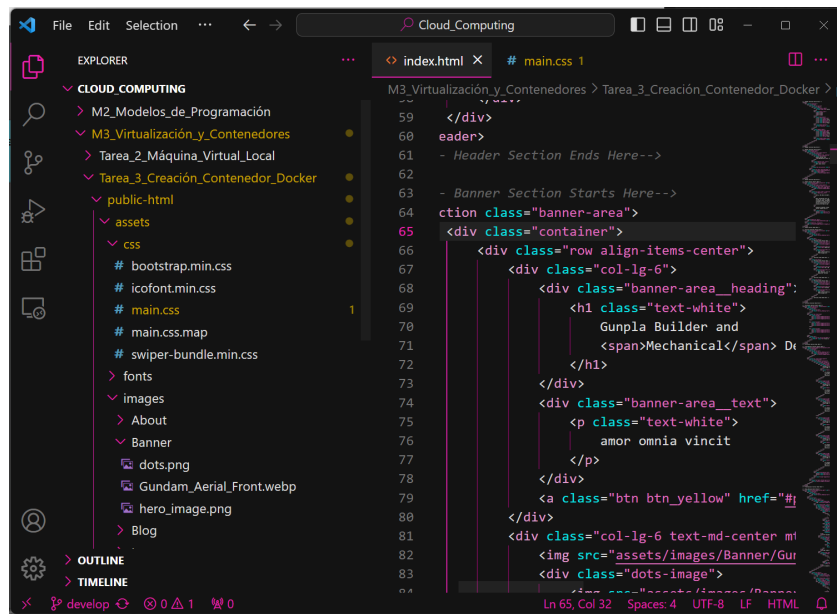


Figura 6: Creación del Contenedor

5 Personalización del sitio web

Al igual que hicimos en la práctica pasada, personalizamos el sitio web como se muestra en la figura 7 utilizando Visual Studio Code. En este caso realizamos nuevas modificaciones para que nuestro ejemplo final fuera diferente al de la práctica pasada. En la figura 8 se pueden observar los cambios realizados en cuando a color de fondo, imagen y descripciones.

Las modificaciones realizadas se pueden consultar en el siguiente repositorio de Github: https://github.com/armandoBringas/Cloud_Computing/tree/main/M3_Virtualizaci%C3%B3n_y_Contenedores/Tarea_3_Creaci%C3%B3n_Contenedor_Docker



```
59 </div>
60 </header>
61 - Header Section Ends Here-->
62
63 - Banner Section Starts Here-->
64 <div class="banner-area">
65   <div class="container">
66     <div class="row align-items-center">
67       <div class="col-lg-6">
68         <div class="banner-area__heading">
69           <h1 class="text-white">
70             Gunpla Builder and
71             <span>Mechanical</span> Design Engineer
72           </h1>
73         </div>
74         <div class="banner-area__text">
75           <p class="text-white">
76             amor omnia vincit
77           </p>
78         </div>
79         <a class="btn btn_yellow" href="#">Ver Proyectos
80       </div>
81       <div class="col-lg-6 text-md-center">
82         
83       </div>
84     </div>
85   </div>
86 </div>
```

Figura 7: Personalización del Sitio Web

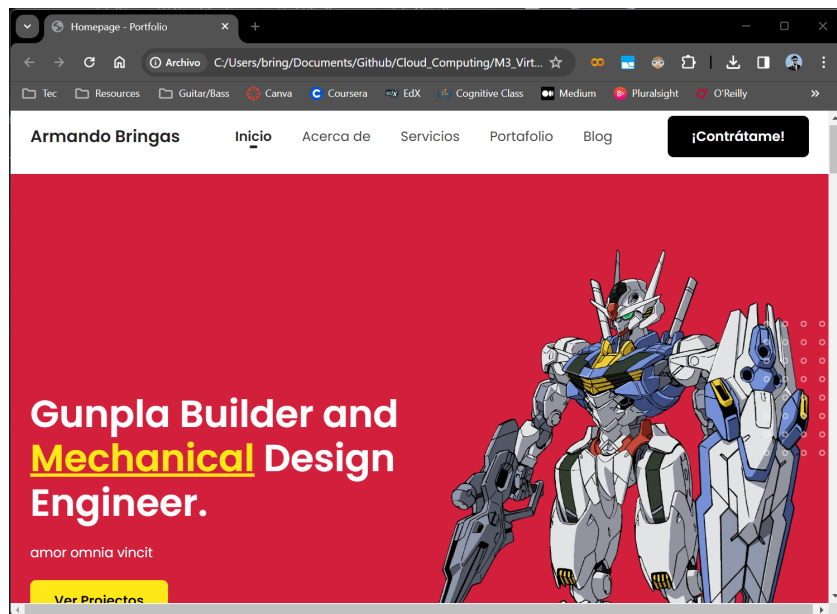


Figura 8: Personalización del Sitio Web

6 Carga del sitio web al contenedor

El paso final que representó el más sencillo fue desplegar el sitio web en el puerto local que designamos cuando hicimos la creación del contenedor. En este caso no fue requerido hacer configuraciones adicionales en comparación con la práctica pasada con la Máquina Virtual en VirtualBox. En este caso lo que es importante es asegurarnos que el contenedor este activo, en caso contrario el sitio web no se desplegará.

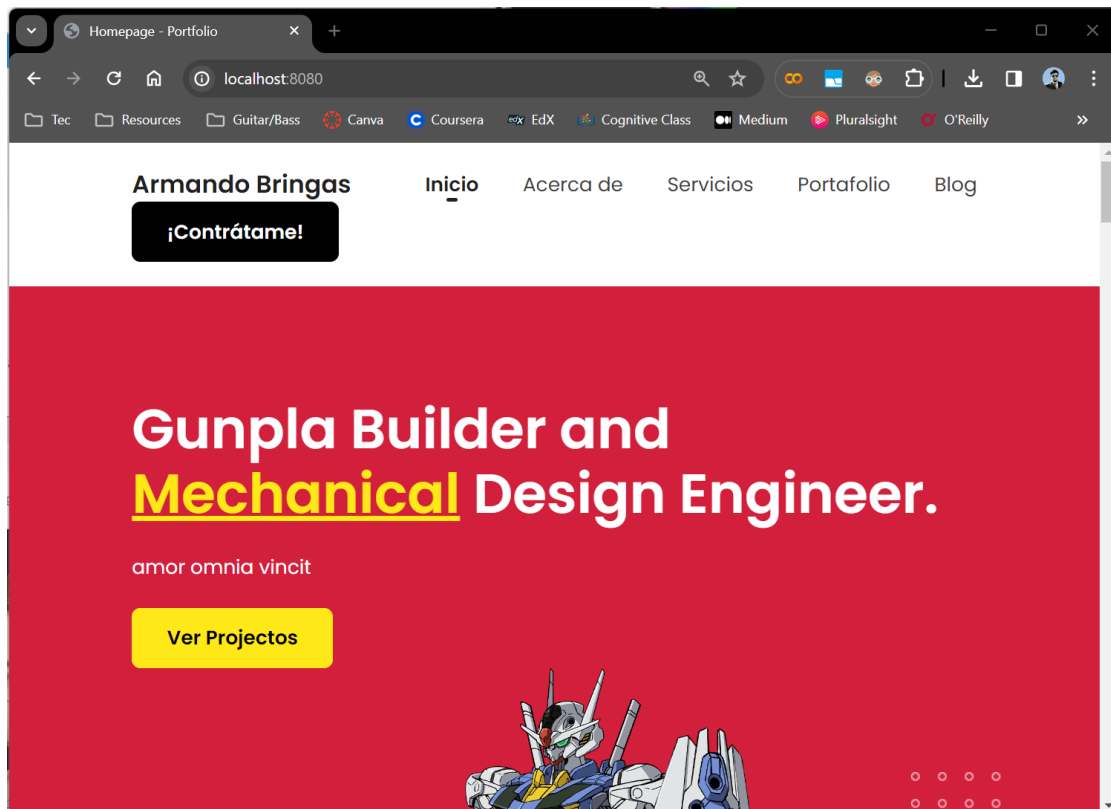


Figura 9: Carga del Sitio Web en el Contenedor

7 Resultados

Finalmente, al igual que hicimos con la práctica pasada observamos que la página web se cargo de forma correcta y que funciona adecuadamente, que es responsiva y no presenta algún error al irse a alguna sección de la página web, como se puede observar en las figuras 10 y 11, la renderización de la página es adecuada, por lo tanto la creación de nuestro contenedor se realizó de forma exitosa.

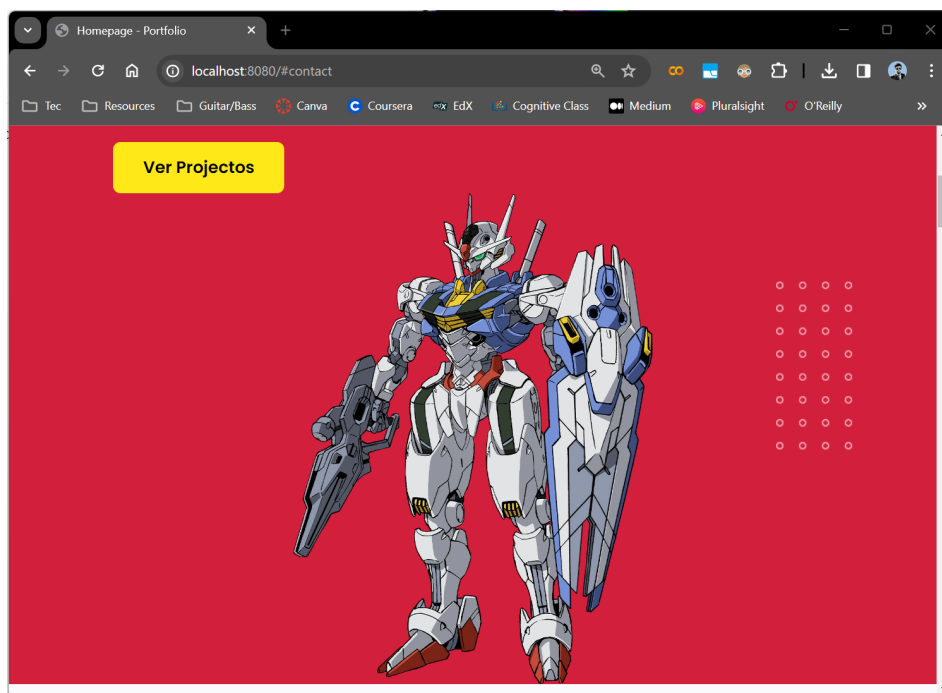


Figura 10: Resultados

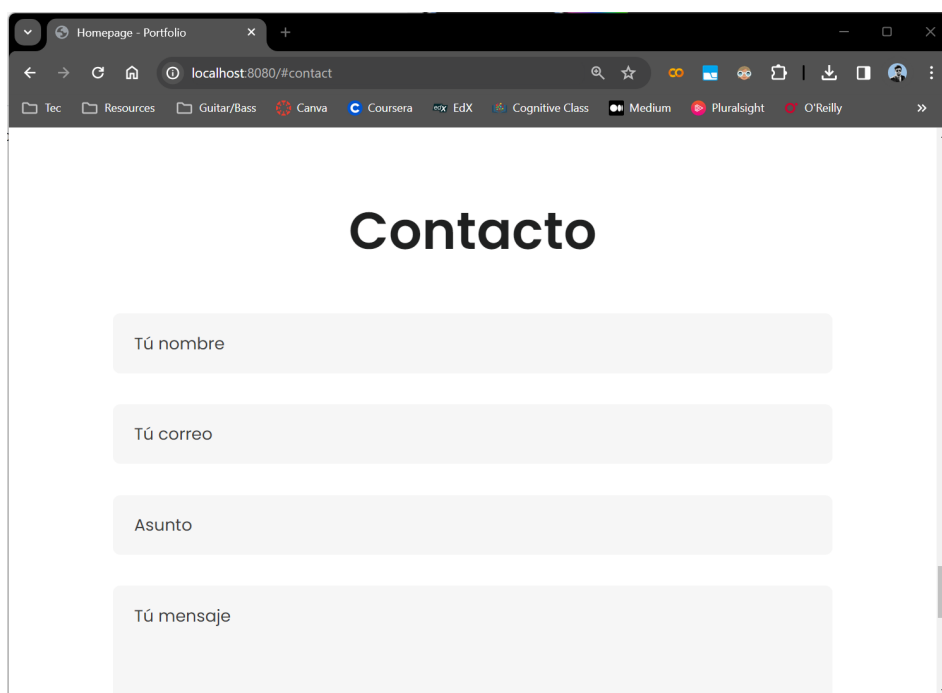


Figura 11: Resultados

8 Reflexión sobre los contenedores

En primera instancia en comparación con la práctica pasada notamos que la creación de la imagen y contenedor fue mucho más sencilla que todo el proceso de hacer una Máquina Virtual y su configuración, por otro lado vemos que los contenedores tienen como ventaja en que podemos optimizar mejor los recursos y adaptarlos a nuestras necesidades específicas.

Otra de las ventajas que podemos observar es que podemos ejecutar varios contenedores en un solo host físico o virtual y nos es necesario administrar el sistema operativo del contenedor. Quizás una desventaja es que los contenedores pueden ser menos flexibles a comparación de una Máquina Virtual cuando se busca configurar características específicas con respecto a la virtualización del hardware como memoria RAM, sistema operativo, etc. Sin embargo, los contenedores para una gran cantidad de aplicaciones son mucho más ligeros y ágiles de implementar.