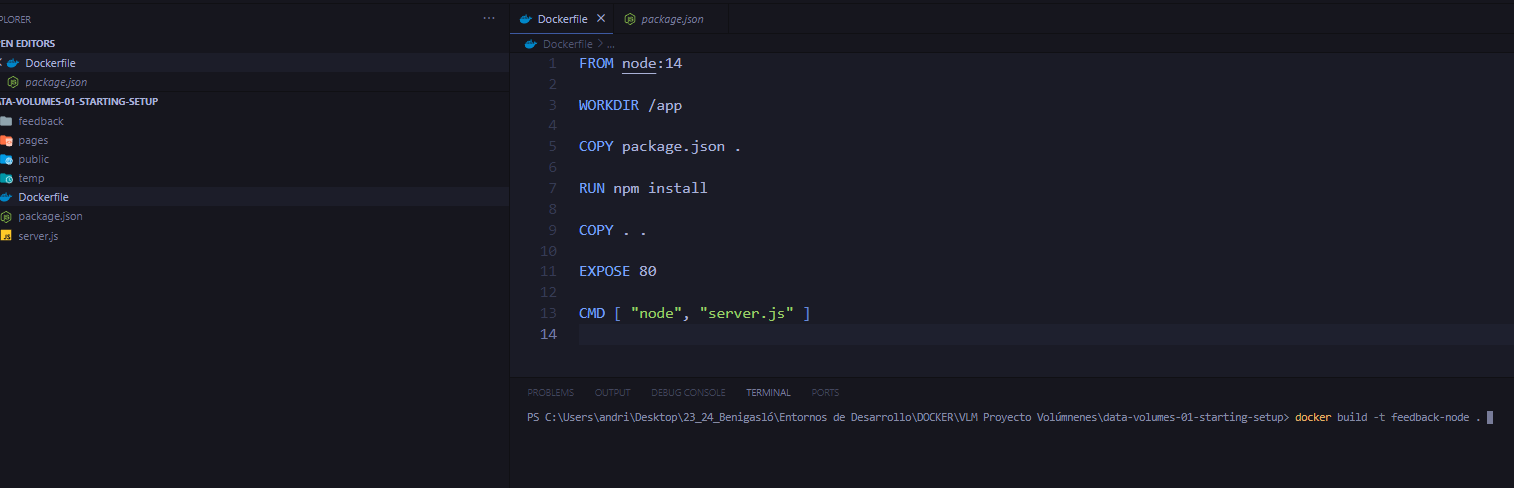
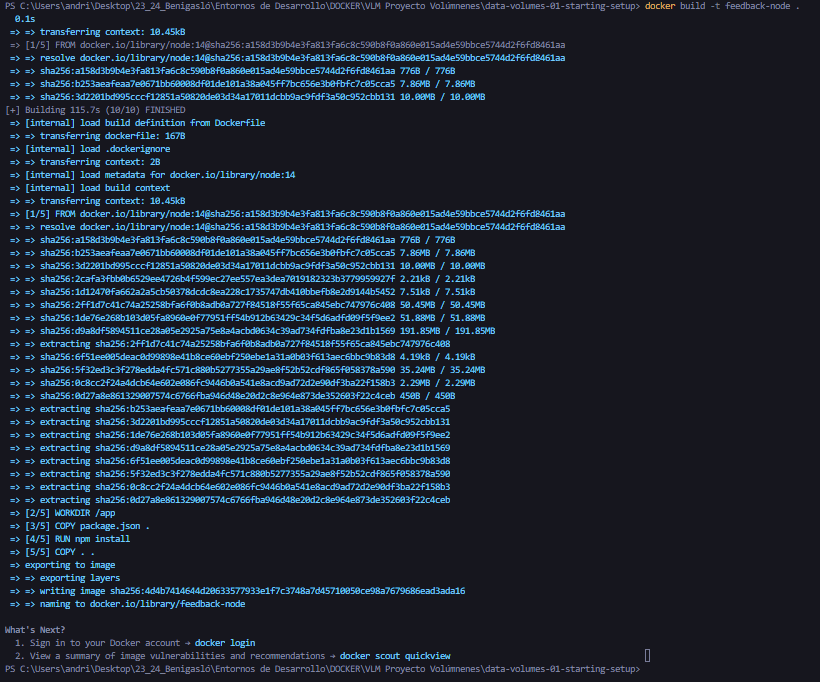
Laboratorio 5. Volumnes. Operaciones con la app NODE

Paso 1. Creamos una carpeta en el directorio le ponemos un nombre proyecto 2Node-Volumenes. Descargamos la carpeta adjunta. Abrimos la carpeta en Visual estudio Code.

Paso 2 . Creamos un Dockerfile

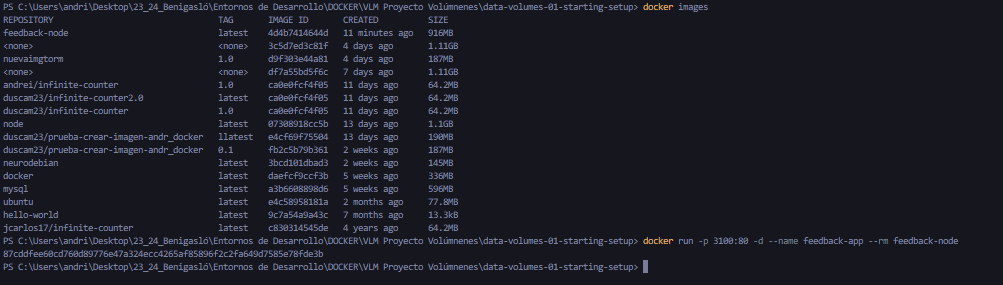


Paso 3. Construimos la imagen.

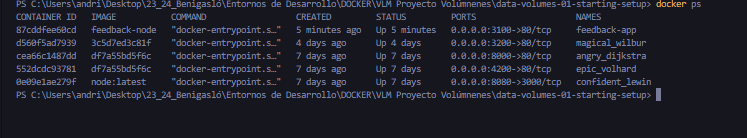
A continuación, podemos ver el resultado (después de unos minutos). 

Paso 4. Iniciar un proceso, es decir un contenedor basado en esa imagen. Configuramos unas cuantas cosas para ello:

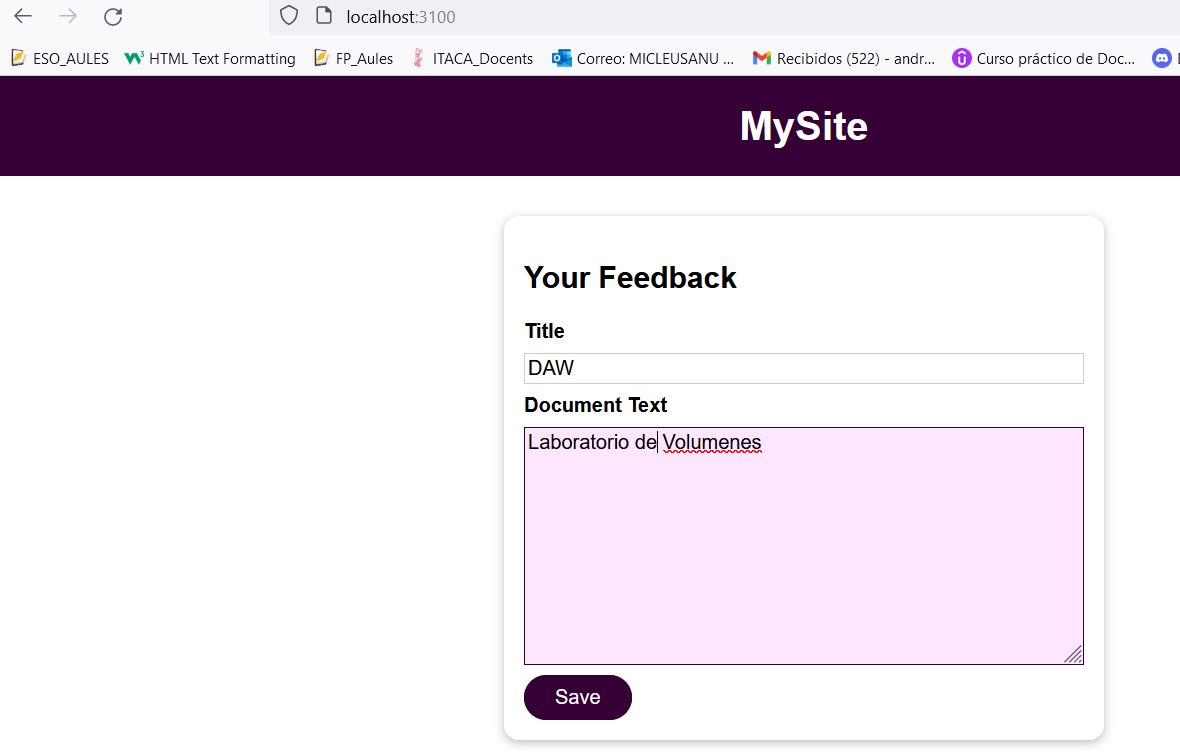
En el puerto local por ejemplo 3100, -**d** segundo plano, **--name** le damos un nombre de **feedback-app** y después le adjuntamos una condición de que se elimine el contenedor (**feedback-node**) una vez se pare, es decir con **--rm**



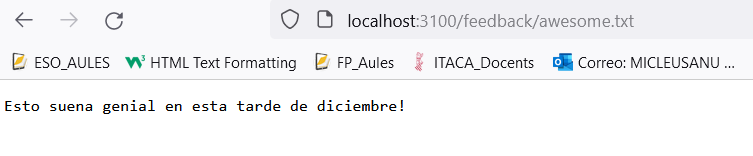
Paso 5. Comprobamos el resultado con docker ps y verificamos en localhost que lo hemos indicado, (en mi caso es 3100).



Paso 6. Comprobar el localhost:



Se puede ver este archivo almacenado usando en localhost la siguiente: Esto suena genial en esta tarde de diciembre !



Paso 7. Detengamos este contenedor:

***docker stop feedback-app***

sí nos devuelve el nombre significa que ha parado.



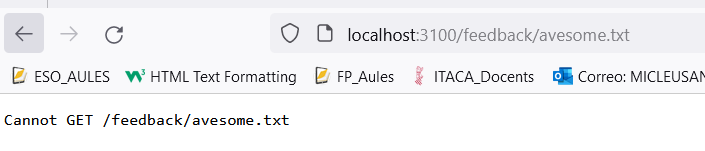
Ahora vamos a buscar a ver si persisten los datos en el contenedor, recordamos que lo hemos eliminado.

Paso 8. Una vez que esté detenido, reiniciémoslo, pero sin el –**rm,** o sea ahora al pararlo no se va eliminar

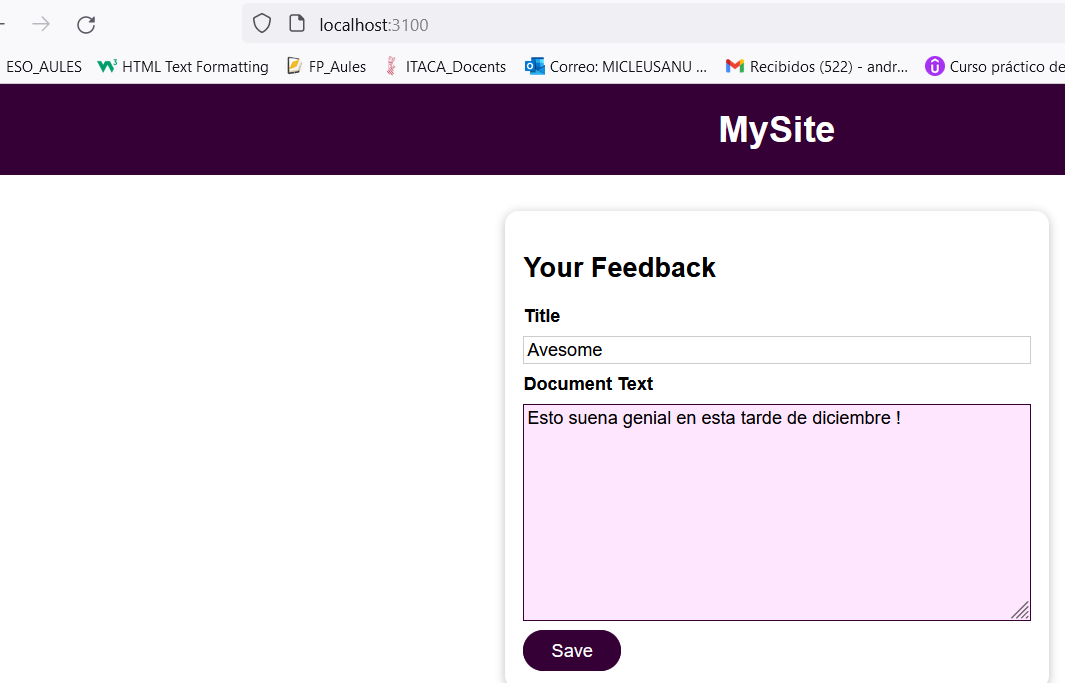
Es decir, con el siguiente comando: *docker run -p 3100:80 -d --name feedback-app feedback-node*

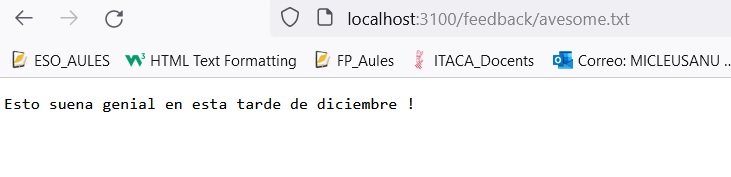


Paso 9. Iniciamos un nuevo contenedor con el mismo nombre, comprobamos en local host, vemos que funciona, pero el archivo *avesome.txt* me va dar un error, no existe este archivo, hace unos minutos funcionaba, pero ahora a dejado de ir.



Paso 10. Volvemos a rescribir en localhost

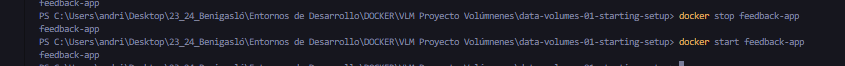




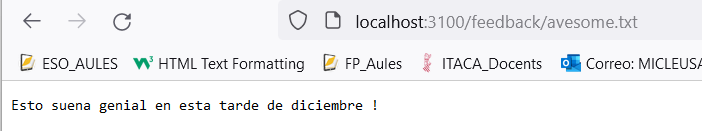
Paso 11. Sí ahora detenemos el contenedor .



Paso 12. Volvemos a iniciar el contenedor otra vez:



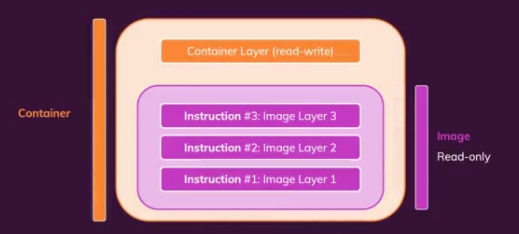
Visitamos el local host veremos que el archivo sigue allí,



La explicación correcta sería: "La explicación es que el contenedor no se ha eliminado."

Vamos a ver en detalle que pasa dentro del contenedor:

Cuando lanzamos el contenedor Docker en función de la imagen, ese contenedor se agrega como una capa de lectura y escritura extra delgada, véase la imagen:



Se agrega como una capa de lectura y escritura extra delgada **“Container Layer** de tipo **read-write”** encima de esta imagen.

Básicamente, tiene acceso a este sistema de archivos de la imagen y puede leer y escribir allí sin manipular la imagen.

Es más, es capaz de leer y escribir allí. Por lo tanto, el contenedor tiene su propia copia de la que podría decirse administrada de una manera muy eficiente. Así que tiene mucho sentido que podamos leer y escribir allí. El problema es que el sistema de archivos está dentro del contenedor.

Entonces, cuando se genera un archivo en el contenedor, no escribe este archivo en la imagen, pero lo escribe en su propia capa de lectura y escritura que se agrega en la parte superior. Por lo tanto, si se quita el contenedor, solo nos queda la imagen que nunca se cambió. Entonces, cuando comenzamos un nuevo contenedor bien, eso comienza con el mismo sistema de archivos básico nuevamente sin los cambios realizados por el contenedor anterior que podría haberse ejecutado en la misma imagen.

¡Esta es la idea central de DOCKER: varios contenedores basados en la misma imagen están totalmente aislados en si!

Imaginar datos que tenemos en contenedores, archivos generados de las aplicaciones, cuentas de usuarios con datos de productos, etc, todo se perderían una vez se eliminan los contenedores

Ahora es lo interesante sabemos que en los contenedores se pueden escribir datos, estos datos se pierden cuando se quita el contenedor,

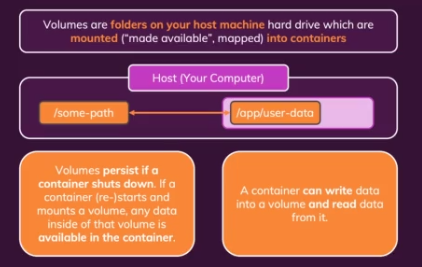
ahora que sabemos esto ,

**¿Cuál es ser la solución?**

Docker tiene una función incorporada llamada **Volúmenes**.

Los volúmenes nos ayudan resolver este problema anterior.

En nuestro caso, en la aplicación de Docker , Los volúmenes son carpetas en su máquina host, por lo que no están en el contenedor, no en la imagen, sino en sino en el disco duro de su máquina host, que están montadas, lo que básicamente significa, disponibles o mapeadas, en contenedores.



Por lo tanto, los volúmenes son carpetas en su máquina host, si se eliminan los contenedores, el volumen no se eliminará, cuando se elimina un contenedor, sobrevive y, por lo tanto, los datos en un volumen sobreviven. Los contenedores pueden leer y escribir datos desde y hacia un volumen. Son los datos que la eliminación de un contenedor. Porque si los datos también se guardan fuera del contenedor, por supuesto sobreviven a la eliminación del contenedor.

Esto lo veremos en una próxima práctica.

Se ha de cambiar el Dockerfile una línea **VOLUME [“ /app/feedback”]** el volumen que quiero guardar es / app / feedback.