2 - PERSISTENCIA Y DOCKER COMPOSE

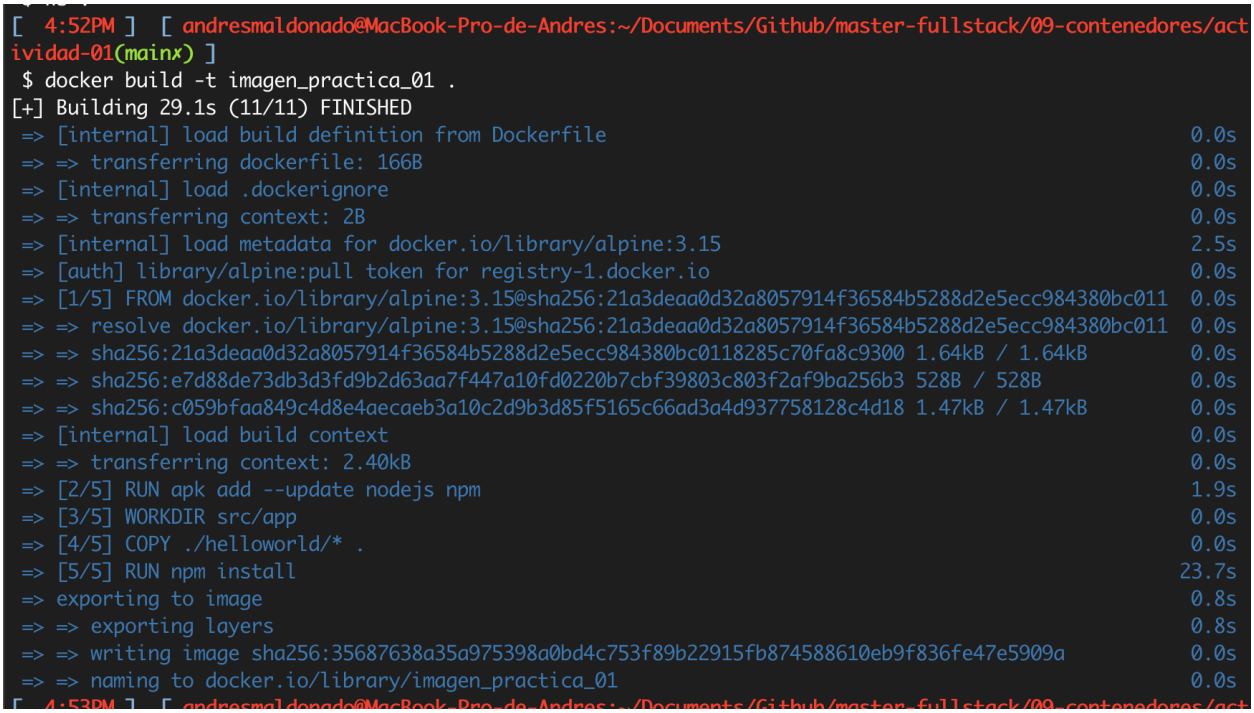
Realizar un bind mount para modificar código dinámicamente

Instalar un contenedor con la imagen de la Práctica 1, usando un bind mount para ser capaz de cambiar el texto que salga por pantalla por un mensaje diciendo el nombre del alumno.

La entrega de este ejercicio debe ser compuesta por las capturas de:

- *Lanzamiendo del docker*
- *Salida de 'docker volume inspect <container>*
- *Captura de imqgen del display de pantalla del servicio web*

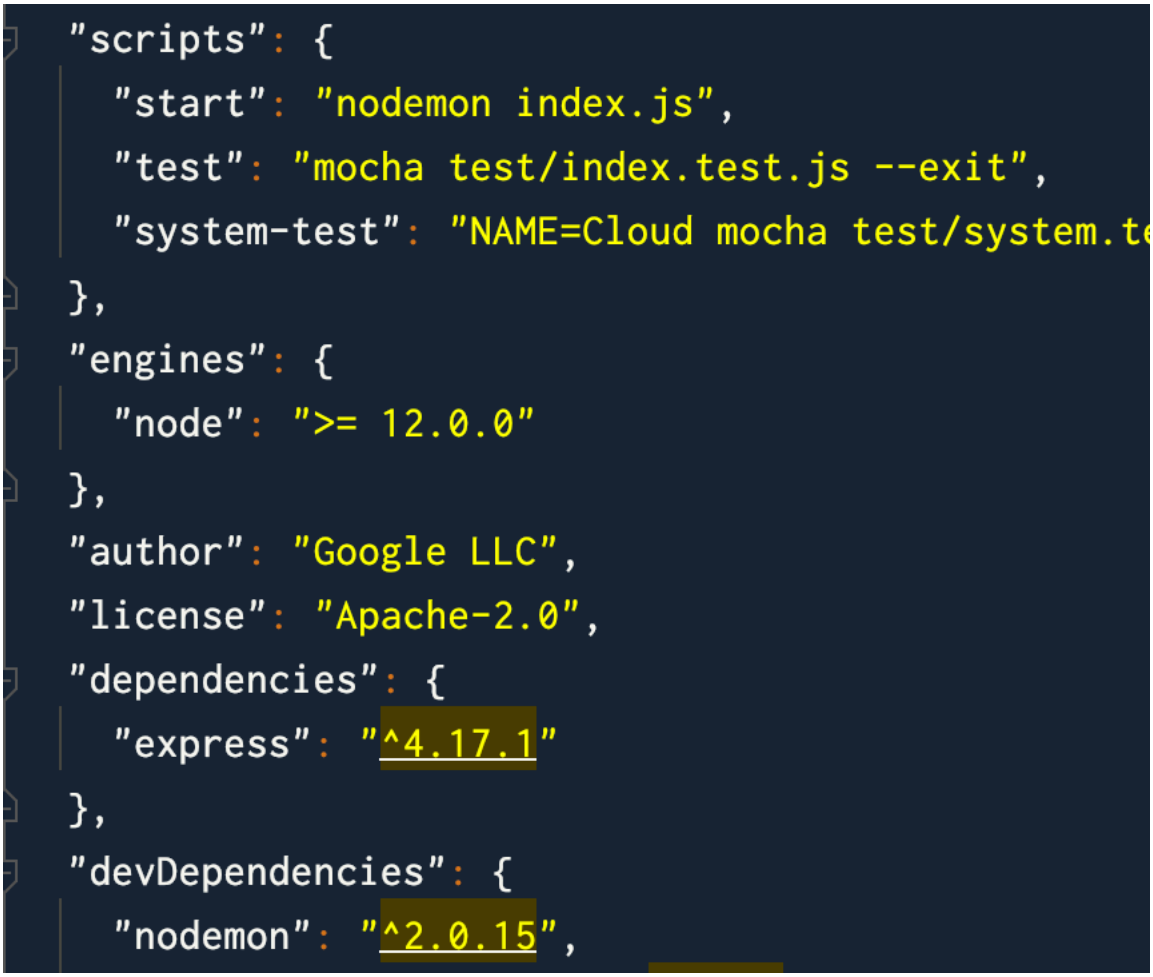
Empiezo por construir la imagen de la práctica 1, le he nombrado “imagen_practica_01”



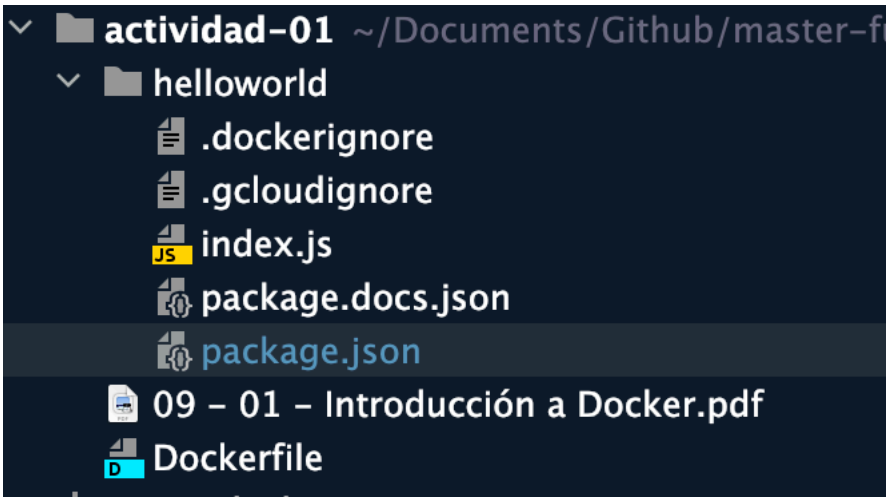
Antes que continuar realizo el siguiente análisis:

- El contenido a cambiar se encuentra en el archivo “index.js” de “Práctica 1”
- Si utilizo **bind mount** a nivel de la **carpeta**, tendré que instalar las dependencias **NPM** en mi máquina ya que no estaré usando las que se encuentran en la imagen (porque viven a nivel de la carpeta), por lo tanto decido utilizar bind mount sobre el **archivo index.js**
- Actualmente, el proyecto corre con node, pero esto impide que se vean cambios en tiempo real si realizo alguna modificación en index.js. En consecuencia, para garantizar el ejercicio, incluyo **nodemon** a mi lista de dependencias en el npm y lo utilizo para iniciar la aplicación en lugar de **node**

Adjunto capturas de las desiciones técnicas anteriormente nombradas en package.json



Estructura de carpeta por la que usaré bind mount en index.js, como se aprecia, los node_modules no se consumirán desde mi local



Me posiciono en proyecto “helloworld”

```
[  5:20PM ]  [ andresmaldonado@MacBook-Pro-de-Andres:~/Documents/Github/master-fullstack/09-contened
es/actividad-01(mainx) ]
 $ cd helloworld
[  5:20PM ]  [ andresmaldonado@MacBook-Pro-de-Andres:~/Documents/Github/master-fullstack/09-contened
es/actividad-01/helloworld(mainx) ]
 $ pwd
/Users/andresmaldonado/Documents/Github/master-fullstack/09-contenedores/actividad-01/helloworld
```

Levanto la imagen configurando como bind mount únicamente el archivo “index.js” que editaré en este ejercicio:

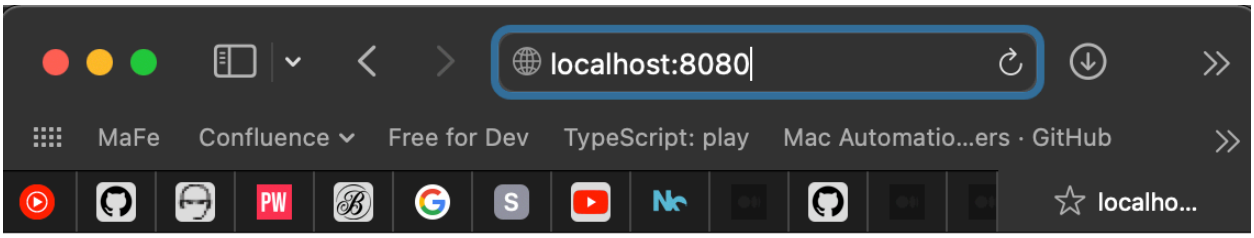
```
[  5:22PM ]  [ andresmaldonado@MacBook-Pro-de-Andres:~/Documents/Github/master-fullstack/09-contenedor
es/actividad-01/helloworld(mainx) ]
 $ docker run -it -p 8080:8080 -v $(pwd)/index.js:/src/app/index.js imagen_practica_01


> helloworld@1.0.0 start
> nodemon index.js

[nodemon] 2.0.15
[nodemon] to restart at any time, enter `rs`
[nodemon] watching path(s): *.*
[nodemon] watching extensions: js,mjs,json
[nodemon] starting `node index.js`
helloworld: listening on port 8080
```

La instancia se encuentra exitosamente corriendo en Docker exponiendo el puerto 8080

```
Last login: Fri Mar  4 16:58:31 on ttys002
[  5:24PM ]  [ andresmaldonado@MacBook-Pro-de-Andres:~/Documents/Github/master-fullstack/09-contenedores/actividad-01/helloworld(m
ainx) ]
 $ docker ps
CONTAINER ID   IMAGE               COMMAND       CREATED              STATUS              PORTS                    NAMES
71319f1f384d   imagen_practica_01   "npm start"   About a minute ago   Up About a minute   0.0.0.0:8080->8080/tcp   great_wilson
```



Hello World!

Cambio el código en index.js con mi nombre como se solicita

```
const express = require('express');
const app = express();

app.get('/', (req, res) => {
  const name = 'Andrés Maldonado';
  res.send(`Hello ${name}!`);
});
```

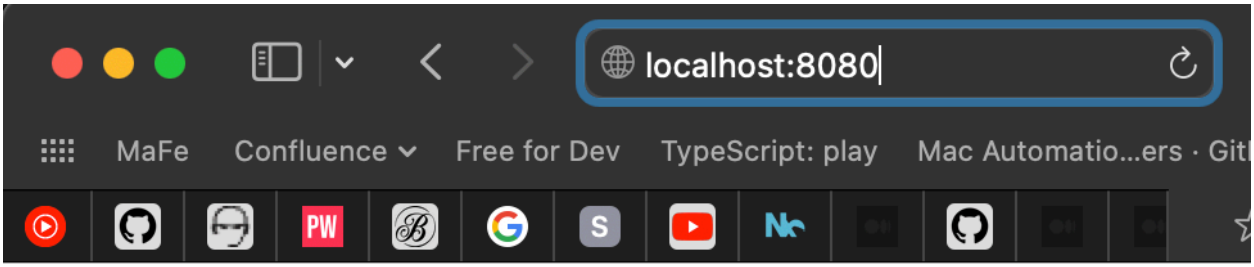
La instancia **NODEMON** que corre actualmente en Docker, escucha el cambio en el archivo y vuelve a lanzar el servidor

```
[  5:22PM ]  [ andresmaldonado@MacBook-Pro-de-And
ainx) ]
 $ docker run -it -p 8080:8080 -v $(pwd)/index.js


> helloworld@1.0.0 start
> nodemon index.js

[nodemon] 2.0.15
[nodemon] to restart at any time, enter `rs`
[nodemon] watching path(s): *.*
[nodemon] watching extensions: js,mjs,json
[nodemon] starting `node index.js`
helloworld: listening on port 8080
[nodemon] restarting due to changes...
[nodemon] starting `node index.js`
helloworld: listening on port 8080
```

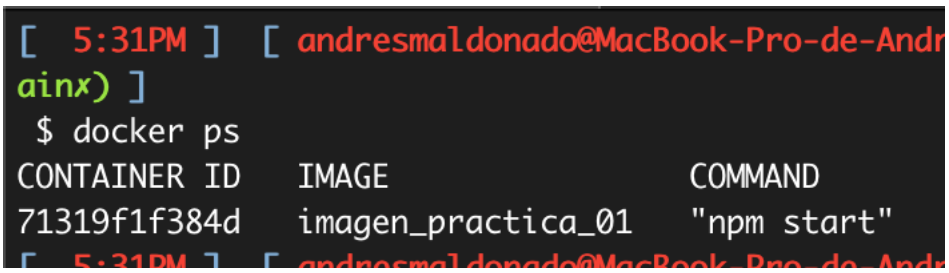
Mi localhost:8080 responde esta vez con mi nombre en pantalla, demostrando así que lee el index.js de mi máquina y no el de Docker



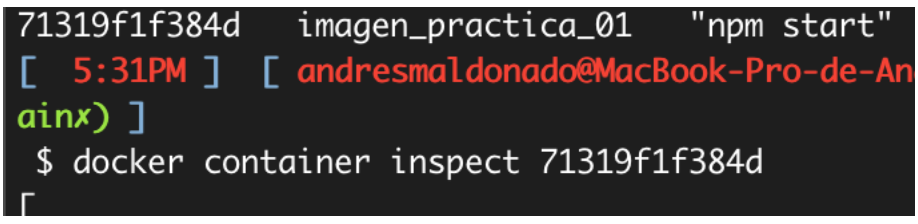
Hello Andrés Maldonado!

Finalmente, se solicita un la *"Salida de 'docker volume inspect <container>"* yo asumo que dado el contexto del ejercicio, se requiere "docker **container** inspect" en lugar de "docker **volume** inspect"

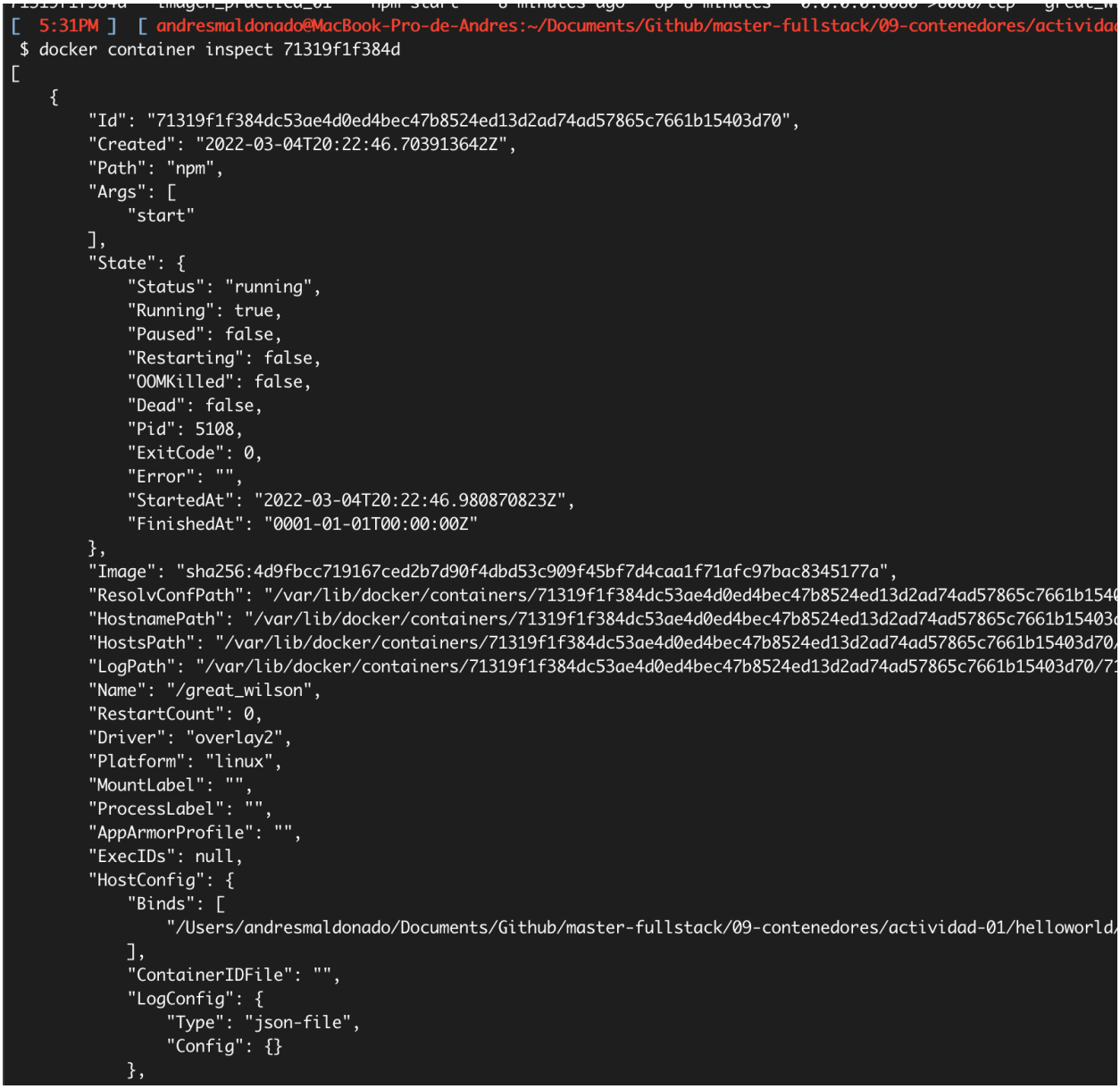
Primero, obtengo el container ID



Procedo a ejecutar el comando "docker container inspect 71319f1f384d"



Esto devuelve un enorme JSON



Considerando el contexto del ejercicio, adjunto captura de “**Mounts**” para demostrar que se trabajó con **bind mount** en **index.js**