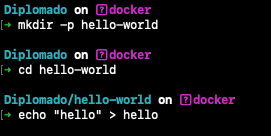
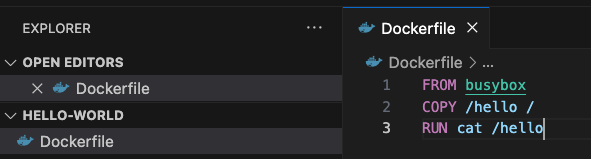
# Taller de Docker

Mi primer Dockerfile

Creamos nuestro build context:

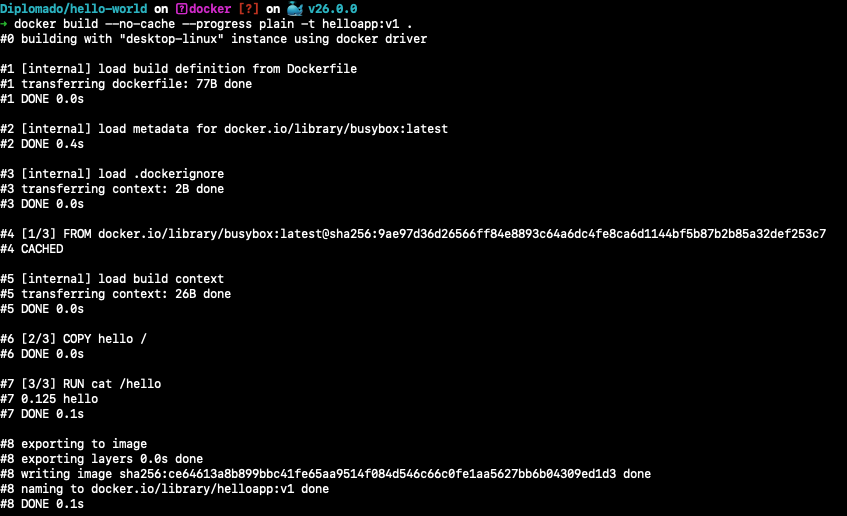


Dentro de este directorio crearemos un archivo llamado Dockerfile con este contenido:

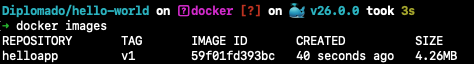


Ahora para crear nuestra imagen usaremos docker build:

docker build -t helloapp:v1 .



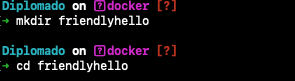
Y podremos ver que una nueva imagen está instalada en nuestro equipo:



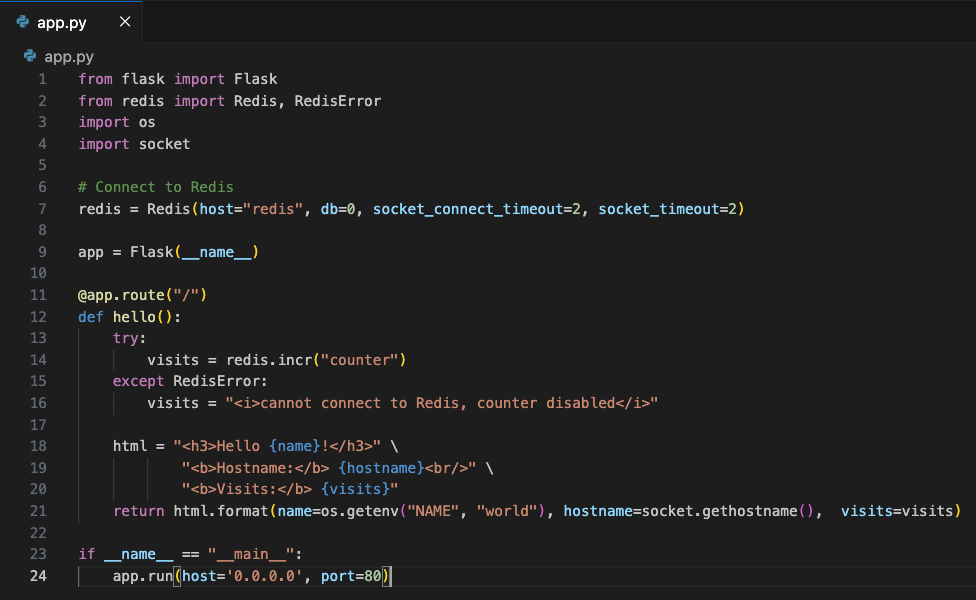
´

Creando aplicaciones en contenedores

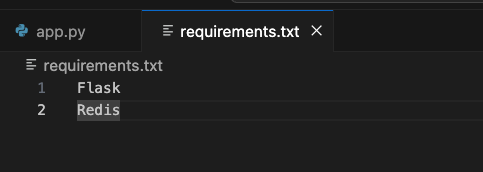
Comenzamos creando un nuevo build context:



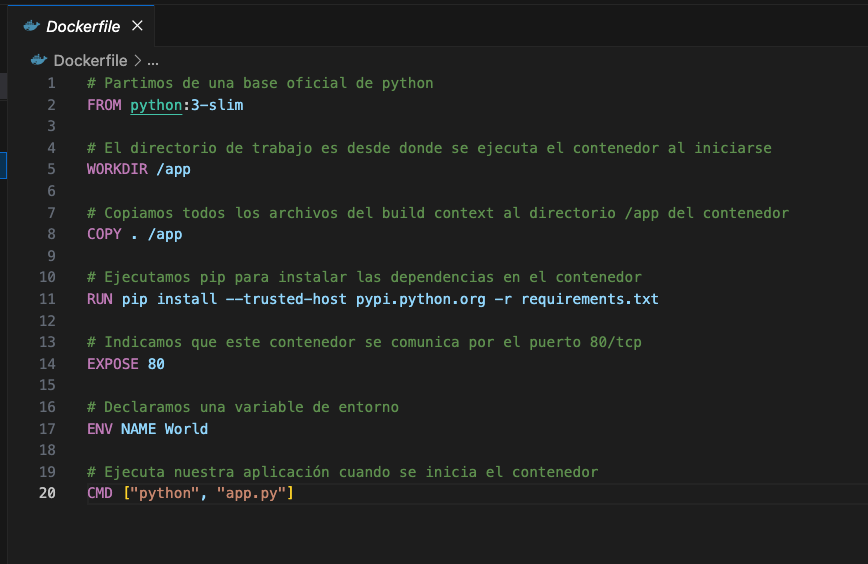
El código de la aplicación es el siguiente, lo guardaremos en un archivo llamado app.py:



Nuestra aplicación tiene una serie de dependencias (librerías de terceros) que guardaremos en el archivo requirements.txt:



Y por último definimos nuestro Dockerfile:

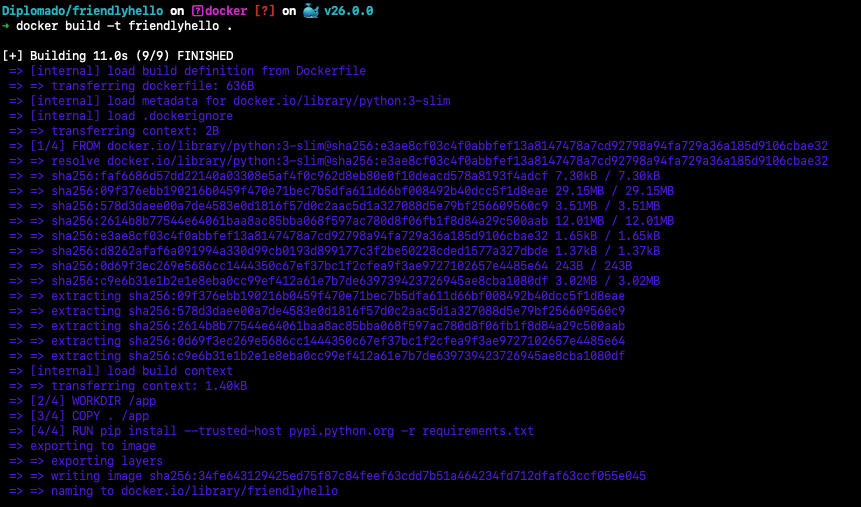


En total debemos tener 3 archivos:

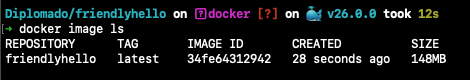


Ahora construimos la imagen de nuestra aplicación:

docker build -t friendlyhello .

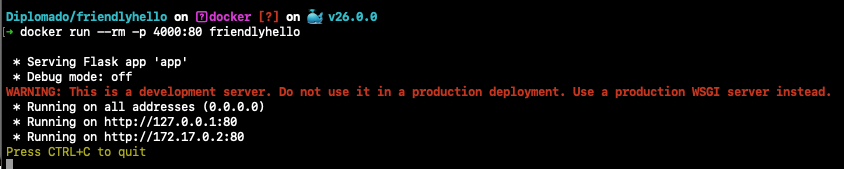


Y comprobamos que está creada:

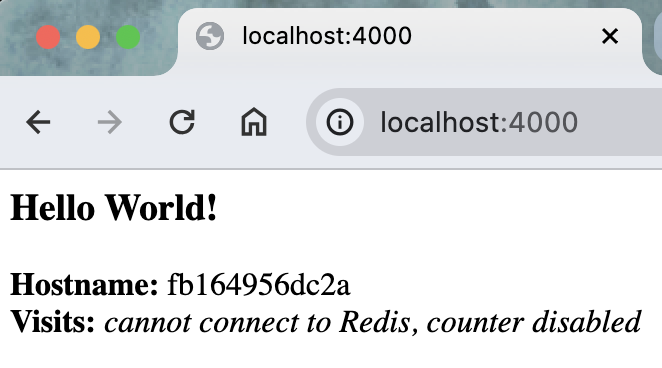


Vamos a arrancar nuestro contenedor y probar la aplicación:

docker run --rm -p 4000:80 friendlyhello

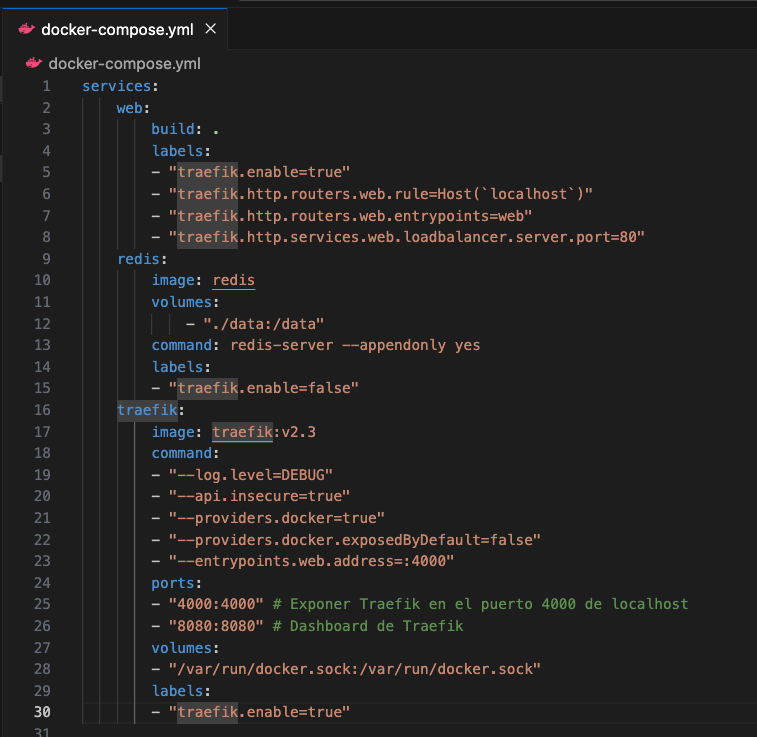


Obtendremos un mensaje como este:



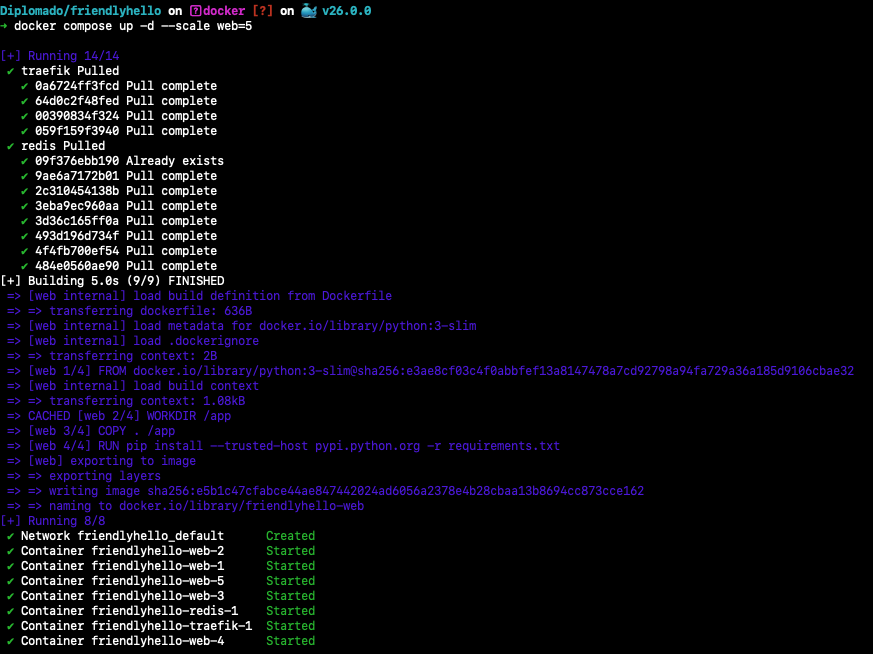
Creando la Aplicación

Vamos a crear el siguiente archivo docker-compose.yaml:

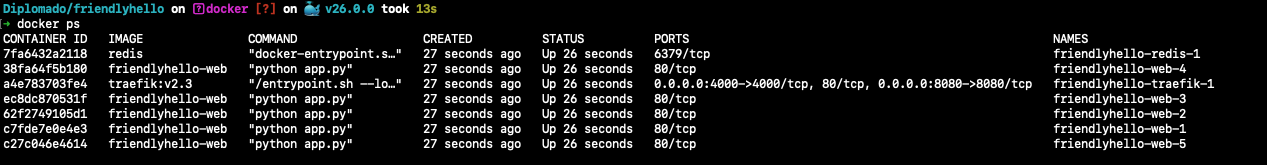


Vamos a arrancar esta nueva aplicación, pero esta vez añadiendo varios servicios web:

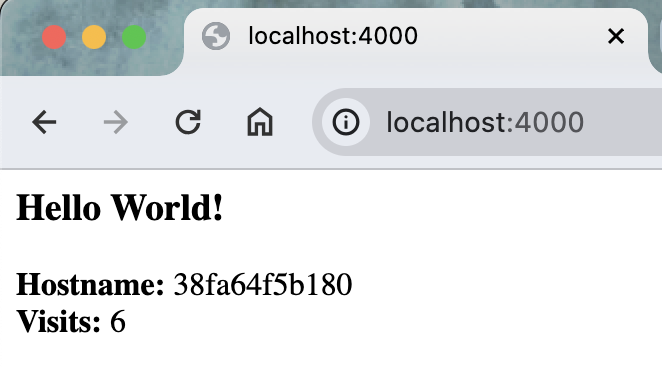
docker compose up -d --scale web=5

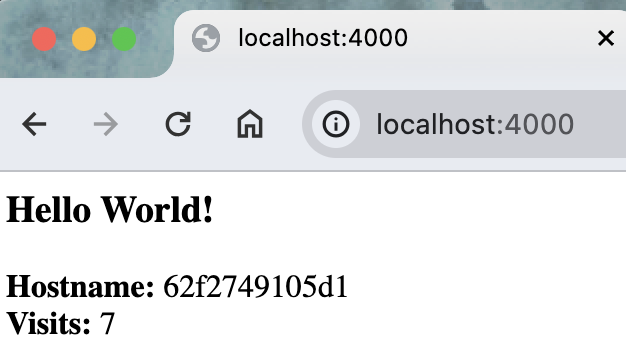


usamos docker ps para ver los contenedores disponibles tendremos:



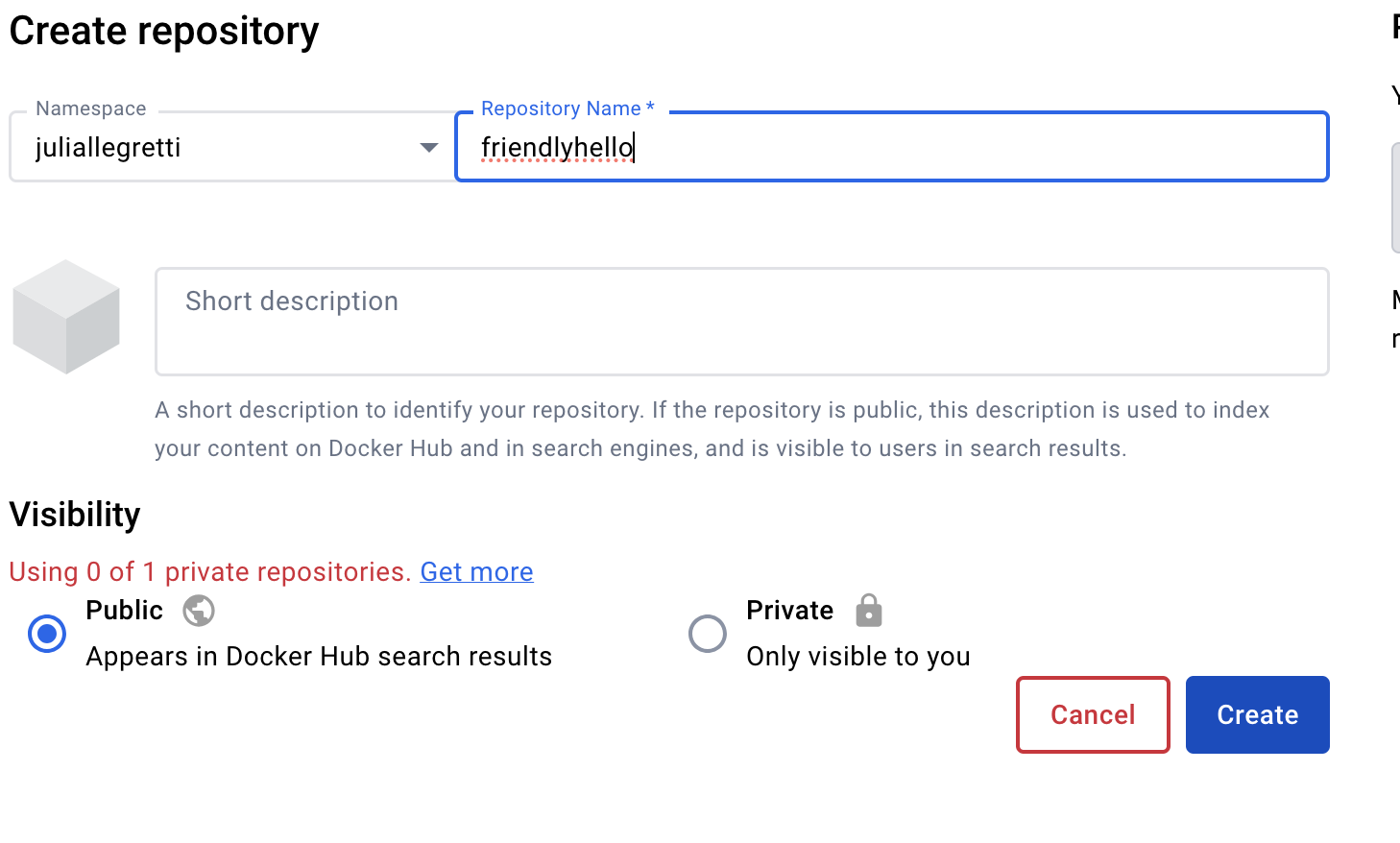
En esta ocasión vamos recargando la página, veremos cómo cambian los hostnames, que a su vez coinciden con los identificadores de los contenedores anteriores.



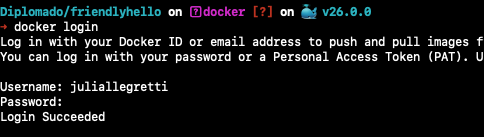


Compartir Imágenes

Creamos un repositorio en DockerHub

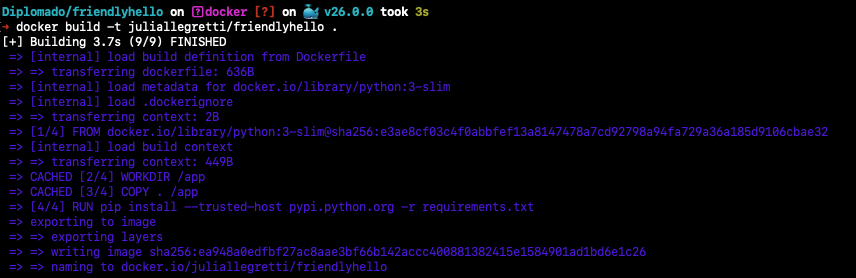


Luego realizamos el docker login

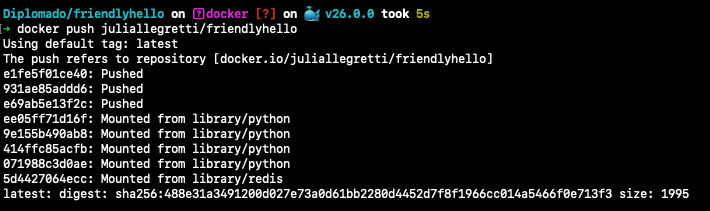


Hacemos build de la imagen con nuestro workspace para que se puedan guardar

docker build -t username/friendlyhello .



Subimos a imagen



Comprobamos

