1- Conceptos de Dockerfiles

Describir las instrucciones

FROM Indica la imagen padre sobre la que se está construyendo *“The FROM instruction initializes a new build stage and sets the Base Image for subsequent instructions”*

RUN Crea una capa del contenedor de escritura sobre la imagen especificada, y al iniciarse utiliza estos comandos *“The RUN instruction will execute any commands in a new layer on top of the current image and commit the results. The resulting committed image will be used for the next step in the Dockerfile.”*

ADD Copia archivos/directorios al sistema de archivos del contenedor *“The ADD instruction copies new files, directories or remote file URLs from <src> and adds them to the filesystem of the image at the path <dest>.”*

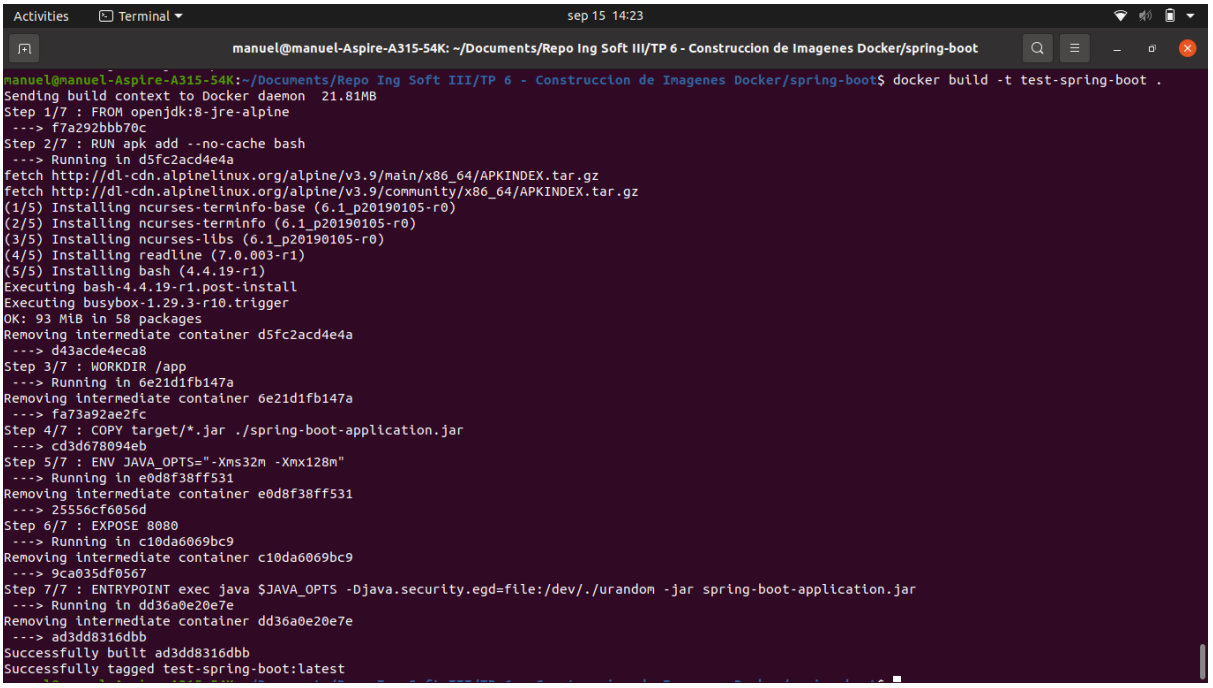
COPY A diferencia de add solamente copia archivos locales a la dirección indicada en el contenedor *“The COPY instruction copies new files or directories from <src> and adds them to the filesystem of the container at the path <dest>.”*

EXPOSE Informa que la imagen creada por el Dockerfile escuchará en el puerto indicado cuando se corra el contenedor *“he EXPOSE instruction informs Docker that the container listens on the specified network ports at runtime. You can specify whether the port listens on TCP or UDP, and the default is TCP if the protocol is not specified. The EXPOSE instruction does not actually publish the port. It functions as a type of documentation between the person who builds the image and the person who runs the container, about which ports are intended to be published“*

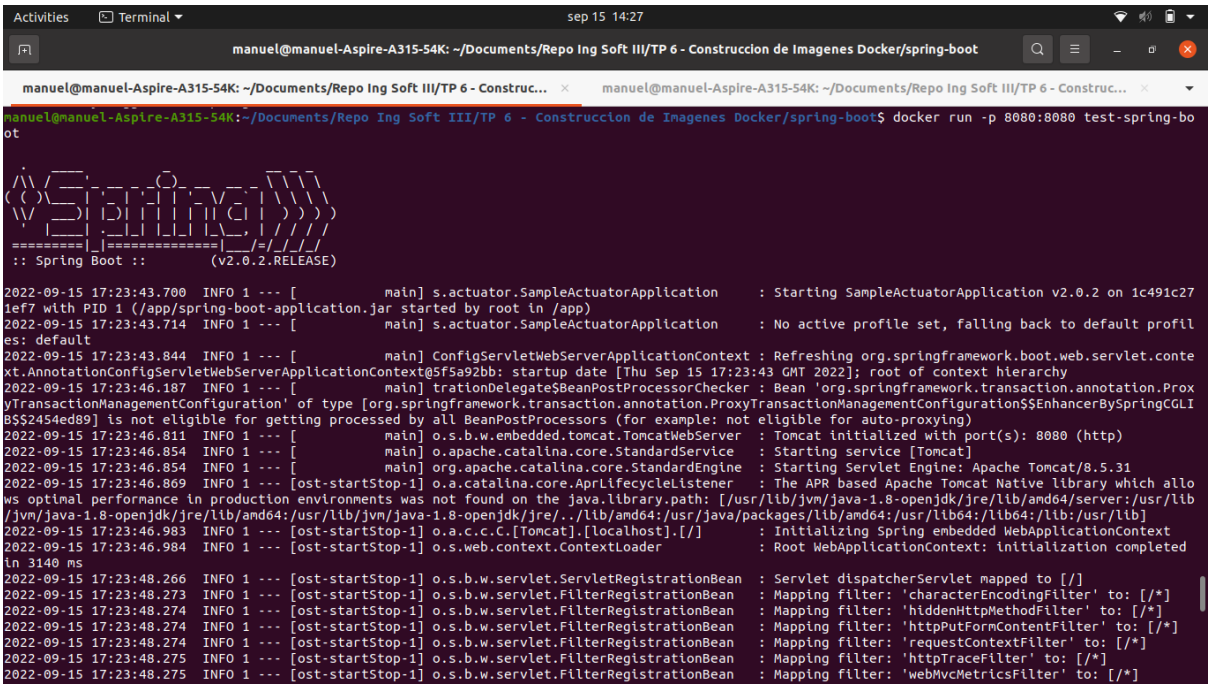
CMD Comando CMD especifica la instrucción que va a ser ejecutada cuando el contenedor se inicialice *“There can only be one CMD instruction in a Dockerfile. If you list more than one CMD then only the last CMD will take effect. The main purpose of a CMD is to provide defaults for an executing container.“*

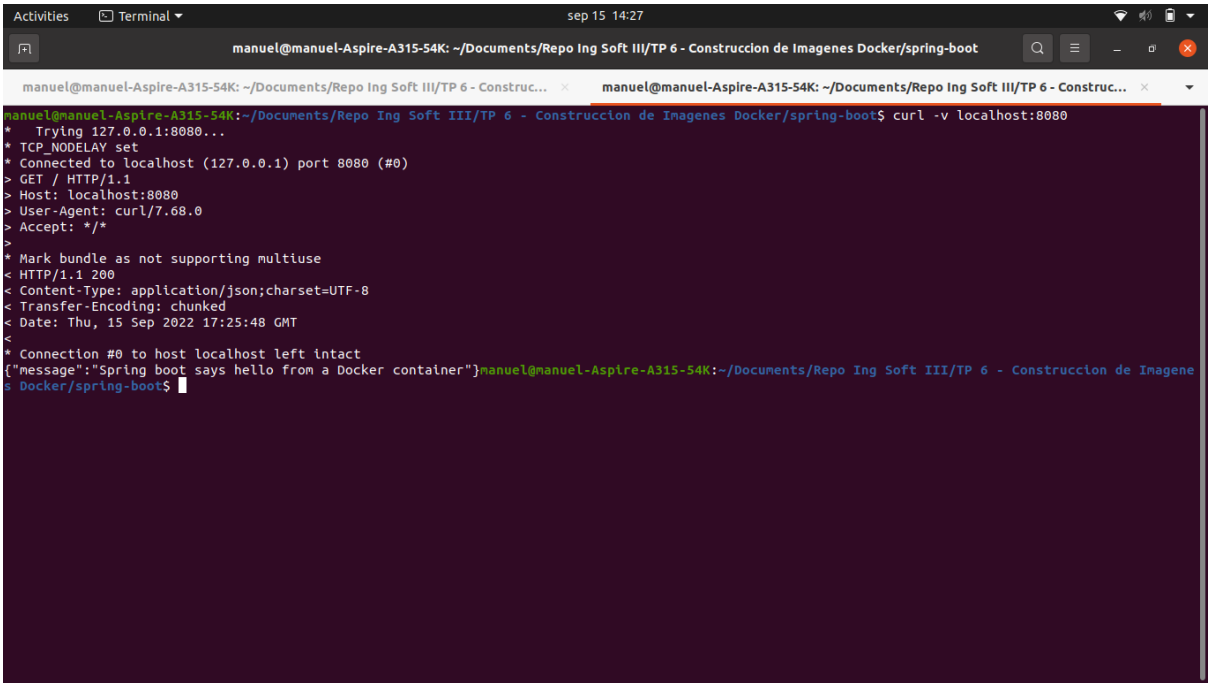
ENTRYPOINT Setea ejecutables que siempre serán corridos cuando se inicialice el contenedor *“An ENTRYPOINT allows you to configure a container that will run as an executable.”*

2- Generar imagen de docker

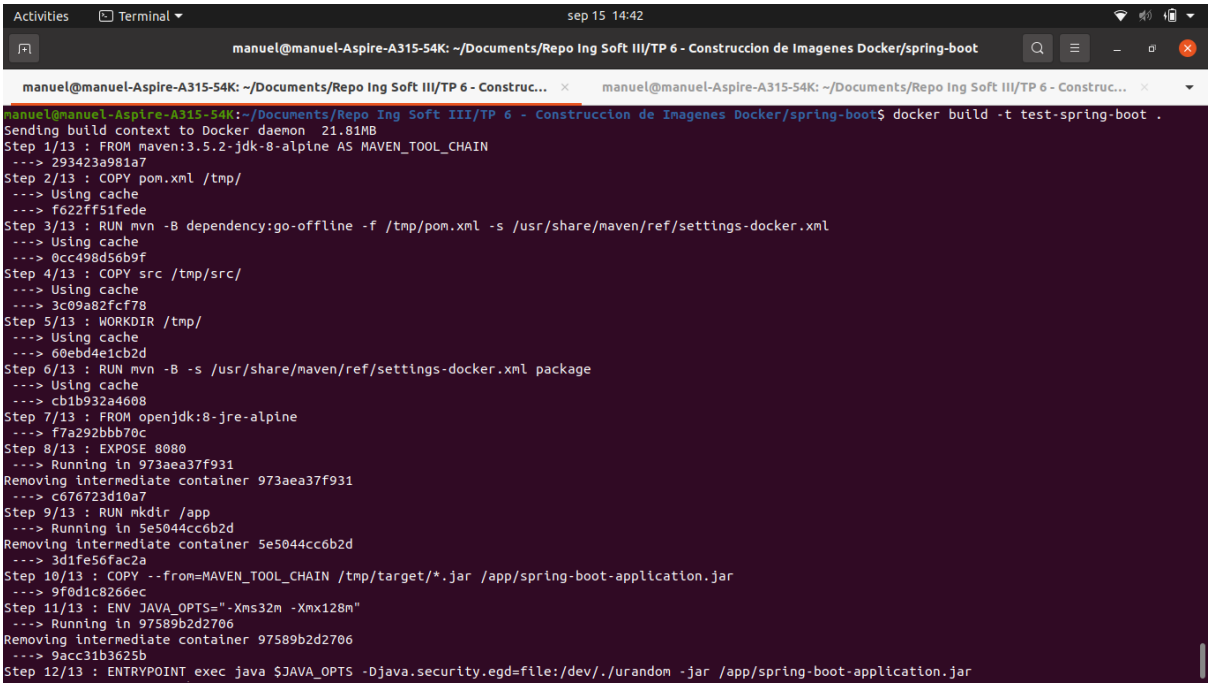


Ejecutar el contenedor




Activities
Terminal
sep 15 14:27
manuel@manuel-Aspire-A315-54K: ~/Documents/Repo Ing Soft III/TP 6 - Construccion de Imagenes Docker/spring-boot
manuel@manuel-Aspire-A315-54K: ~/Documents/Repo Ing Soft III/TP 6 - Construc...
manuel@manuel-Aspire-A315-54K: ~/Documents/Repo Ing Soft III/TP 6 - Construc...
manuel@manuel-Aspire-A315-54K:~/Documents/Repo Ing Soft III/TP 6 - Construccion de Imagenes Docker/spring-boot$ curl -v localhost:8080
*   Trying 127.0.0.1:8080...
* TCP_NODELAY set
* Connected to localhost (127.0.0.1) port 8080 (#0)
> GET / HTTP/1.1
> Host: localhost:8080
> User-Agent: curl/7.68.0
> Accept: */*
>
* Mark bundle as not supporting multiuse
< HTTP/1.1 200
< Content-Type: application/json;charset=UTF-8
< Transfer-Encoding: chunked
< Date: Thu, 15 Sep 2022 17:25:48 GMT
<
* Connection #0 to host localhost left intact
{"message":"Spring boot says hello from a Docker container"}manuel@manuel-Aspire-A315-54K:~/Documents/Repo Ing Soft III/TP 6 - Construccion de Imagene
s Docker/spring-boot$

3- Dockerfiles Multi Etapas



Analizar y explicar el nuevo Dockerfile, incluyendo las nuevas instrucciones.

FROM maven:3.5.2-jdk-8-alpine AS MAVEN_TOOL_CHAIN

Trae una imagen de maven para trabajar sobre ella

COPY pom.xml /tmp/

Copia el archivo pom.xml a /tmp/

RUN mvn -B dependency:go-offline -f /tmp/pom.xml -s /usr/share/maven/ref/settings-docker.xml

Corre el comando mvn -B dependency:go-offline -f /tmp/pom.xml -s /usr/share/maven/ref/settings-docker.xml

COPY src /tmp/src/

Copia el archivo src a /tmp/src

WORKDIR /tmp/

Setea /tmp/ como directorio de trabajo del contenedor "*The WORKDIR command is used to define the working directory of a Docker container at any given time. The*

command is specified in the Dockerfile. Any RUN, CMD, ADD, COPY, or ENTRYPOINT command will be executed in the specified working directory."

RUN mvn -B -s /usr/share/maven/ref/settings-docker.xml package

Corre el comando mvn -B -s /usr/share/maven/ref/settings-docker.xml package

FROM java:8-jre-alpine

Trae una imagen de java para trabajar sobre ella

EXPOSE 8080

Indica que hay que exponer el puerto 8080

RUN mkdir /app

Corre el comando mkdir /app

COPY --from=MAVEN_TOOL_CHAIN /tmp/target/*.jar /app/spring-boot-application.jar

Copia --from=MAVEN_TOOL_CHAIN /tmp/target/*.jar a /app/spring-boot-application.jar

ENV JAVA_OPTS="-Xms32m -Xmx128m"

Setea variable de entorno JAVA_OPTS="-Xms32m -Xmx128m"

ENTRYPOINT exec java $JAVA_OPTS -Djava.security.egd=file:/dev/./urandom -jar /app/spring-boot-application.jar

Cada vez que se corra el contenedor se ejecuta el comando java $JAVA_OPTS -Djava.security.egd=file:/dev/./urandom -jar /app/spring-boot-application.jar

HEALTHCHECK --interval=1m --timeout=3s CMD wget -q -T 3 -s http://localhost:8080/actuator/health/ || exit 1

Realiza periódicamente chequeos de la salud del contenedor

4- Python Flask

¡Explicar que sucedió!

Al correr el contenedor con docker-compose up -d con la opción -d corre el proceso en el background, al llamar el docker compose, también se llama al dockerfile, que mediante las líneas

COPY ./requirements.txt /requirements.txt

RUN pip install -r /requirements.txt

Instala las dependencias mediante el archivo requirements.txt

¿Para qué está la key build.context en el docker-compose.yml?

El context define el path/directorio que contiene el dockerfile

5- Imagen para aplicación web en Nodejs

Archivo Dockerfile creado:

```
FROM node:16.17.0-alpine
RUN mkdir -p /usr/src/app
WORKDIR /usr/src/app
RUN npx create-react-app my-app
RUN cd my-app
WORKDIR /usr/src/app/my-app
RUN npm install
CMD ["npm","start"]
EXPOSE 3000
```

Comando para la creación de la imagen

docker build -t node-test .
docker run -p 3000:3000 node-test

6- Publicar la imagen en Docker Hub.

```
manuel@manuel-Aspire-A315-54K:~$ docker login
Login with your Docker ID to push and pull images from Docker Hub. If you don't have a Docker ID, head over to https://hub.docker.com to create one.
Username: manuelbobadilla
Password:
WARNING! Your password will be stored unencrypted in /home/manuel/.docker/config.json.
Configure a credential helper to remove this warning. See
https://docs.docker.com/engine/reference/commandline/login/#credentials-store

Login Succeeded
manuel@manuel-Aspire-A315-54K:~$ docker tag node-test manuelbobadilla/node-test:latest
manuel@manuel-Aspire-A315-54K:~$ docker push manuelbobadilla/node-test:latest
The push refers to repository [docker.io/manuelbobadilla/node-test]
e686a063df71: Pushed
6bc31c2eb673: Pushed
9b4e76917ace: Pushed
f1ed0bba6314: Mounted from library/node
2808ff9120f2: Mounted from library/node
cb6eda6d73f0: Mounted from library/node
994393dc58e7: Mounted from library/node
latest: digest: sha256:4b353a65688372d80d6a4edf578009e8dc9d79b0b30ad1fac0c1dd89e5e92736 size: 1788
```