**Consigna:**

**Adjunto hay un archivo .jar que tiene la aplicación PasswordApi.**

**Esa aplicación es una aplicación Java que se ejecuta con el comando: java -jar passwordapi.jar. Eso levanta una webapi en el puerto 8080 (ver captura de pantalla).**

**Lo que se pide es generar una imagen Docker que corra esa aplicación. Más concretamente lo que hay que hacer es:**

**1. Escribir un dockerfile**

**2. Generar la imagen a partir del dockerfile ejecutando un docker build**

**3. Publicar la imagen en Dockerhub**

**Resolver el ejercicio en un directorio "ejercicio\_04" del repositorio personal.**

**Mencionar en el readme el link a la imagen publicada en dockerhub**

**Entregar el link al repositorio github.**

**RESOLUCIÓN:**

1. **Se crea el dockerfile**

FROM openjdk:8-alpine

COPY passwordapi.jar /usr/share/app.jar

ENTRYPOINT ["java", "-jar", "/usr/share/app.jar"]

FROM openjdk:8-alpine

Indica que queremos usar la versión JDK-8 sobre un [sistema mínimo como Alpine Linux](https://alpinelinux.org/) para asegurarnos de que el tamaño del archivo generado es lo más pequeño posible.

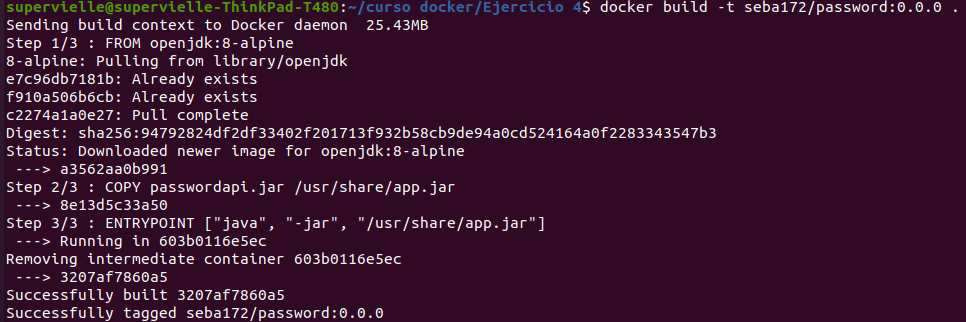
COPY passwordapi.jar /usr/share/app.jar

Indicamos la ubicación de nuestro jar de aplicación y a donde lo vamos a copiar en el contenedor

ENTRYPOINT ["java", "-jar", "/usr/share/app.jar"]

Se indica el comando de arranque que debe utilizar el contenedor una vez inicializado.

1. **Se genera la imagen a partir del dockerfile del punto 1**

****

**Explicación:**

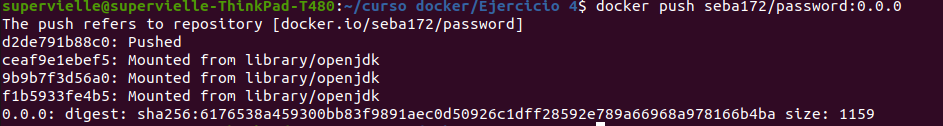
**docker build -t my\_docker\_hub\_username/my\_image\_name:my\_image\_version .**

Importante el punto (.) al final, este indica el directorio actual, en donde está el fichero Dockerfile. el argumento -t user/image:tag permite nombrar la imagen.

**Se verifica que la imagen se generó:**

****

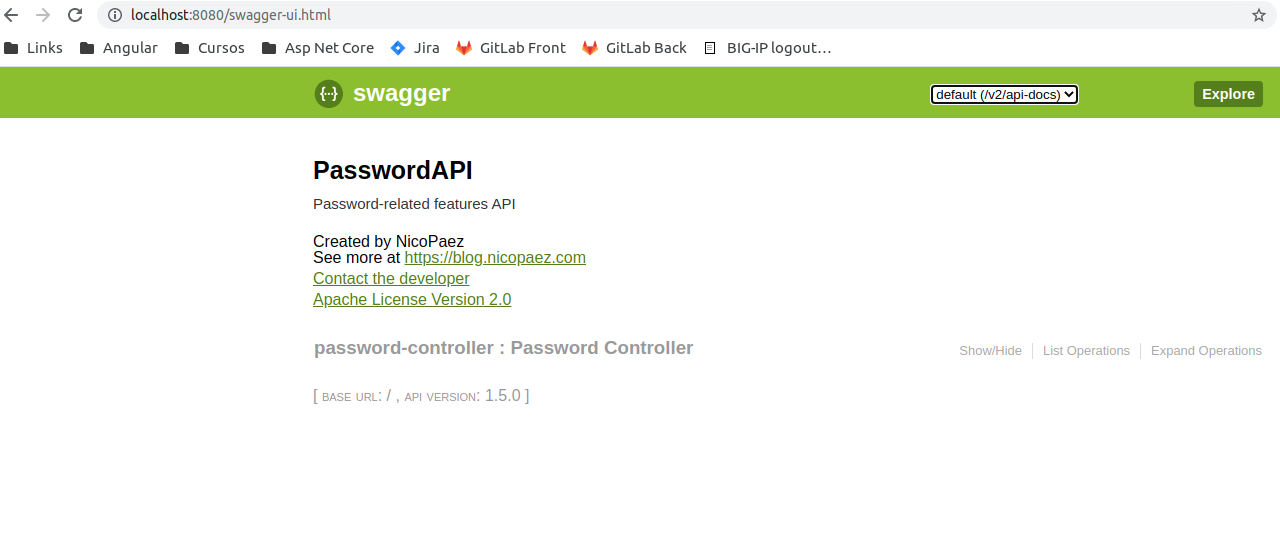
1. **Se sube al repositorio de dockerhub la imagen generada**

****

1. **Se corre la imagen mapeando en los puertos 8080**

****

**Se verifica que está funcionando**

****