RSE 2021-2022 / Alejandro Albert Casañ

Memoria práctica 8

1.- Describe linea por linea el contenido del fichero Dockerfile

```
# Elige la imagen base

FROM alpine

# Instala python y pip

RUN apk add --update py3-pip

# Actualiza pip

RUN pip install --upgrade pip

# Instala el módulo Flask para poder ejecutar la aplicación

RUN pip install -U Flask

# Copiar los ficheros necesarios para el funcionamiento de la app

COPY app.py /usr/src/app/

COPY templates/index.html /usr/src/app/templates/

# Se especifica el puerto que debe tener abierto el contenedor

EXPOSE 5000

# Iniciar la aplicación

CMD ["python3", "/usr/src/app/app.py"]
```

2.- Explica cuál es el uso del parámetro -p 8888:5000

Este parámetro sirve para hacer una redirección de puertos. En este caso estamos haciendo que el puerto 8888 de la máquina que está ejecutando docker se corresponda con el puerto 5000 del contenedor.

3.- Apunta el nombre completo de tu imagen en el documento a entregar.

El nombre de mi imagen es alejandalb98/myfirstapp

4.- Tienes que crear el Dockerfile necesario para implementar un subscriber al broker broker.hivemq.com para los mensajes con topic test/#. Como imagen base puedes utilizar alpine y como codigo python base para el subscriber puedes reutilizar el sisub.py de las sesiones de IoT.

Modificamos el archivo sisub.py para que se conecte al broker y topic indicados.

dockerfile | sisub.py

sisub.py > ...

```
4
5   import paho.mqtt.client as mqtt
6
7   THE_BROKER = "broker.hivemq.com"
8   THE_TOPIC = "test/#"
9   CLIENT_ID = ""
10
11  # The callback for when the client receives a CONNACK response from the server.
12  def on_connect(client, userdata, flags, rc):
13      print("Connected to ", client._host, "port: ", client._port)
14      print("Flags: ", flags, "returned code: ", rc)
15
16      client.subscribe(THE_TOPIC, qos=0)
17
```

Comprobamos que funciona

```
vmrse@vmrse-VirtualBox:~/Escritorio/rse/pract8/mqttconsumer$ python3 sisub.py
Connected to  broker.hivemq.com port:  1883
Flags:  {'session present': 0} returned code:  0
sisub: msg received with topic: test/gateways/sc1/devices/bridge/active and payl
oad: b'{"timestamp":"1639150467177","content":"1"}'
sisub: msg received with topic: test/mode and payload: b'06507'
sisub: msg received with topic: test/gateways/sc2/devices/sensor4/info and paylo
ad: b'{"timestamp":"1639146834575","content":"Device with ID \'sensor4\' is read
y."}'
sisub: msg received with topic: test/lab and payload: b'hola'
sisub: msg received with topic: test/gateways/sc1/devices/spotter/info and paylo
ad: b'{"timestamp":"1639150128942","content":"Device with ID \'spotter\' is read
y."}'
```

Creamos el dockerfile

dockerfile | sisub.py

dockerfile

```
1   # our base image
2   FROM alpine
3
4   # Install python and pip
5   RUN apk add --update py3-pip
6
7   # upgrade pip
8   RUN pip install --upgrade pip
9
10  # install Python modules needed by the Python app
11  RUN pip3 install paho-mqtt
12
13  # copy files required for the app to run
14  COPY sisub.py /usr/src/app/
15
16  # tell the port number the container should expose
17  EXPOSE 5000
18
19  # run the application
20  CMD ["python3", "/usr/src/app/sisub.py"]
```

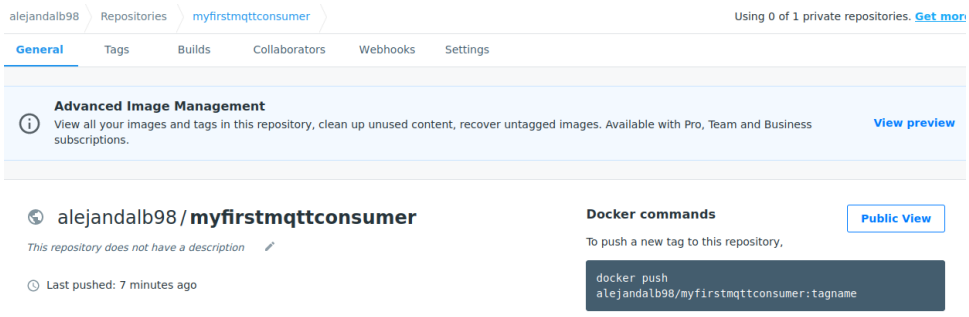
Creamos la imagen a partir del dockerfile con docker build -t alejandalb98/myfirstmqttconsumer . y comprobamos que funciona

```
vmrse@vmrse-VirtualBox:~/Escritorio/rse/pract8/mqttconsumer$ sudo docker run -p
8888:5000 --name myfristmqttconsumer alejandalb98/myfirstmqttconsumer
Connected to  broker.hivemq.com port:  1883
Flags:  {'session present': 0} returned code:  0
sisub: msg received with topic: test/gateways/sc1/devices/bridge/active and payl
oad: b'{"timestamp":"1639150467177","content":"1"}'
sisub: msg received with topic: test/mode and payload: b'06507'
sisub: msg received with topic: test/gateways/sc2/devices/sensor4/info and paylo
ad: b'{"timestamp":"1639146834575","content":"Device with ID \'sensor4\' is read
y."}'
sisub: msg received with topic: test/lab and payload: b'hola'
sisub: msg received with topic: test/gateways/sc1/devices/spotter/info and paylo
ad: b'{"timestamp":"1639150128942","content":"Device with ID \'spotter\' is read
y."}'
sisub: msg received with topic: test/topic12/1 and payload: b'[{"temp2":23.8}]'
sisub: msg received with topic: test/gateways/sc2/devices/waste/connected and pa
```

Ahora solo queda subirlo a docker hub con el nombre de repositorio alejandalb98/myfirstmqttconsumer

```
vmrse@vmrse-VirtualBox:~/Escritorio/rse/pract8/mqttconsumer$ sudo docker push al
ejandalb98/myfirstmqttconsumer
Using default tag: latest
The push refers to repository [docker.io/alejandalb98/myfirstmqttconsumer]
aca985df34d2: Pushed
2b9321a1bf83: Pushed
3abbec5f6dcc: Mounted from alejandalb98/myfirstapp
21f54fe48881: Mounted from alejandalb98/myfirstapp
8d3ac3489996: Mounted from alejandalb98/myfirstapp
latest: digest: sha256:13390e09110bf524f94cc66255015fcbcf213e7f76d614c97b412cc59
87a9075 size: 1368
```

Y con esto ya habríamos acabado.



Código de la práctica

- Fichero dockerfile
- sisub.py
- Imagen docker hub: **alejandalb98/myfirstmqttconsumer**

Se adjuntarán los ficheros en el envío de la tarea, pero también podrán obtenerse en [mi repositorio de github]