Ejercicio 2

Vilchis Domínguez Miguel Alonso

1. Contexto

1.1. Qué es docker

Es una plataforma de virtualización, además de un conjunto de comandos para establecer *workflows* de trabajo que permitan crear, instalar, compartir, etc, diversas aplicaciones.

1.2. Componentes

- *Imagenes*: Una imagen es una plantilla no modificable, que contiene todas las dependencias del proyecto. El comando para ver cuales son las imagenes instaladas: $ *docker images*
- *Dockerfile*: Existen dos maneras de obtener una imagen, la primera de ellas es descargandola directamente de algún depósito, la segunda es definiendo el archivo de texto Dockerfile (Este archivo debe tener ese nombre), en donde se definen los comandos que serán ejecutados cuando la imagen se cree. Nosotros trabajaremos con la segunda opción. Ejemplo de archivo docker:

```
################################################
#
# Seleccionamos la imagen de linux con la que trabajaremos
#
################################################
FROM ubuntu:latest

################################################
#
#El metodo run ejecuta el comando siguiente
#Instalamos las dependencias necesarias
#
################################################
RUN apt-get update && \
apt-get install -y --no-install-recommends \
python\
vim \
git\
build-essential\
python-qt4\
python-pyaudio\
python-opencv


########################################
#
#Creamos el directorio de trabajo
```

```
#
##########################################
RUN mkdir Redes-17


##################################################
#
#Agregamos todos los archivos que se encuentran
# en la carpeta que contiene este archivo
#Dockerfile al directorio de trabajo
#
##################################################
ADD . Redes-17

##################################################
#
#Nombramos el directorio creado como directorio de trabajo
#
##################################################
WORKDIR Redes-17
```

- *Contenedor*: Es creado a partir de una imagen, es a través de estos que se lleva a cabo la interacción. Cada contenedor es un ambiente aislado, aunque se pueden ligar varios contenedores. Y cada contenedor cuenta con un sistema de archivo raíz, procesos, memoria, dispositivos, puertos de red. El comando para ver cuales son los contenedores existentes es: $ *docker ps -a*
  Nota: los contenedores existirán mientras haya un proceso corriendo en ellos, ya sea que ustedes lo definan con la instrucción CMD en el Dockerfile, o que ingresen a la terminal del contenedor. Una vez que termine el proceso o que salgan de la terminal el contenedor se cerrará.

1.3. Comando útiles

- *Crear una imagen desde un archivo docker* : Al situarnos en la carpeta que contiene el archivo Dockerfile y ejecutar el siguiente comando : $ *docker build -t nombre_de_imagen* . se creará la imagen basada en el archivo Dockerfile.

- *Crear un contenedor de una imagen*: Después de haber construido la imagen docker, para crear el contenedor ejecutamos el siguiente comando: $ *docker run nombre_de_imagen*

  La funcionalidad de este comando puede extenderse, pueden agregar variables de entorno con la bandera $-e$, pueden ligar carpetas o archivos de su computadora host al contenedor con la bandera $-v$, pueden compartir dispositivos con la bandera $--$ *device*.

- *Ingresar a un contenedor ya creado*: Para ingresar a un contenedor que se encuentra en ejecución, se tiene que obtener el id del contenedor, con el comando *docker ps*. Y con el id del contenedor al que queremos ingresar ejecutamos $*docker exec -it id_del_contenedor bash*. Con esto obtendremos la terminal del contenedor que se encuentra en ejecución.

2. Ejercicio

2.1. Especificaciones

- Deben entregar 2 archivos Dockerfile, que construya las imagenes necesarias para trabajar con el directorio de ubicación, y el otro trabajará con los clientes.

- No debe haber interacción con la terminal del contenedor, el usuario solo ejecutará *docker run* con los parametros que establezcan y debe de mostrarse la ventana de login. (Revisar el comando CMD del dockerfile).
- El readme debe contener la manera en la que ejecutan el contenedor.
- La imagen debe de funcionar de manera local, conectando a más de 3 clientes. (Note que en la mayoría de los casos solo un cliente podrá tener acceso a la camara web y al micrófono, los otros deben poder escuchar y reproducir lo que el primer cliente les manda).

3. Entrega

Fecha de entrega: 17 de Octubre del 2016

La entrega es por parejas, el ejercicio debe estar almacenado en un depósito de github llamado Redes2017, en la rama Ejercicio2, el readme de la rama debe contener el nombre de los 2 integrantes del equipo.

Deben mandar el link del depósito con el asunto *[Redes-17]Ejercicio2* al siguiente correo:

mvilchis@ciencias.unam.mx

Nota: Basta con que un integrante del equipo suba su código al depósito.
Nota: A las 11:59 del díal de la entrega se descargará el contenido de los depósitos, para evaluación, asegurense que se encuentre su versión final para esa hora.