Trabajo práctico Docker

1. ¿Qué es Docker?
2. ¿Qué es un contenedor? Según la definición oficial del sitio web docker.com
3. ¿Qué es una imagen de Docker?
4. ¿Qué es DockerHub?
5. ¿Qué es un Dockerfile?
6. ¿En qué se diferencia Docker de una máquina virtual?
7. ¿Qué es docker compose?
8. ¿Qué comando se usa para verificar la versión de docker?
9. ¿Qué comando se usa para iniciar un docker?
10. ¿Por que un desarrollador debería aprender docker?
11. ¿Qué comando se usa para detener un contenedor en ejecución?
12. ¿Qué comando se usa para ver todos los contenedores en ejecución?
13. ¿Cuál es la dirección IP predeterminada del host Docker?
14. ¿Qué comando se usa para eliminar Docker?
15. Docker es un sistema operativo (o runtime) para contenedores. El motor de Docker se instala en cada servidor en el que desee ejecutar contenedores y proporciona un conjunto sencillo de comandos que puede utilizar para crear, iniciar o detener contenedores.
16. “Empaquete aplicaciones como imágenes de contenedores portátiles para ejecutarlas en cualquier entorno de manera uniforme, desde Kubernetes locales hasta AWS ECS, Azure ACI, Google GKE y más
17. Una imagen en Docker es un archivo o file que se encuentra compuesto de diversas capas y que se utiliza con el objetivo de ejecutar un código dentro de un contenedor de Docker. Estas imágenes contienen todo el sistema de ficheros inicial en los que se va a basar el container para su funcionamiento, así como su punto de entrada o entrypoint.
18. Docker hub es un repositorio público en la nube, similar a Github, para distribuir los contenidos. Está mantenido por la propia Docker y hay multitud de imágenes, de carácter gratuito, que se pueden descargar y así no tener que hacer el trabajo desde cero al poder aprovechar “plantillas”.
19. Un Dockerfile es un archivo de texto plano que contiene una serie de instrucciones necesarias para crear una imagen que, posteriormente, se convertirá en una sola aplicación utilizada para un determinado propósito. Similar a lo explicado anteriormente, y la base del funcionamiento de Docker es mediante Dockerfiles.
20. En Docker, los contenedores que se ejecutan comparten el núcleo del sistema operativo host. Una máquina virtual, por otro lado, no se basa en la tecnología de contenedores. Se componen de espacio de usuario más espacio de kernel de un sistema operativo. En máquinas virtuales, el hardware del servidor está virtualizado.
21. Docker Compose es una herramienta para definir y ejecutar aplicaciones de Docker de varios contenedores. En Compose, se usa un archivo YAML para configurar los servicios de la aplicación. Después, con un solo comando, se crean y se inician todos los servicios de la configuración.
22. Como vemos, soporta la mayoría de los sistemas Linux, Mac e incluso Windows. Si queremos obtener más información sobre la instalación, podemos usar el comando docker version o bien docker info.
23. Para iniciar una imagen de Docker, utiliza el comando docker run con la siguiente estructura base: $ docker run [OPTIONS] IMAGE [:TAG|@DIGEST] [CMD] [ARG...].
24. Gracias a Docker siempre tendremos el mismo entorno y evitamos tener inconsistencias en desarrollo, producción, testing, etc. Todas las versiones y configuraciones del software con el que hayamos creado nuestro entorno de desarrollo se replicarán en todos los entornos en los que utilicemos la imagen de Docker.
25. **docker stop** El comando se usa para detener un contenedor en ejecución. Aquí tenemos que poner el nombre o ID del contenedor junto con esto. En caso de éxito, devolvería el nombre o ID de la ventana acoplable. Esto devolverá el CONTAINER ID que puede utilizar para detener el contenedor.
26. La orden para ver el listado de contenedores del sistema es **docker container ls** o la forma abreviada docker ps
27. Al instalar Docker, se configura automáticamente para usar la red 172.17.0. xx, el propio servidor docker es **172.17.0.1** y cada contenedor corriendo adquiere un IP posterior al último
28. Si al crear un contenedor sabe que no querrá conservarlo una vez que lo termine, puede ejecutar **docker** run --rm para eliminarlo automáticamente después de cerrarlo. Ejecutar y **eliminar**: **docker** run --rm image\_name.