1. ¿Qué es Docker?

Docker es una herramienta que nos sirve para desarrollar, desplegar y correr aplicaciones con contenedores

1. ¿Cuál es la diferencia entre una VM y un contenedor?

Una máquina virtual es una simulación de otro dispositivo o computador donde esta reserva o divide la memoria de la computadora host o para su uso, y un contenedor es más ligero ya que este comparte los recursos con la computadora host utilizando solamente la memoria que necesita.

1. ¿Cuáles son las diferentes funcionalidades y aplicaciones de usar Docker?

Da la posibilidad de montar cualquier tipo de aplicación con la facilidad de portabilidad haciendo que la aplicación funcione correctamente de un Host A a un Host B.

1. ¿Cuál es el ciclo de vida de un contenedor?

Los contenedores son efímeros, cuando son creados cumplen con sus objetivos o tareas y una vez que estos terminan se apagan, a menos que se configuren al iniciarlos (*-d*) para funcionar por el background.

1. En Docker, ¿qué es una imagen?

Es un archivo que contiene todas las paqueterías, librerías y programas que se necesitan para crear el contenedor de la aplicación.

1. ¿Como obtienes un listado de todos los contenedores que están corriendo, detenidos y pausados?

Con el comando: *docker ps –a* donde el parámetro es para indicar que muestre todo.

1. Supón que tienes 3 contenedores corriendo, cuyos ids son as65d4as, t6r54ght y e658d4hg respectivamente, ¿cómo accedes a uno de ellos de forma interactiva con la terminal bash?

Si el contenedor está detenido primero se arrancaría con el comando: *docker start as6*

Una vez que ya está encendido, para entrar en él se utiliza el comando: *docker exec –it as6 bash*

1. Al momento de usar el comando "docker images", obtenemos un listado con las imágenes almacenadas localmente,entre ellas, vemos la imagen "ubuntu", al usar el comando "docker image rm ubuntu" ¿cuáles podrían ser los 2 posibles resultados y a que se deben?

Uno puede ser de error y este puede suceder porque un contenedor está utilizando esa imagen impidiendo la eliminación de esta. Y el otro resultado es que lo borre exitosamente debido a que no está siendo utilizada por ningún contenedor.

1. ¿Qué pasa con las modificaciones hechas en un contenedor al momento que este se para?

Se quedan guardadas, pero solo existen dentro de ese contenedor, si se elimina se eliminan las modificaciones o si se levanta un contenedor de la misma imagen que el otro contenedor este no tendrá esas modificaciones.

1. ¿Cuál es la diferencia entre docker stop y docker kill?

Con docker stop se manda un trigger al contenedor indicando que será detenido para que vaya parando los servicios o tareas que esté realizando y así se detenga de la forma correcta y segura. Con docker kill estamos forzando al contenedor a que se detenga, sin importar que esté haciendo se detendrá lo que podría llegar a causar problemas en el contenedor.

1. ¿A que nos referimos cuando decimos que un contenedor es la instancia de una imagen?

Es el cambio de estado de la imagen a ejecución convirtiéndose en un contenedor.

1. ¿Qué es un Dockerfile?

Es un archivo sin extensión que contiene instrucciones para poder crear una imagen y esta contenga todo lo necesario, ya sean programas, archivos, instrucciones de arranque, etc.

1. ¿Como persistimos data a través de contenedores?

Utilizando los volúmenes mapeando un directorio, así cuando se crea un contendor y se le monta el volumen este podrá tener acceso a los datos que estén dentro del directorio mapeado por el volumen.

1. Crea una imagen a través de un dockerfile usando como base la imagen de centos:7, la imagen tendrá que estar updateada y tener vim y wget instalados

FROM centos:7

RUN yum update –y

RUN yum install wget –y

RUN yum install vim -y

1. ¿Cómo corremos un contenedor usando la imagen de ubuntu montando el volumen xy-products a la carpeta /data del contenedor?

Con el comando: *docker run –it –v xy-products:/data Ubuntu bash*

1. ¿Por qué hicimos una Docker-machine, cuál es su funcionalidad?

Porque nos encontramos en un entorno Windows y los sistemas que íbamos a utilizar son Linux por lo que se requería una maquina donde estuviéramos en un entorno Linux para poder trabajar.

1. ¿Cuáles son los pasos para tener mi entorno listo para comenzar a correr contenedores?

Suponiendo que la docker machine es la default se realizan los siguientes pasos:

* Primero se enciende la docker machine: *docker-machine start*
* Como segundo paso es especificar las variables de entorno: *docker-machine env*

Este comando arrojara unas líneas como resultado, se copiarán las primeras 5 lineas y se pegaran en la consola y después se copiará la última línea desde el símbolo & y se pega en consola.

* Por ultimo como buena práctica se entra a la docker machine con secure Shell: *docker-machine ssh*

1. Durante el diseño de una aplicación, voy a tener 3 servicios, A, B y C respectivamente, los cuales van a estar contenidos cada uno dentro de un contenedor. El servicio A necesita comunicarse con el servicio B, y el B necesita comunicarse con el C, pero nos es esencial, que el servicio A, por cuestiones de seguridad, este completamente aislado del servicio C. ¿de que manera podemos asegurar este asilamiento?

Esto puede lograrse por medio del *Networking* creando dos redes una por ejemplo llamada *red1* y la otra *red2*, donde en *red1* estarán conectados los servicios A y B y en *red2* los servicios B y C. De esta forma el servicio B al estar en ambas redes tendrá comunicación con el resto de los servicios los servicios A y C no podrán ya que se encuentran en redes diferentes.

1. Tenemos una base de datos la cual esta conectada a un servicio CRUD, sin embargo, nuestro servicio CRUD lo queremos separar en 2 contenedores, uno que se encargue solamente CUD y otro solamente a R. ¿de que manera podemos lograr esto?

Se crearán dos contenedores donde ambos se puedan conectar al servicio, pero un contenedor tendrá los permisos de lectura y escritura y el otro contenedor solamente de lectura.

1. Levantamos un contenedor de centos:7, al cual le generamos un archivo de texto llamado schedule.txt, apagamos el contenedor. Después de un tiempo, por error, borramos nuestro archivo schecule.txt, y generamos otro con el mismo nombre, pero con contenido diferente, ¿cuál será el output al usar el comando Docker container diff en este contenedor?

Se visualizará el que se ha creado el archivo shecule.txt en el contendor y con el comando *diff* se compara el contenedor con la imagen y la imagen al inicio no lo tenía.

1. Al momento de correr un contenedor de Ubuntu, generamos un archivo llamado todo.txt, este contenedor le hacemos commit como una nueva imagen llamada myimg, al levantar un contenedor de esta nueva imagen, elimino el archivo todo.txt y genero otro con un nuevo contenido. ¿Cuál será el output al usar el comando Docker container diff en este contenedor?

Mostrará que el archivo todo.txt fue editado.