**SEGUNDO PARCIAL DE INFRAESTRUCTURA COMO CÓDIGO**

*Datos Obligatorios:*

Nombre y Apellido:

Dni:

Fecha:

**Fila 2**

El examen consta de una parte teórica y una parte práctica, y se evaluará de acuerdo a

los siguientes criterios:

Criterios de evaluación para la parte teórica (30 puntos en total):

* Respuestas correctas y completas a cada una de las preguntas planteadas.
* Presentación clara y ordenada de las respuestas.
* La parte teórica del examen debe estar desarrollada en el mismo archivo de word proporcionado para el parcial, junto con el diseño del diagrama que se pide en la parte práctica.

Criterios de evaluación para la parte práctica (70 puntos en total):

* Proporcionar un archivo de Docker Compose completo y estructurado para la aplicación web y las bases de datos.
* Configurar el mapeo de puertos en Docker Compose para acceder a la aplicación web desde el puerto local de los desarrolladores.
* Establecer la configuración de volúmenes en Docker Compose para persistir datos de la aplicación web y las base de datos.
* Desarrollar el archivo Docker Compose siguiendo buenas prácticas de desarrollo, con código legible, estructurado y comentarios adecuados.
* Proporcionar un diagrama de arquitectura que muestre la interacción entre los componentes en el entorno Docker Compose.
* Todo el código debe estar subido a un repositorio de GitHub, con el nombre del título y el commit que pide la consigna, en el horario correspondiente de entrega. Al subirlo se deben ignorar los volumenes y el node\_modules.

**Teórico**

Explique la diferencia entre imagen y contenedor. (5 Puntos)

¿Ventajas de la Infraestructura como Código? (5 Puntos)

¿Qué es el Docker Engine? (5 Puntos)

¿Para qué sirve el Dockerfile? (5 Puntos)

¿A qué se refiere el término “Orquestar Contenedores” y para que se usa en el mundo IT? (5 Puntos)

¿Para qué sirve Docker Compose? (5 Puntos)

**Práctica**

En Aguas de Formosa SA, se ha anunciado recientemente que se dejará de brindar soporte a MySQL, el motor de base de datos utilizado en el sistema desde sus inicios. Como programador con más experiencia en la empresa, se te ha encomendado la tarea de preparar una nueva infraestructura de desarrollo utilizando Cloud Computing.

Antes de considerar la adopción de servicios en la nube como AWS, es necesario abordar la migración del sistema antiguo desarrollado en Delphi hacia tecnologías más modernas. En este sentido, se ha decidido utilizar MariaDB como sucesora de MySQL para evitar problemas a largo plazo. Además, se ha contratado un nuevo equipo de desarrolladores especializados en el stack MERN (MongoDB, Express.js, React.js y Node.js) para trabajar en otra parte del nuevo sistema.

Tu objetivo es configurar el entorno de desarrollo para hacer uso de MariaDB y MongoDB, y asegurarte de que ambas bases de datos estén en línea y accesibles. Esto permitirá a los nuevos desarrolladores y a los desarrolladores antiguos trabajar de manera eficiente en el sistema.

Se solicita implementar endpoints para verificar el estado de las bases de datos. Por ejemplo.

Endpoint de verificación de conexión a MariaDB:

URI: /check-mariadb-connection

Endpoint de verificación de conexión a MongoDB:

URI: /check-mongodb-connection

Recuerda que debes migrar el sistema existente a MariaDB y configurar MongoDB. Asegúrate de que los nuevos desarrolladores y los desarrolladores antiguos puedan acceder a las bases de datos y trabajar en conjunto de manera fluida.

Objetivos

* Diseñar un diagrama que permita entender a simple vista cómo funciona la arquitectura, agregando los puertos en los que escuchan, los endpoints y las imagenes que utilizan. (20 Puntos).
* Desarrollar e implementar los archivos y paquetes necesarios para dejar funcionando el entorno  (50 Puntos).

Una vez hecho toda la parte práctica se debe subir todo a un repositorio en github con las siguientes caracteristicas:

**Título repositorio:** *“SEGUNDO\_PARCIAL”*

**Commit**: *“NOMBRE\_APELLIDO\_ALUMNO-FECHA\_HORA”* .Ejemplo: ***“juan\_perez-21-06-2023:12:00”***

Cabe aclarar que solamente se permite un *commit* en el repositorio, se debe respetar el formato anterior, en caso de no respetar se le restaran puntos.

**DESARROLLO**

**Teoria:**

Explique la diferencia entre imagen y contenedor. (5 Puntos)

Las diferencias entre imagen y contenedor son que una imagen de Docker es como plantilla estática que contiene todo lo necesario para ejecutar una aplicación, incluyendo el código, las bibliotecas, las dependencias y los recursos del sistema operativo. Y un contenedor es una instancia en ejecución de una imagen de Docker. Se lo puede tartar como un entorno aislado y ligero que contiene todos los componentes necesarios para que una aplicación se ejecute. Cada contenedor se basa en una imagen y se ejecuta de forma independiente, con su propio sistema de archivos aislado.

¿Ventajas de la Infraestructura como Código? (5 Puntos)

-Automatización: La IaC permite automatizar la implementación y configuración de la infraestructura. El código se encarga de gestionar y desplegar la infraestructura de manera eficiente, ahorrando tiempo y esfuerzo.

-Reproducibilidad y consistencia: Al utilizar código para describir la infraestructura, se obtiene una forma de reproducir exactamente el mismo entorno una y otra vez.

-Control de versiones y seguimiento de cambios: El tener el código de infraestructura en un repositorio de control de versions, permite realizar un seguimiento de los cambios y gestionar diferentes versiones.

-Escalabilidad y flexibilidad: La IaC permite escalar la infraestructura de manera rápida y fácil, ya que se puede modificar el código para agregar o eliminar recursos según sea necesario.

-Documentación y conocimiento compartido: El código de infraestructura sirve como documentación viva y detallada de la infraestructura, lo que facilita la comprensión y el conocimiento compartido del entorno.

¿Qué es el Docker Engine? (5 Puntos)

Docker Engine es el componente central de Docker que permite la creación y ejecución de contenedores. Es el motor de tiempo de ejecución responsable de administrar los contenedores y proporcionar un entorno aislado y portátil para ejecutar aplicaciones. Docker Engine tiene tres apartados clave: el docker daemon que es un proceso que se ejecuta en Segundo plano en el Sistema host, la api de docker y el cliente docker.

¿Para qué sirve el Dockerfile? (5 Puntos)

El dockerfile sirve para construer una imagen de Docker, éste trae instrucciones que describen como construir una imagen dentro de su archivo y estas instrucciones son utilizadas por Docker Daemon para automatizar el proceso de construcción de la imagen.

¿A qué se refiere el término “Orquestar Contenedores” y para que se usa en el mundo IT? (5 Puntos)

El término orquestar contenedores se refiere a no trabajar contenedores de manera individual, el orquestar contenedores permite una gestion eficiente de multiples contenedores en un entorno distribuido. Se utiliza en elm undo IT para gestionar tareas como el despliegue, escalade, balanceo de carga, recuperacion de fallos y gestion de configuraciones de forma centralizada.

¿Para qué sirve Docker Compose? (5 Puntos)

Docker Compose sirve para simplificar la gestion de aplicaciones por multiples contenedores, permite definir y configurar servicios, redes y volumenes relacionados a un archivo YAML llamado “docker-compose-yml”.