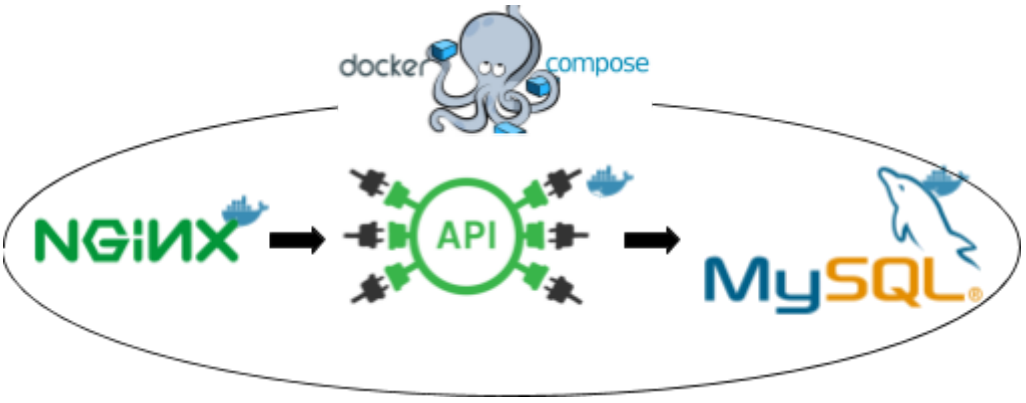
Mercadolibre challenge

Motivación:

Creemos que hoy en día es necesario tener un conocimiento amplio de todos los componentes que intervienen en el desarrollo de un proyecto. Por este motivo, con este *challenge,* buscamos que todos los miembros del equipo de seguridad tengan un conocimiento básico de Networking, Infraestructura, Desarrollo y Base de Datos.

Objetivo:

En este challenge se presentan 3 contenedores de Docker orquestados con *docker-compose*. Detallaremos a continuación la arquitectura propuesta:



Al momento de inicializar *docker-compose* vemos que todo levanta bien, pero no podemos acceder a los métodos de API.

1. **Se deberá hacer los cambios necesarios respetando la arquitectura propuesta para llegar a los métodos de la API.**
    - **Ejemplos:**
        - **POST** http://localhost:8080/item
        - **GET** http://localhost:8080/item/1
2. **También se necesitará agregar ciertos métodos que interactúen con la API de google (*https://developers.google.com/drive/v3/web/about-sdk*):**

GET /search-in-doc/:id

Este método lo que hará es recibir un ID de un archivo y buscarlo mediante la api de google dentro del Drive del usuario. Luego deberá verificar si la palabra enviada por parámetro se encuentra dentro del mismo.

Parámetros:

Nombre	Tipo	Descripción
word	string	Indica la palabra a buscar dentro del archivo cuyo id es *:id*

Respuesta en el caso de encontrarlo

Status: *200*

Respuesta en el caso de no encontrarlo

Status: *404*

Ejemplo:

Request:
GET /search-in-doc/1?word=dev

Response:
HTTP/1.1 200 OK

POST /file

Este método lo que hará es recibir un título y una descripción con las cuales creará un archivo nuevo en el Drive del usuario. Luego almacenará en la base de datos el id, título y descripción y finalmente los devolverá en forma de JSON.

Parámetros:

Nombre	Tipo	Descripción
titulo	string	Indica el título del archivo a crear
descripción	string	Indica el contenido del archivo

Respuesta en el caso de poder crearlo

Status: *200*

Respuesta en el caso de que los parámetros estén mal

Status: *400*

Respuesta en el caso de que no pueda crearlo

Status: *500*

Ejemplo:

Request:
POST /file

{“titulo”:”Pagos a prov”, “descripcion”:”Tengo que hacer un pago”}

Response:
HTTP/1.1 200 OK
{“id”:”2”, “titulo”:”Pagos a prov”, “descripcion”:”Tengo que hacer un pago”}

Aclaraciones:

- Para almacenar los datos en el ejemplo propuesto la API utiliza **SQLITE**, la entrega deberá utilizar la base de datos mysql que se encuentra en el contenedor de docker de **MYSQL**.
- En todo momento se deberá respetar la arquitectura propuesta.

Entregables:

- Descripción de la aplicación realizada, problemas y soluciones con los que se encontró al realizar el challenge.
- docker-compose.yml
- Archivos Dockerfile
- Código de la API
- Archivo .sql con las sentencias requeridas para hacer cambios en el esquema de base de datos de ser necesario

Bonus:

- Armar un repositorio en Github como entregable
- Test unitarios de la API
- Test de integración