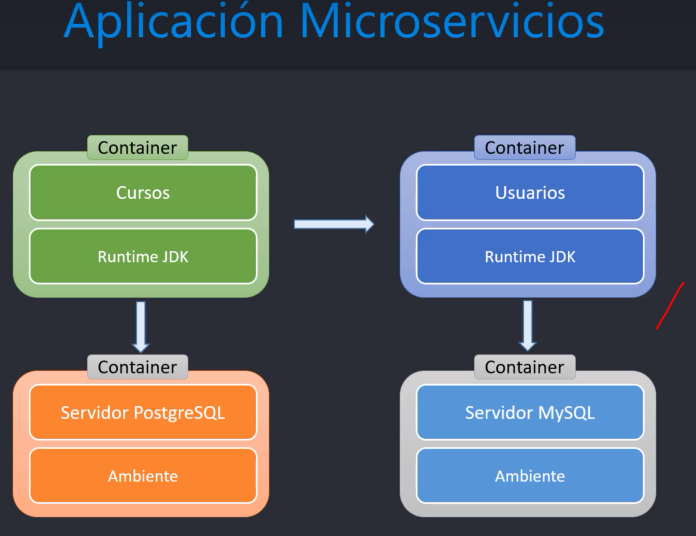
10. Docker Compose: Orquestador para definir y ejecutar multi-contenedores

Introducción Docker Compose

La idea es minimizar el tema de comandos cada vez que se actualiace, se ejecute o realizar algo en algun contendor.

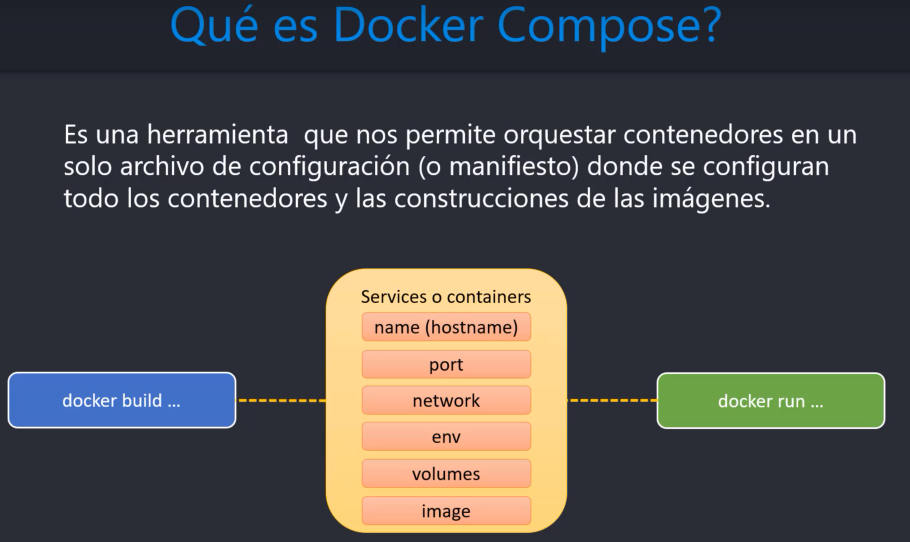
Para ello el Docker Compose tendrá todos los comandos para CRUD sobre los contenedores y se levantara solo con un solo comandos



Si no controlamos el flujo en un app pequeña para administrar una app grande serán problemas, por buenas practicas se debe realizar el Docker compose y no manejar todo por comandos independientes



Ojo: si se implementara el Docker Compose es primordial que todos los contenedores se encuentren dentro de la misma maquina(Para un sacar provecho un 100%)

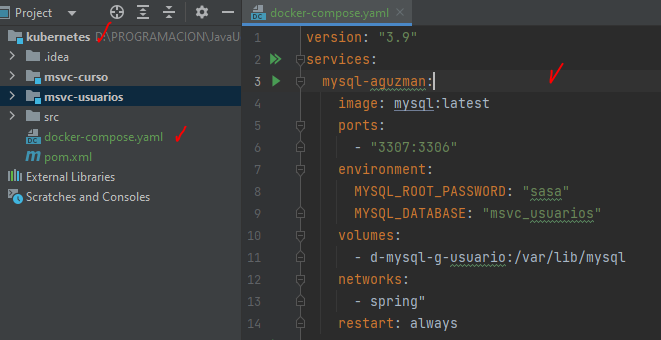


Instalando Docker Compose en Linux

Creando el archivo docker compose

Docker compose, utiliza los dockerFIle y trabaja con las imágenes que ya se tienen. Para el ejemplo eliminaremos todos los contenedores de microservicios y DB

* Crear Proyecto modular kubernetes, que contendrá los demás proyectos
  + Configurar packagin a POM padre
  + Refresh los pom Hijos
  + Configurar POM HIJOS para que la etiqueta PARENT HIJO se corte y quede en el pom Modular y las etiquetas groupId, artifacId y versión copiadas del modular en el HIJO.
  + Agregar la etiqueta module en el PADRE el nombre de los proyectos hijos
  + Refresh los pom Hijos
  + Probar
* Crear archivo Docker-compose.yaml (puede ser también yml) desde la raíz (kubernetes)
* Configurar archivo Docker-compose.yaml
  + Validar la versión del Docker Compose a utilizar <https://docs.docker.com/compose/compose-file/compose-file-v3/>
  + Realizar en cascada la ejecución de cada servicio y luego agregar detallada mente los atributos en ejecución para cada servicio



DATO: Kubernetes(Orquestador mult maquinas) es muy parecido a Docker swarm (Orquestador) solo que kubernetes es más Robusto

DATO: Para este caso crear el NETWORK es opcional ya que el Docker compose crea su propia red que permite la conexión entre todos los componentes, igualmente si se quere utilizar la red se puede dejar en la sección general del archivo Docker-compose (red por defecto)y no dentro de cada servicio

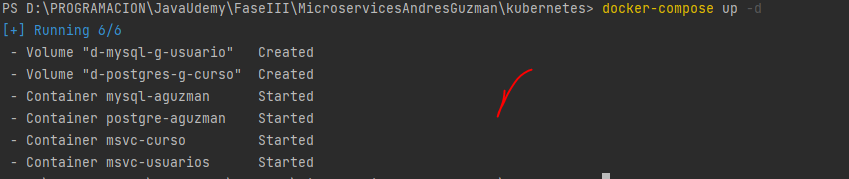
DATO: Todo lo que se utiliza en común entre los contenedores se puede hacer a nivel raziz del Docker-compose, entre ellos el network, volumes …

Añadiendo contenedores de microservicios

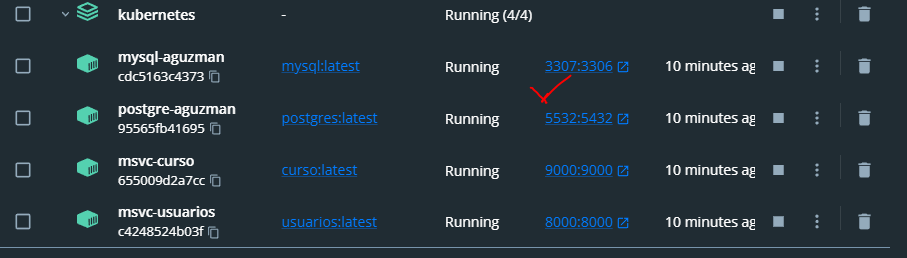
* Configurando los microservicios dentro del Docker compose
* Atributo depends\_on de Docker compose el contenedor depende de un contendor padre para levantarse

Ejecutando todo con docker-compose Up y Down para detener y eliminar todo

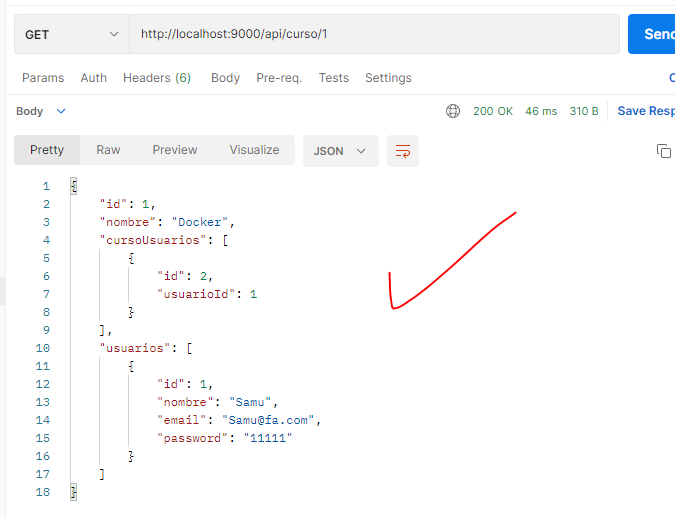
* Levantando el Docker-compose up -d



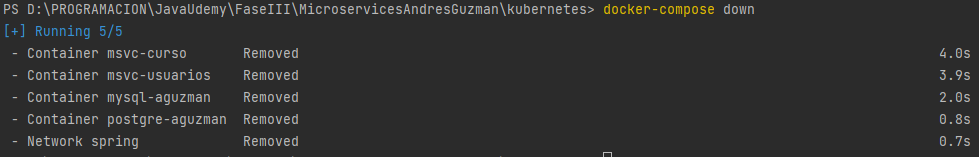
IMPORTANTE: Nunca pero nunca eliminar Volúmenes, declarar muy bien los volúmenes con su name en el Docker compose igualmente a los nombre de los containers.



* Prueba que todo ese funcionaod bien



* Puedo ver el log de cada contenedor
  + docker logs postgre-aguzman
* Detener la ejecución y elminar los contenedores

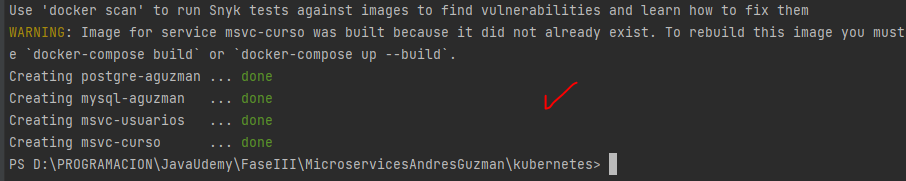


Build imagen en docker compose

EN este caso se automatizará la construcción de las imágenes, el Build se realizará desde el Docker-compose,

* Ajustar los dockerfile para que trabajen en compilación con el Parent.
  + Ajustar las carpetas que van a la cache o que se van mantener guardados y solo actualice la carpeta SRC cuando tenga cambios y así reducir los tiempos y tamaños de compilación por Capas, previsualizando una imagen y reutilizando la previsualizando.
* Ajustando el Docker-compose, para que no cargue las imágenes en la compilación si no que las ejecute dentro de la compilación (Build) solo para los microservicios
* Levantar el Docker-compose, es opcional si quiero levantarfromzada las imágenes





* Pruebas

