**PLATAFORMAS e INFRAESTRUCTURAS**

**PUNTO CONTROL: CONCEPTOS GENERALES DOCKER**

**(2020/10/05)**

Al finalizar el punto de control, cada alumno deberá subir un documento (similar al de las prácticas) con los resultados obtenidos durante los pasos ejecutados en la prueba a la plataforma Mudle.

Para ello se ha creado una tarea dentro de la asignatura. No habrá posibilidad de subir trabajos a partir de las 16.30.

1. **Cuestionario (1.5 puntos)**

Link del cuestionario en Mudle. (15 minutos. Activo hasta las 15.00)

1. **Contenedorizando una aplicación (8.5 puntos)**

En este ejercicio, tendrás que poner en marcha una aplicación distribuida y basada en microservicios.

Aplicación Votación

====================

Aplicación distribuida que corre en múltiples contenedores Docker.

Getting started

---------------

## Contenedores Linux

Los stack de Linux usan Python, Node.js, Java, con Redis para la mensajería y Postgres para almacenamiento.

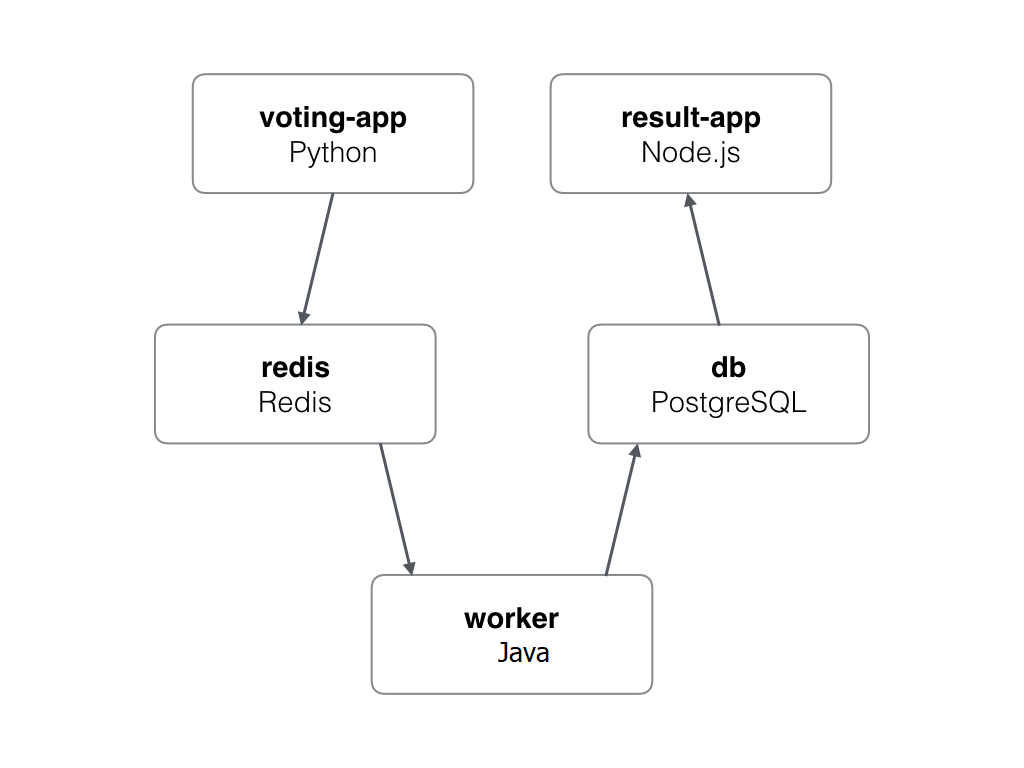
Una vez terminado todo el desarrollo, ejecuta:

docker-compose up

La aplicación debería de ejecutarse en [http://localhost:5000](http://localhost:5000), y los resultados deberían de aparece en [http://localhost:5001] (http://localhost:5001).

Arquitectura

-------------



\* Aplicación web front-end en [Python](/vote). Su funcionalidad será permitir votar entre dos opciones

\* Cola [Redis](https://hub.docker.com/\_/redis/) qque recoge los nuevos votos

\* Trabajador [Java](/worker/src/main) que consume los votos y los guarda en …

\* Base de datos [Postgres](https://hub.docker.com/\_/postgres/) que database que se apoya en un volumen Docker

\* Aplicación web en [Node.js](/result) que muestra los resultados de la votación en tiempo real

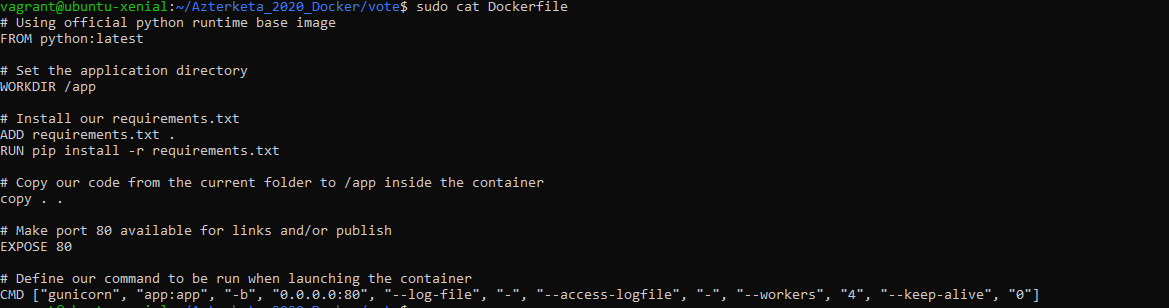
Nota

----

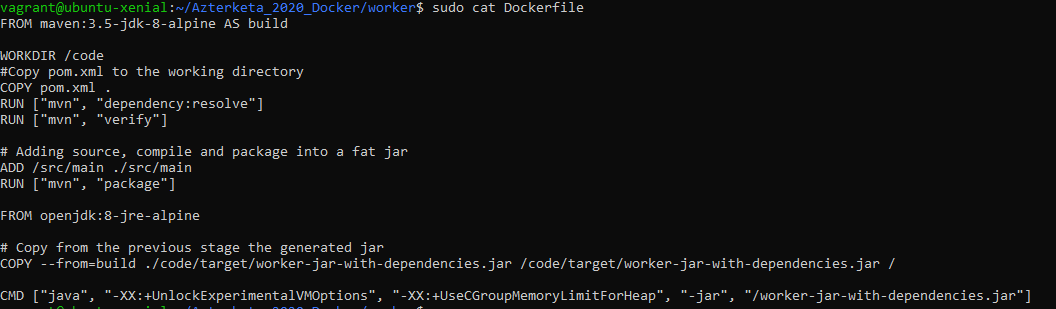
La aplicación acepta un único voto por cliente. No registra votos que hayan sido introducido por el mismo cliente.

Tareas a realizar:

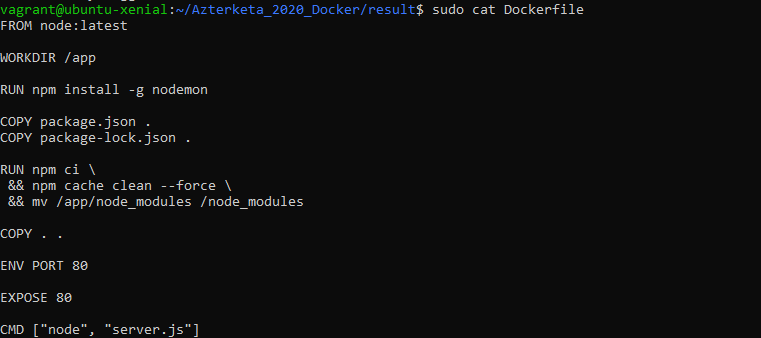
* Voting-app: escribir el Dockerfile para esta aplicación. **(2 puntos).**
  + La imagen base será Python (tal y como se indica en el propio Dockerfile) y el directorio de trabajo es /app.
  + Para que la instalación sea adecuada y tenga en cuenta los requisitos, hay que copiar el fichero de requisitos, en el directorio de trabajo.
  + Una vez instalado todas las librerías indicadas en los requisitos, se copiarán todos los ficheros de la aplicación, en el directorio de trabajo
  + Exponer el puerto 80
  + El comando a ejecutar ya está escrito en el Dockerfile

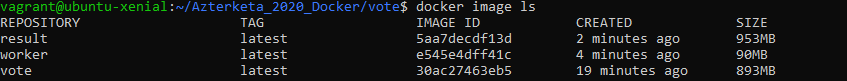


* Worker: escribir el Dockerfile para esta aplicación. **(2 puntos)**
  + La imagen base es de maven (ya incluida en el Dockerfile), y el directorio de trabajo está situado en /code.
  + Copiar el fichero pom.xml en el directorio de trabajo.
  + Ejecuta los comandos ya escritos en RUN (ya incluidos en el Dockerfile) y añade el contenido/programa/desarrollo que se encuentra en /src/main en la misma estructura pero dentro del directorio de trabajo dentro del contenedor.
  + Ejecuta el comando RUN (ya incluido en el Dockerfile) que generará el .jar.
  + La segunda etapa del Dockerfile se basa en la imagen de java (ya incluido en el Dockerfile), y copiar el jar generado en el stage anterior en el directorio indicado en el Dockerfile (ya incluido en el Dockerfile)
  + El comando a ejecutar ya está escrito en el Dockerfile

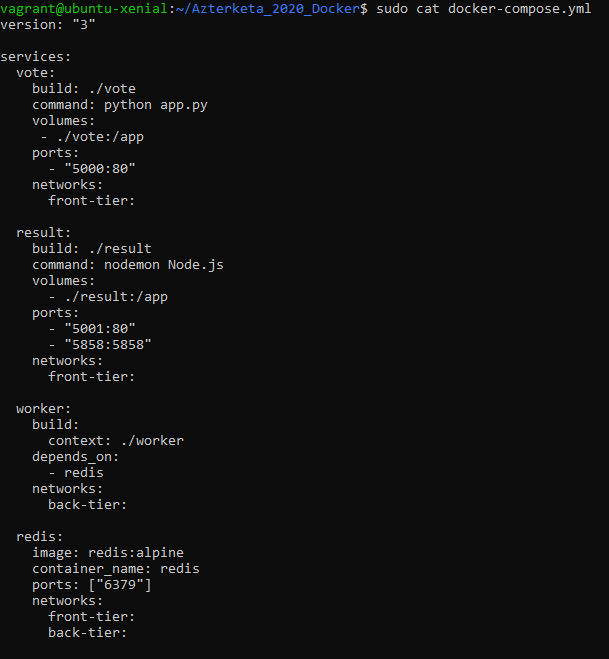


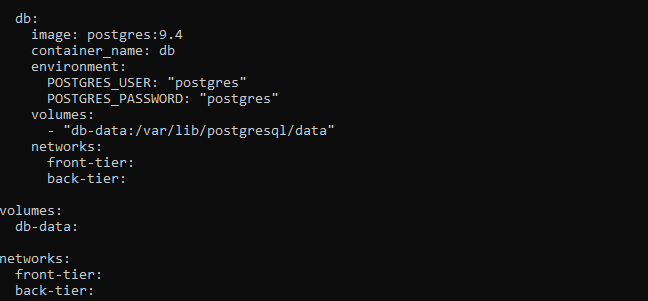
* Result: escribir el Dockerfile para esta aplicación. **(2 puntos)**
  + La imagen base es node configura el directorio de trabajo en /app.
  + Ejecuta el comando RUN para instalar todo lo necesario para la aplicación node (ya incluido en el Dockerfile)
  + Copiar todos los ficheros .json que están en el directorio Result en el directorio de trabajo.
  + Ejecuta los comando RUN para generar la aplicación node (incluido en el Dockerfile).
  + Copiar todo lo generado en el fichero que tenemos en el directorio result del host en el directorio de trabajo.
  + Declarar como variable de entorno PORT con el valor 80 y exponer el puerto 80.
  + Ejecutar la aplicación server.js con el comando node (incluido en el dockerfile)

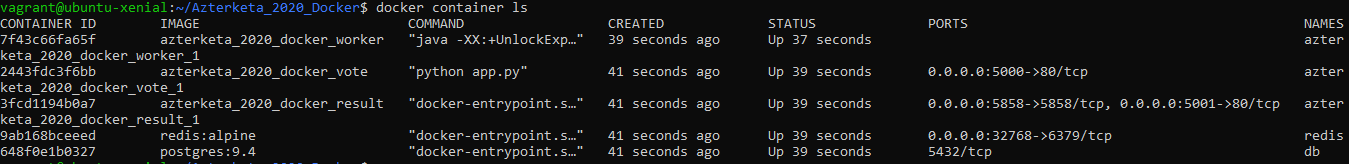


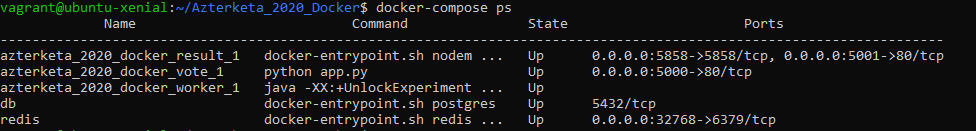


* Docker Compose: escribir el fichero docker-compose para lanzar toda la aplicación mediante el comando docker-compose up.**(2.5 puntos)**









En el siguiente enlace, encontrarás toda la infraestructura necesaria para poner en marcha la práctica:

https://gitlab.danz.eus/macc/pi\_docker/Azterketa\_2020\_Docker.git

FEEDBACK

¿Cambiarías o quitarías algún tema? ¿Plantearías la asignatura de otra forma?

¿Qué te ha parecido el ritmo de impartición? ¿Y el nivel de dificultad?

¿Qué te ha parecido la primera parte de la asignatura? ¿Crees que ha sido útil para tu formación?

¿Alguna propuesta de mejora?