Práctica 4: Despliegue de un sistema de información Web con persistencia de datos utilizando contenedores Docker

Sistemas de Información

Grado de Informática
Dpto. de Informática e Ingeniería de Sistemas, Universidad de Zaragoza
Escuela de Ingeniería y Arquitectura

27 de noviembre de 2019

1. Objetivos

Las aplicaciones Web son aplicaciones que se ejecutan en al menos un servidor Web y a las que los usuarios acceden desde un cliente de propósito general (por ejemplo: un navegador Web en un PC, teléfono móvil, etc.).

Para conseguir este objetivo, una vez diseñada y desarrollada nuestra aplicación, es necesario desplegarla en un servidor que ejecute los distintos componentes de la misma, a saber:
- Servidor de aplicaciones
- Servidor de base de datos
- (Opcional) Servidor http/https

En mayor detalle, los objetivos de esta práctica son:

- Identificar y preparar los componentes de la aplicación que debemos desarrollar. (Código, datos, elementos estáticos...)

- Diseñar y desarrollar los scripts y ficheros de configuración para el despliegue de los componentes de la aplicación sobre contenedores docker.
- Verificar la correcta ejecución de los scripts para el despliegue de la aplicación desde cero.

El entorno descrito en el presente documento presupone una aplicación desarrollada en JEE sobre Tomcat, y con MariaDB como base de datos. No obstante, los diferentes grupos de prácticas pueden seleccionar tecnologías y herramientas análogas, adaptando los scripts de despliegue a la tecnología seleccionada.

2. Identificación y preparación de los elementos

En primer lugar, crearemos o identificaremos un directorio de nuestro equipo en el que copiaremos todos los fichero necesarios para crear nuestro contexto de despliegue. Este será nuestro directorio de trabajo. Puede ser, por ejemplo, una subcarpeta dentro de nuestra carpeta del proyecto, en el Workspace de nuestro IDE (Interface Development Environment). Llamaremos a esta carpeta "docker".

$ $HOME/Workspace/sisinf/docker

Los elementos básicos que necesitamos para desplegar la aplicación son:
- Fichero WAR (o equivalente) con el código, parte estática y dinámica de la aplicación web.
- Ficheros con los datos de nuestro modelo de datos en MariaDB (o equivalente).

El fichero WAR ya lo hemos creado y probado en las prácticas anteriores, y cualquier IDE adaptado a JEE (como Eclipse o IntelliJ), nos generar el fichero WAR de la aplicación con todas las garantías. Es importante acordarse de incluir en el directorio WEB-INF/lib todas las librerías (*jar) que necesite nuestra aplicación, y que no estén incluídas en Tomcat, como por ejemplo, la librería de los tags estándar de JSP (jstl). En Eclipse, esta inclusión se configura en el Java Build Path de las propiedades de nuestro proyecto.

Copiaremos el fichero WAR de nuestra aplicación en la carpeta "tomcat" dentro de nuestro directorio de trabajo

$HOME/Workspace/sisinf/docker/tomcat/<*proyecto*>.war

Para la base de datos, vamos a aprovecharnos de que, tal como configuramos en su día la base de datos con Docker, utilizando un volumen externo, todos los datos están

fuera del contenedor Docker, en una carpeta de nuestro equipo. Si copiamos esa carpeta en otro equipo, en la ubicación y con los permisos correctos, y arrancamos una instancia de MariaDB, ya tenemos configurada y funcionando una nueva base de datos, con todos los datos preservados.

Por esa razón, para poder desplegar la aplicación y la base de datos en otro equipo, iremos a la carpeta donde están los datos, y la comprimiremos en un fichero ZIP.

Originalmente, los datos estaban en la carpeta identificada por la variable $PORTAL_PATH del script de creación de MariaDB

$ $HOME/sisinf/mysql

Así pues, comprimiremos todo el contenido de sisinf en un fichero zip, de manera que, al descomprimir, nos encontremos, en primer lugar, con la carpeta "mysql". Con objeto de no guardar en el ZIP más información de la necesaria, antes de esta compresión eliminaremos los ficheros ib_logfile* y aria_log_*. Estos ficheros serán regenerados automáticamente al iniciar la base de datos.

Copiaremos el fichero sisinf.zip en el directorio *mariadb* de nuestro directorio de trabajo

$HOME/Workspace/sisinf/docker/mariadb/sisinf.zip

3. Preparación de scripts de despliegue

Ahora que ya tenemos todos los componentes necesarios, vamos a construir los scripts para regenerar todos los componentes de nuestra aplicación, utilizando los scripts que ya teníamos en prácticas anteriores.

A) Regeneración de la base de datos.
La única ampliación que necesitamos hacer al script *create_mariadb.sh* de la práctica 2 es la necesidad de descomprimir el fichero comprimido de la base de datos en la ubicación correcta, y probablemente cambiar los permisos y/o el propietario de los ficheros, para que el contenedor de MariaDB pueda leerlos sin problemas. Por seguridad, borraremos los datos que tengamos de pruebas o aplicaciones anteriores. Incluiremos estas instrucciones en el script de creación de la base de datos.

```
rm -R -f $PORTAL_PATH/*
unzip -X -o -d $PORTAL_PATH mariadb/sisinf.zip
chown -R 1001:root $PORTAL_PATH
```

B) Servidor de aplicaciones

Con los scripts y los ficheros de configuración de las prácticas anteriores, tenemos todo el material necesario para generar el servidor de aplicaciones. Si hemos cambiado algún elemento de la configuración original (directorios de despliegue, nombre de la base de datos...), adaptaremos los scripts y los ficheros de configuración (server.xml, context.xml, etc.) a nuestra configuración particular.

Podemos juntar la creación de la base de datos (MariaDB) y del servidor de aplicaciones (Tomcat), fusionando los dos scripts en uno solo.

4. Pruebas del sistema

Antes de cerrar la práctica y entregarla, es importante validar que funciona todo correctamente. Para ello, bien en un equipo "limpio", o en uno de los equipos de desarrollo, pero previo borrado de todos los ficheros, carpetas y contenedores Docker preexistentes de la aplicación, se probará a ejecutar el script de despliegue, junto con el resto de ficheros, y se verificará que funciona todo correctamente. Esta comprobación debe hacerse, al menos, en dos equipos distintos, para estar seguros de que todo funciona correctamente.

5. Entrega de la práctica

La práctica se realizará en grupos de tres personas. Cuando se finalice la práctica se debe entregar en un fichero denominado práctica3_NIP1_NIP2_NIP3.tar o práctica3_NIA1_NIA2_NIA3.zip (donde NIA1, NIA2 y NIA3 son los NIAs de los autores de la práctica) con el siguiente contenido:

1. Un fichero de texto denominado autores.txt que contendrá el NIA, los apellidos y el nombre de los autores de la práctica en las primeras líneas del fichero. Por ejemplo:

NIA	Apellidos	Nombre
545689	Rodríguez Quintela	Sabela

745689	Caamaño Cives	Antón
677893	López Marín	Adrián

2. Un fichero ZIP con el contenido de nuestra carpeta de trabajo (sisinf). Dentro de ese directorio estará el script que deberá desplegar nuestra aplicación completa, así como todos los fichero necesarios:
   a. carpeta Tomcat con los ficheros de configuración, el Dockerfile, el driver de la base de datos y el WAR de la aplicación.
   b. Carpeta MariaDB con el zip de los datos.
3. Una memoria de la práctica, en formato PDF, con los siguientes apartados:
   a. Alcance funcional final de la aplicación.
   b. Diferencias (a favor o en contra) entre la versión presentada en la práctica 0 y la versión final de la aplicación.
   c. Procedimiento para el despliegue e instalación de la aplicación.
   d. Cuestiones necesarias para el uso de la aplicación: Por ejemplo, cuentas y contraseñas de acceso del administrador del sistema, o forma de autenticarse en el mismo.
   e. Cronograma aproximado indicando esfuerzos dedicados al desarrollo del sistema, y valoración del grupo sobre el trabajo realizado. ¿Qué os ha costado más? ¿Qué habéis aprendido? ¿Qué os habría gustado implementar y no habéis podido?

Al descomprimir el fichero .tar se deben extraer los ficheros y directorios indicados en el directorio practica3_Nip1_Nip2_y_Nip3.tar. Es importante seguir las convenciones de nombrado y la estructura de ficheros y directorios descrita.

Para la realización de esta práctica se ha planificado una sesión.

Por tanto, la fecha límite de entrega es el día anterior (a las 23:59) a la siguiente sesión de prácticas en el laboratorio. Para la entrega del fichero .tar, se utilizará el enlace correspondiente disponible en la página de la asignatura en el Anillo Digital Docente (ADD) de la Universidad de Zaragoza en la plataforma Moodle 2 (http://moodle2.unizar.es). En caso de que no sea posible realizar la entrega a través del ADD se enviará dicho tar al profesorado de la asignatura mediante correo electrónico.

6. Procedimiento de corrección y recomendaciones

Una vez realizadas las prácticas y entregadas estas, cada grupo debe presentarlas al

profesorado de prácticas en la sesión de prácticas correspondiente al inicio de la práctica 5. Al realizar la presentación el profesorado le formulará cuestiones sobre las decisiones que ha realizado el grupo de trabajo.

La práctica debe entregarse en los términos indicados anteriormente, y la aplicación deberá funcionar correctamente en los equipos de los profesores, tras ejecutar el script de despliegue suministrado por cada grupo.

- *Requisitos mínimos a cumplir (7 puntos)*
    - Funcionamiento correcto de los scripts de despliegue, y arranque correcto de la aplicación. (1 punto)
    - Funcionamiento global de la aplicación, sin errores de ejecución, y con la funcionalidad visible operativa (es decir, que no haya partes de la aplicación "aparentes", pero que no funcionen realmente). (3 puntos)
    - Aspecto global de la aplicación. Diseño del interfaz, accesibilidad y usabilidad. Alcance funcional. (2 puntos)
    - Contenido de la memoria y documentación del sistema de información: manual de despliegue, configuración y utilización. Metodología y costes de ejecución. (1 punto)
- *Otros requisitos (3 puntos)*
    - Aspectos relacionados con la presentación del informe (estructura, redacción, puntuación, índice, referencias y bibliografía ...) (1 punto).
    - Originalidad, aspectos novedosos y diferenciales que se han considerado en el proyecto en su conjunto (2 puntos).

Anexo. Opcional: utilización de docker-compose

Docker-compose es una utilidad, desarrollada en Python, que permite automatizar algunas de las tareas que efectuamos con Docker. Es particularmente útil cuando nuestro sistema está formado por varios contenedores, que tienen que comunicarse y coordinarse entre sí. Docker-compose tiene varias ventajas sobre la utilización de *docker* en estos casos:

- Permite configurar todos los componentes de nuestro sistema en un único fichero de configuración, *docker-compose-yml.*

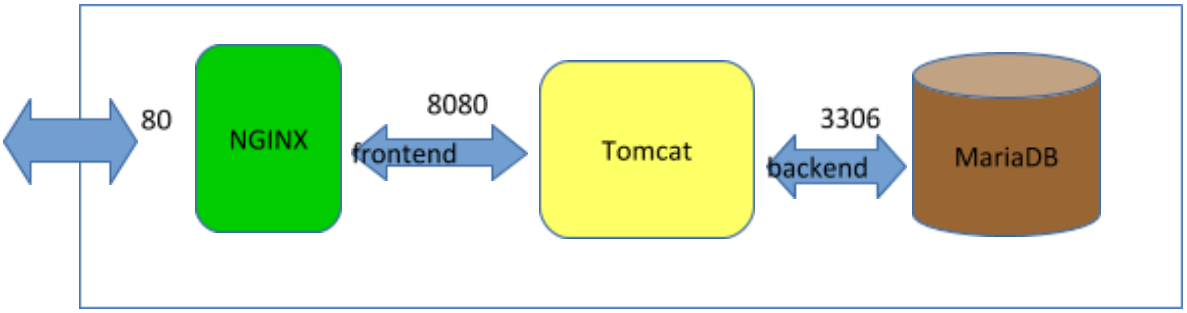
- Permite establecer dependencias entre los contenedores, de manera que, al arrancar el sistema, se arrancarán en el orden correcto. Por ejemplo, en nuestro caso, podemos hacer que el servicio "tomcat" dependa del servicio "mariadb", de manera que siempre se inicie mariadb antes que tomcat.

- Permite crear, arrancar, parar y borrar todos los servicios con un único comando.

En esta parte de la práctica, os proponemos que construyais el script y la configuración de la aplicación utilizando docker-compose, para lo cual, en primer lugar, deberéis instalar esta utilidad.

https://docs.docker.com/compose/install/

Además, aprovechando esta utilidad, y la facilidad de configuración de servicios, vamos a intentar configurar nuestro sistema con un plus de seguridad. En concreto, vamos a montar la siguiente configuración de servicios:



Añadiremos un frontal con el software Nginx. Este programa es un servidor HTTP, que configuraremos como proxy. Podría utilizarse también, si tuvieramos un certificado válido, para configurar el protocolo HTTPS, pero de momento, lo vamos a obviar.

Este proxy nos permite publicar en nuestro servidor únicamente el puerto 80 (http), pero ocultar los puertos 8080 de Tomcat y 3306 de MariaDB. El puerto 3306 de MariaDB sólo es visible para el contenedor de Tomcat a través de la subred virtual "backend", y el puerto 8080 de Tomcat sólo es visible para Nginx, a través de la subred virtual "frontend".

La configuración de Nginx es muy sencilla. Basta con crear un fichero *nginx.conf* con el siguiente contenido:

```
http {
   root /www/data;

   server {
      listen 80;

      location / {
         proxy_pass http://sisinf-tomcat:8080/<mi_proyecto>/;
      }
   }
}

events {
}
```

y ubicarlo en la ruta */etc/nginx/nginx.conf* de nuestra imagen Docker de Nginx. Utilizaremos como base la imagen *nginx* estándar de DockerHub.

Respecto a la configuración de docker-compose, es preciso definir dos redes (frontend y backend), tres servicio (mariadb, tomcat y ngnix) con sus dependencias indicadas anteriormente. Tomcat tendremos que construirlo (build) para incluir el WAR y demás componentes. MaríaDB no es necesario, y utilizaremos la imagen estándar. Nginx, deberemos construirlo también, porque hay que incluir el fichero de configuración. El resto de elementos, están ya indicados, o se pueden inferir del gráfico de configuración.

¿Os atrevéis a intentarlo? Esta opción vale 1 punto adicional a los 10 indicados en el apartado 6.