University of Seville

School of Computer Engineering

Dockerize a War+MySQL project

Software engineering

Design and testing 2

2018-2019

06/06/2019

Group 26

Content

What is Docker?

Docker is a tool designed to facilitate the creation, implementation and execution of applications with containers. Containers allow a developer to package an application with all the parts it needs, such as libraries and other dependencies, and send everything in a single package.

Containers allow a developer to package an application with all the parts it needs, such as libraries and other dependencies, and ship everything in one package. They are designed to facilitate the provision of a consistent experience as developers and system administrators move code from development environments to production in a fast and replicable manner.

Steps to dockerize our project

Step 1: Generate a Dockerfiles for Tomcat and MySQL

Dockerfile is a text document that contains all the commands a user could call on the command line to assemble an image. We must specify a Dockerfile per dependency in our application. In this case, a Dockerfile for Tomcat (v7.0) and MySQL (v5.5).

```
FROM tomcat:7

COPY Acme-Packages.war /usr/local/tomcat/webapps/
```

1. Dockerfile for Tomcat

```
FROM mysql:5.5

ADD my.cnf /etc/mysql/my.cnf

ENV MYSQL_ALLOW_EMPTY_PASSWORD="yes"

COPY Create-Acme-Packages.sql /docker-entrypoint-initdb.d
```

2. Dockerfile for MySQL

With **FROM** directive, we indicate the base image to build. In this case, as mentioned before, we use version 7 of Tomcat. This directive must be in every Dockerfile, and it's necessary to place it in first line, because Dockerfiles are executed by order.
With **COPY** directive we indicate that Acme-Packages.war must be copied into /webapps directory.

In the case of MySQL, we must take into account that lower_case_table_names variable in my.cnf configuration file must be set to 1, so that way tables will be stored in lowercase on disk. This can be done by copying my.cnf file with that variable already changed, because there's no instruction that allows us to edit a part of a file. So, the only option we have is to overwrite the file. Of course, we also must copy our script to create the database.

ADD directive is used to overwrite my.cnf file as mentioned before. The second parameter that this directive takes is the path for the file.

Step 2: Generate Docker Compose file

When we are done creating the Dockerfiles with every dependency, we need create a file called "docker-compose.yml". In this file, we specify all services our application needs and how they will interact between them and the application itself.

```yaml
version: '3.2'
services:
  mysql:
    build:
      context: ./DB
      dockerfile: Dockerfile
    container_name: acme_db
    restart: always
    networks:
      acme_network:
        aliases:
          - AcmePackageDB
    volumes:
      - acme_data:/var/lib/mysql
  tomcat:
    build:
      context: ./tomcat
      dockerfile: Dockerfile
    container_name: tomcat
    restart: always
    networks:
      acme_network:
        aliases:
          - tomcat
    ports:
      - target: 8080
        published: 80
        protocol: tcp
        mode: host
volumes:
  acme_data:

networks:
  acme_network:
    driver: bridge
```

3. Docker-compose.yml

Step 3: Change data.xml in our project

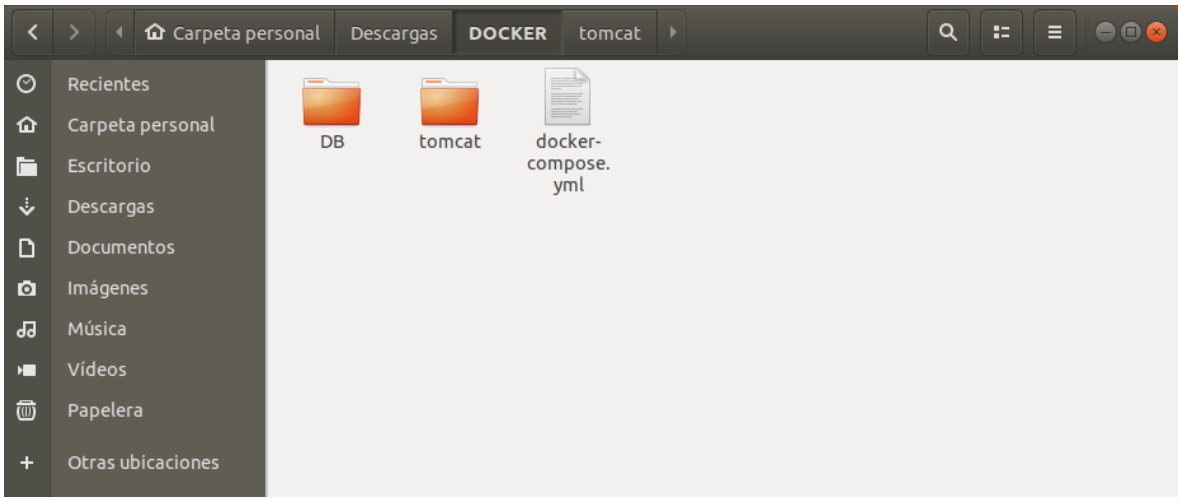
```xml
<bean id="dataSource" class="com.mchange.v2.c3p0.ComboPooledDataSource" destroy-method="close">
    <property name="driverClass" value="com.mysql.jdbc.Driver" />
    <property name="jdbcUrl" value="jdbc:mysql://AcmePackageDB:3306/Acme-Packages" /><!-- -->
    <property name="user" value="acme-user" />
    <property name="password" value="ACME-Us3r-P@ssw0rd" />
    <property name="minPoolSize" value="1" />
    <property name="maxPoolSize" value="5" />
</bean>
```

Step 4: Deployment with Docker

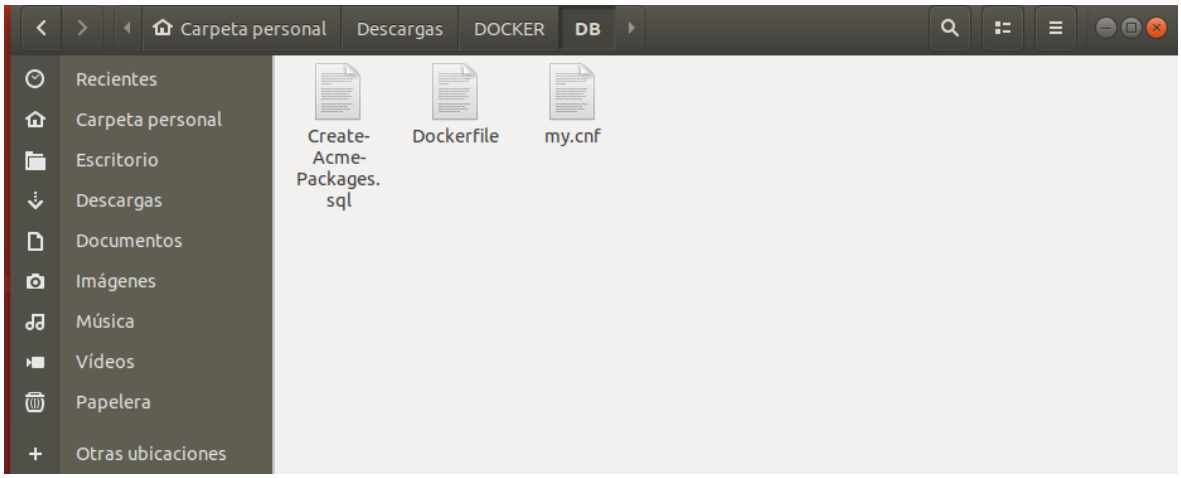
We will use an Ubuntu system to deploy our container.

Open a terminal in the folder where we have 2 folders and the docker-compose.yml

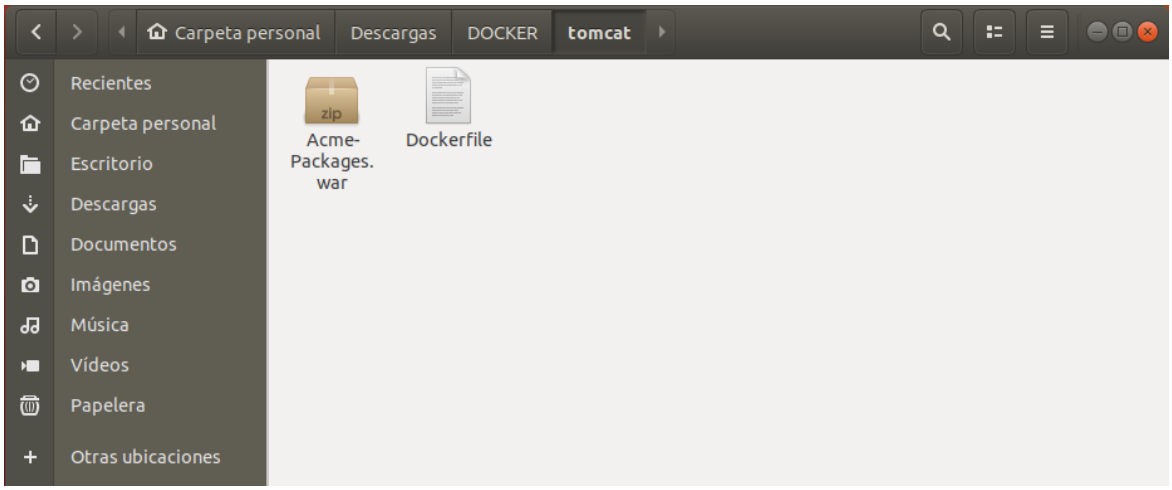
The folder will contain the following files and subfolders:



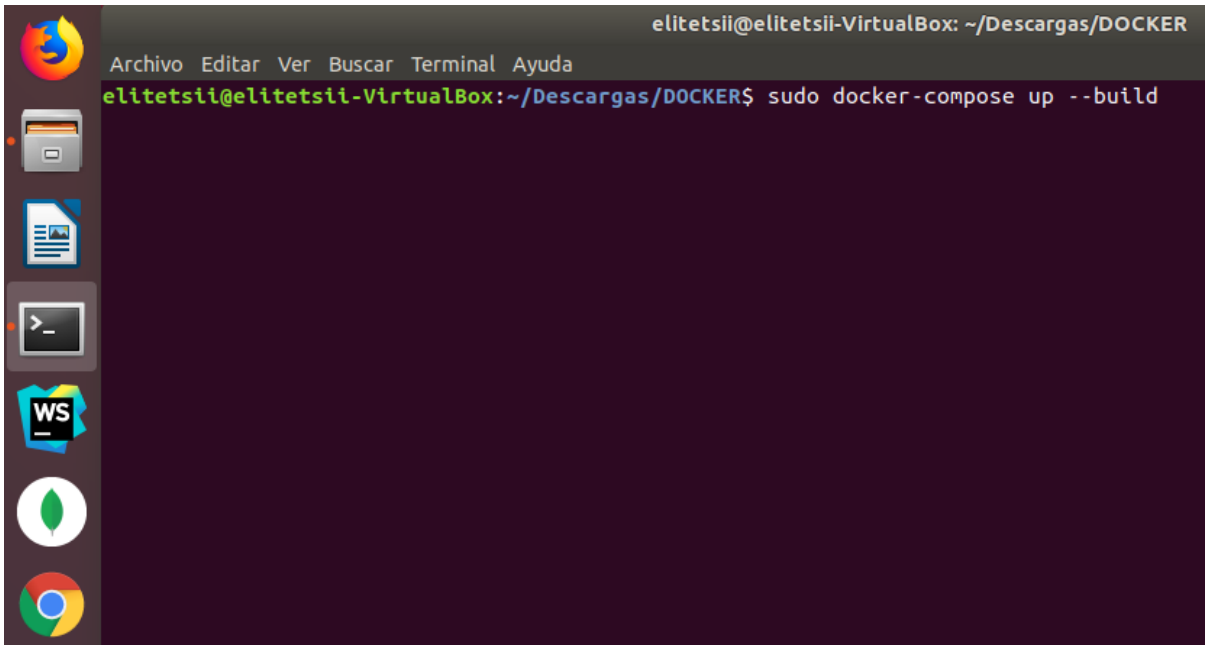
4. Folder where open the terminal



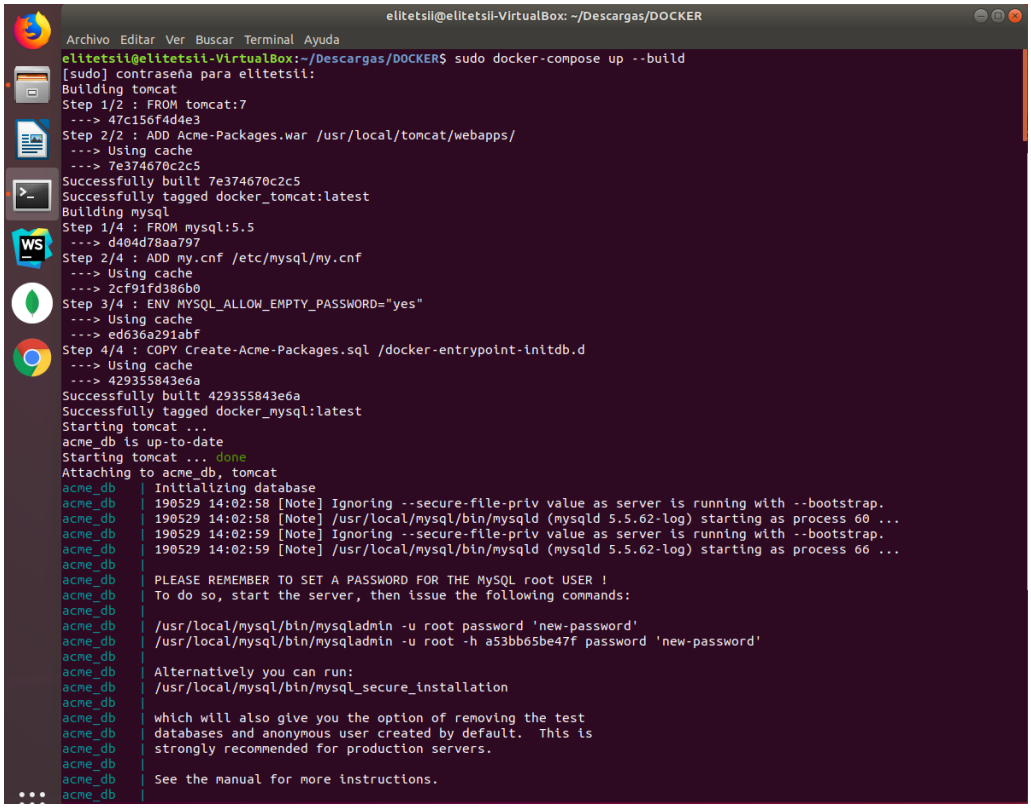
5. Folder DB



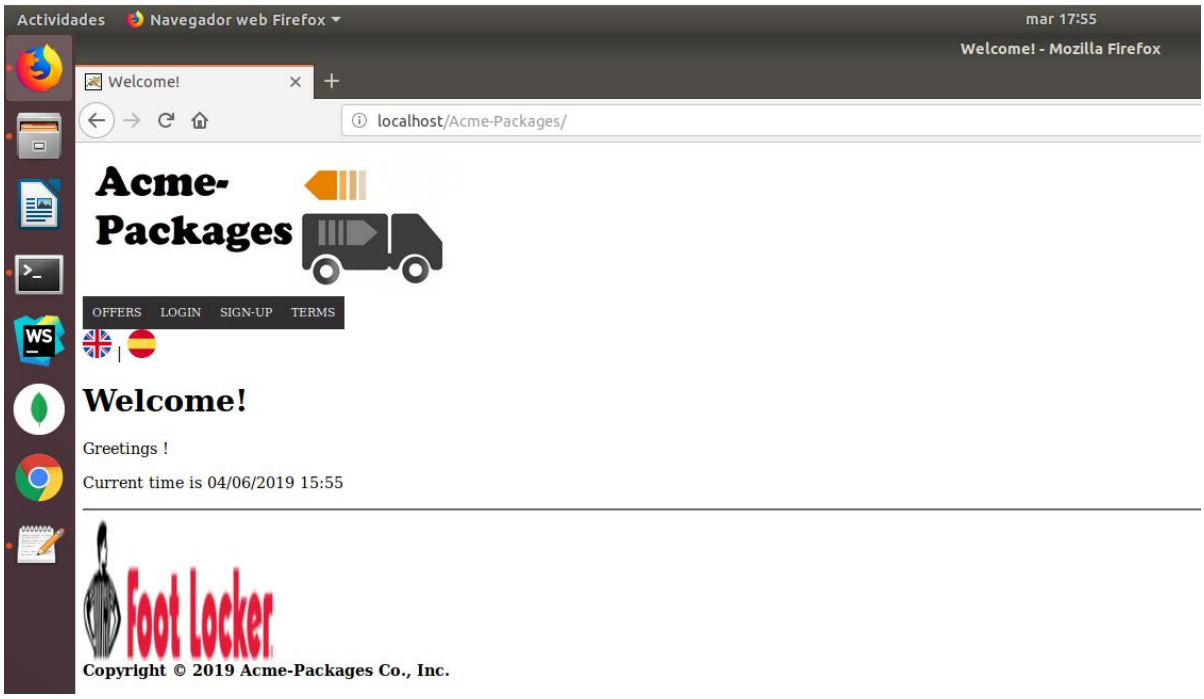
6. Folder tomcat



7. Execute "sudo docker-compose up --build"



8. Deployment in progress



9. Deployment completed