

# Realización de Cursos y Talleres

Manual de despliegue

Versión: v01r00

Fecha: 26-12-2023



Pablo Merinas Soto  
Manual de despliegue

## HOJA DE CONTROL

Título	Realización de Cursos y Talleres		
Entregable	Manual de despliegue		
Nombre del Fichero	UD02-Actividad Evaluable 01-Realización de Cursos y Talleres PMS.pdf		
Autor	Pablo Merinas Soto		
Versión/Edición	v01r00	Fecha Versión	26-12-2023
Aprobado por		Fecha Aprobación	DD/MM/AAAA
		Nº Total Páginas	023

## REGISTRO DE CAMBIOS

Versión	Causa del Cambio	Responsable del Cambio	Área	Fecha del Cambio

## CONTROL DE DISTRIBUCIÓN

Nombre y Apellidos	Cargo	Área	Nº Copias



## Indice

<b>Desplegar app Spring Boot con Docker.....</b>	<b>4</b>
<b>Montar tu WebApp de Angular en Docker.....</b>	<b>14</b>
<b>Despliega con Docker Compose tu app de Spring Boot, Angular y PostgreSQL.....</b>	<b>21</b>



# Desplegar app Spring Boot con Docker

Primero instalamos el IDE IntelliJ para trabajar siguiendo los pasos del curso

## ¡Gracias por descargar IntelliJ IDEA!

La descarga comenzará en breve. Si no es así, utilice el [enlace directo](#).

Descargue y verifique el checksum SHA-256 del archivo.  
Software de terceros utilizado por IntelliJ IDEA Ultimate Edition



## Preguntas frecuentes

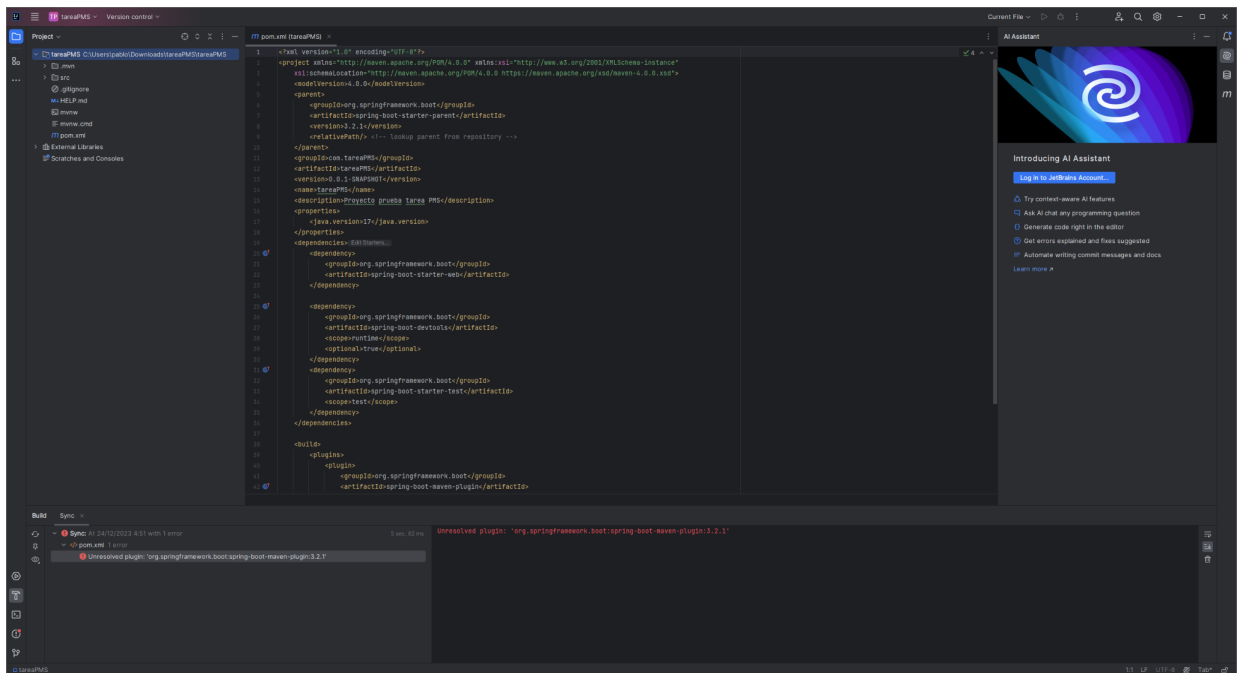
- + ¿La prueba es gratuita?
- + ¿Cómo puedo hacer que mi prueba sea más eficaz?
- + Quiero probar los productos de JetBrains en mi empresa, pero no puedo utilizar Internet desde nuestra red corporativa (trabajamos en un entorno seguro, los productos se evalúan offline, etc.). ¿Cómo puedo hacerlo?

Creamos el proyecto con spring initializr



The screenshot shows the Spring Initializr web application interface. The 'Project' section is configured with 'Language' set to 'Java' and 'Spring Boot' set to '3.2.2 (SNAPSHOT)'. The 'Project Metadata' section shows 'Group' as 'com.tareaPMS', 'Artifact' as 'tareaPMS', 'Name' as 'tareaPMS', 'Description' as 'Proyecto prueba tarea PMS', and 'Package name' as 'com.tareaPMS.tareaPMS'. The 'Packaging' is set to 'Jar' and 'Java' version is '17'. The 'Dependencies' section shows 'Spring Web' and 'Spring Boot Dev Tools' selected. A button 'ADD DEPENDENCIES... CTRL + B' is visible.

## Importo el proyecto a IntelliJ



## Creamos los controladores modelos al igual que el curso



```
1 package model;
2
3 public record Customer(Long id, String name) {
4 }
5
```

## Y el controlador REST

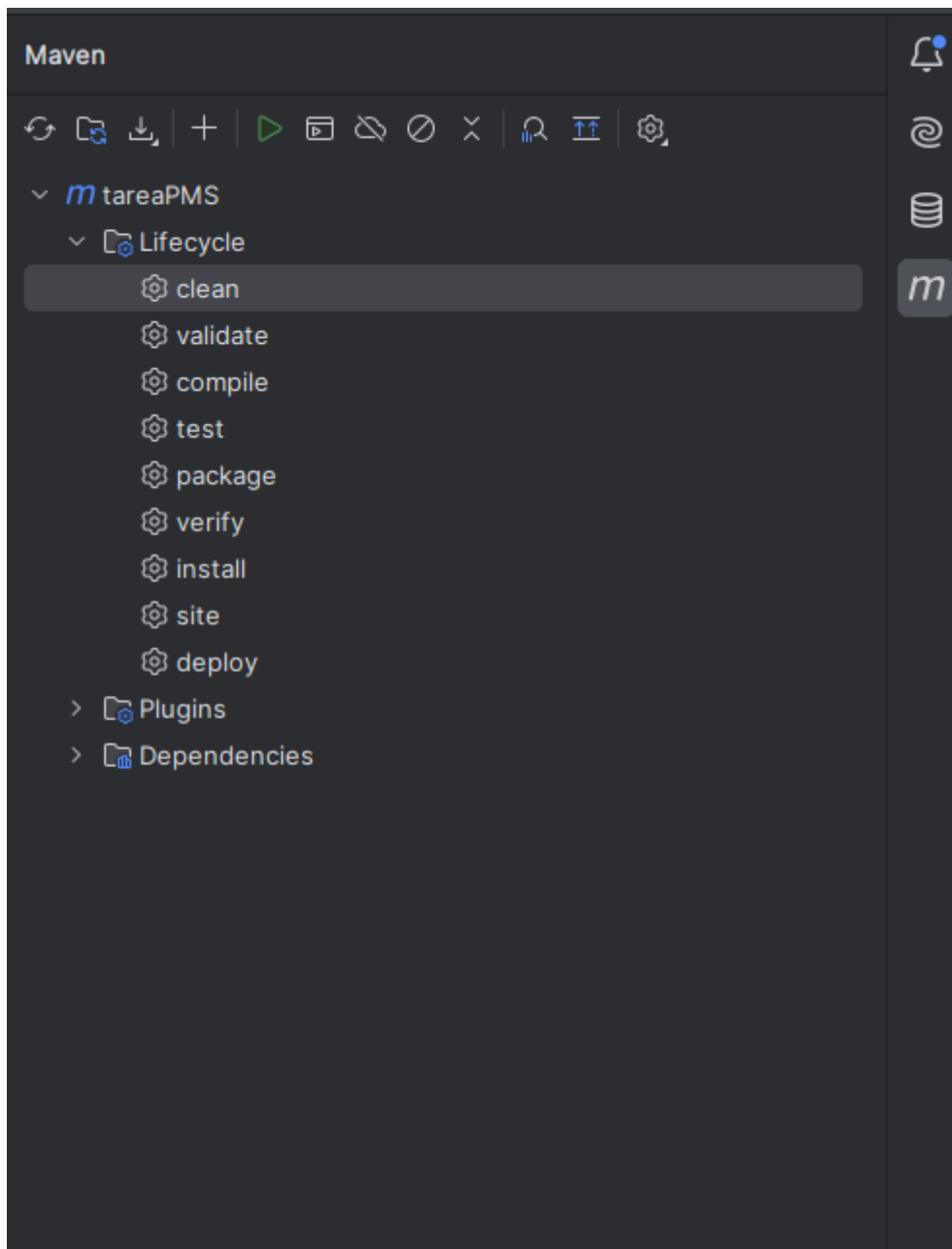
```
1 package controller;
2
3 import model.Customer;
4 import org.springframework.web.bind.annotation.GetMapping;
5 import org.springframework.web.bind.annotation.RequestMapping;
6 import org.springframework.web.bind.annotation.RestController;
7
8 @RestController
9 @RequestMapping("/api/customers")
10 public class CustomerController {
11
12     @GetMapping("/{id}")
13     public Customer findById(@PathVariable Long id){
14         return new Customer(id, name: "Jhon Doe Pablo Merinas");
15     }
16 }
17
18
```

Cambiamos la versión del proyecto a la 1.0



```
m pom.xml (tareaPMS) × TareaPmsApplication.java
1 <?xml version="1.0" encoding="UTF-8"?>
2 <project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema"
3   xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 https://maven.apache.org/xsd/maven
4   <modelVersion>4.0.0</modelVersion>
5   <parent>
6     <groupId>org.springframework.boot</groupId>
7     <artifactId>spring-boot-starter-parent</artifactId>
8     <version>3.2.1</version>
9     <relativePath/> <!-- lookup parent from repository -->
10  </parent>
11  <groupId>com.tareaPMS</groupId>
12  <artifactId>tareaPMS</artifactId>
13  <version>1.0</version>
14  <name>tareaPMS</name>
15  <description>Proyecto prueba tarea PMS</description>
16  <properties>
17    <java.version>17</java.version>
18  </properties>
19  <dependencies> Edit Starters...
20  <dependency>
21    <groupId>org.springframework.boot</groupId>
22    <artifactId>spring-boot-starter-web</artifactId>
23  </dependency>
```

Hacemos un clean con maven

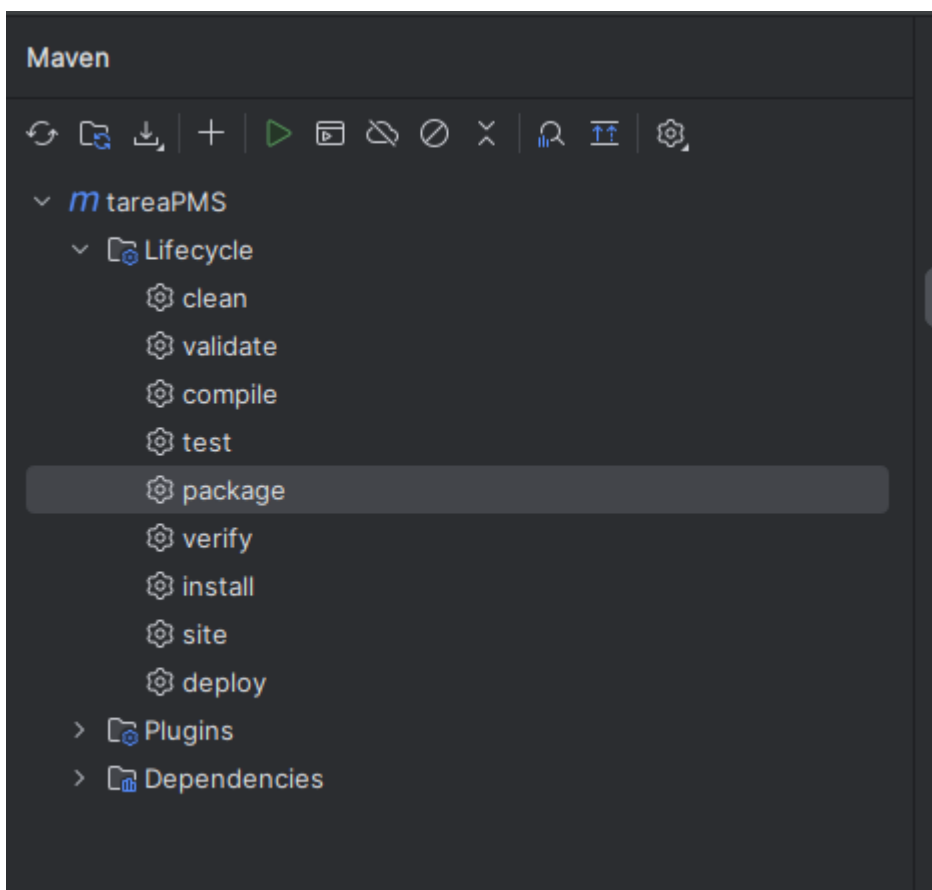






```
[INFO] --- clean:3.3.2:clean (default-clean) @ tareaPMS ---  
[INFO] Deleting C:\Users\pablo\Downloads\tareaPMS\tareaPMS\target  
[INFO] -----  
[INFO] BUILD SUCCESS  
[INFO] -----  
[INFO] Total time: 0.216 s  
[INFO] Finished at: 2023-12-24T05:01:08+01:00  
[INFO] -----  
  
Process finished with exit code 0
```

Con maven empaquetamos

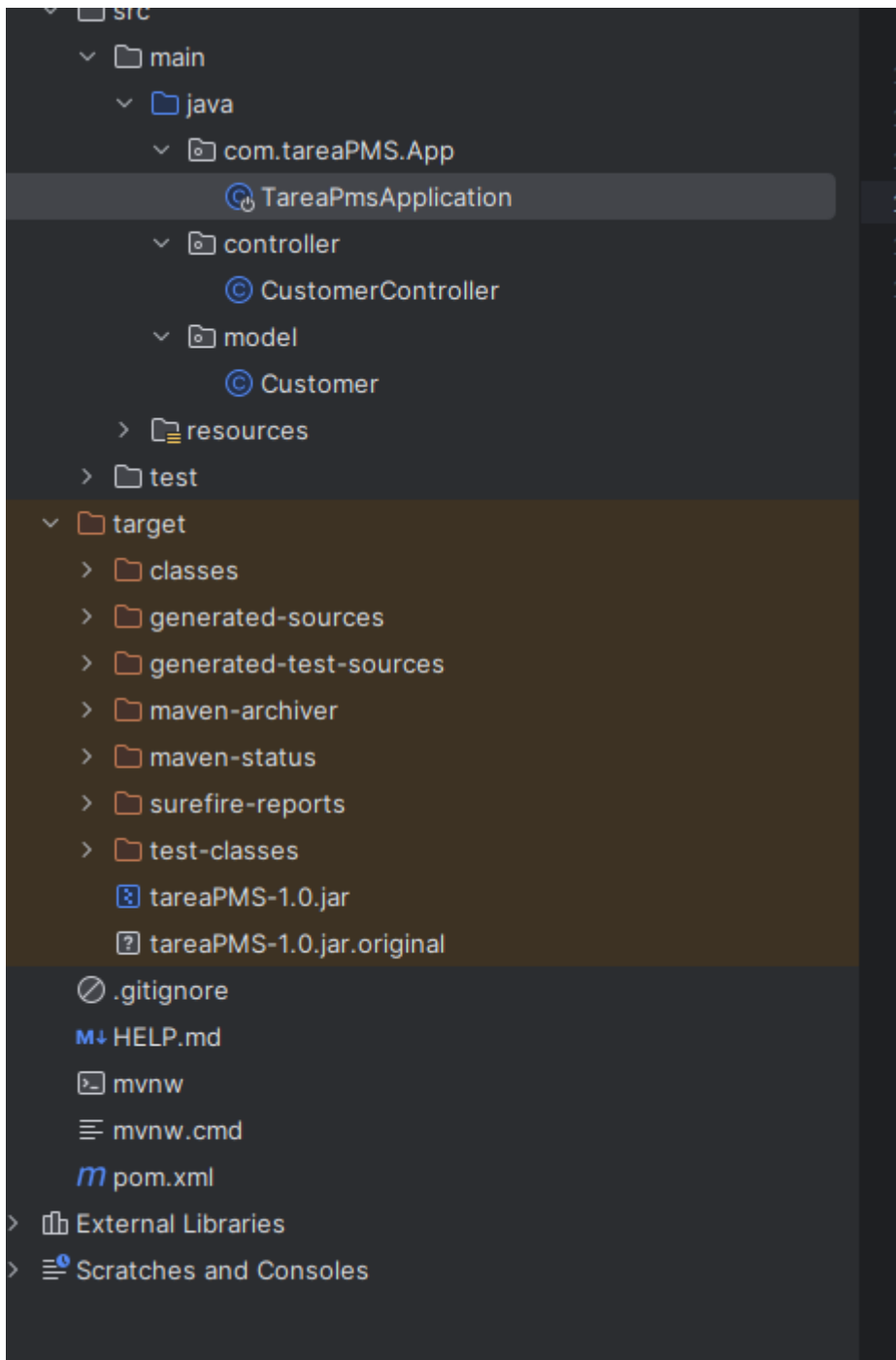




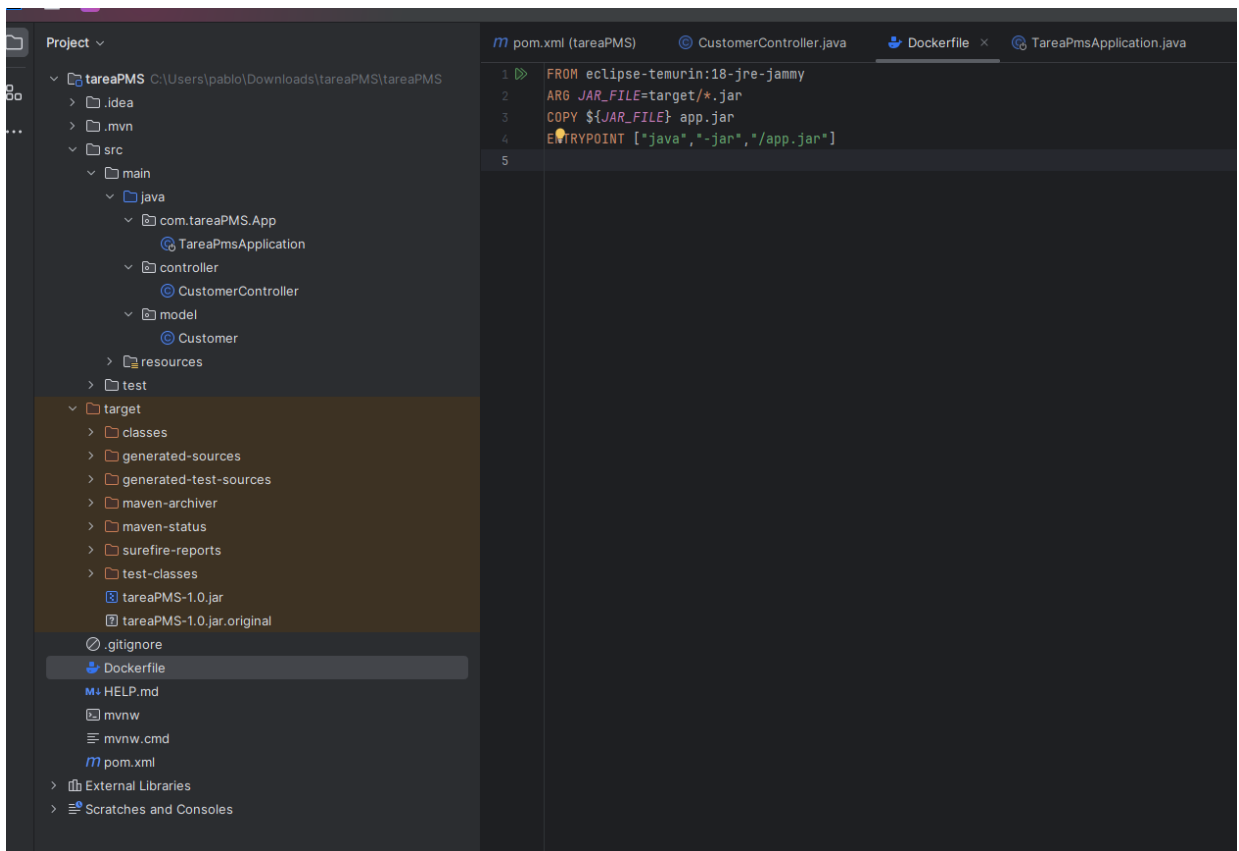
```
[INFO] Replacing main artifact C:\Users\pablo\Downloads\tareaPMS\tareaPMS\target\tareaPMS-1.0.jar with repackaged archive, adding nested dependencies in BOOT-INF/.
[INFO] The original artifact has been renamed to C:\Users\pablo\Downloads\tareaPMS\tareaPMS\target\tareaPMS-1.0.jar.original
[INFO] -----
[INFO] BUILD SUCCESS
[INFO] -----
[INFO] Total time: 3.201 s
[INFO] Finished at: 2023-12-24T05:01:30+01:00
[INFO] -----

Process finished with exit code 0
```

Para comprobar vemos que tenemos la carpeta target



Creamos el archivo Dockerfile con la configuración



Creamos la imagen de docker

```
PS C:\Users\pablo\Downloads\tareaPMS\tareaPMS> docker build -t springbootapp:1.0 .
2023/12/24 05:31:49 http2: server: error reading preface from client //./pipe/docker_engine: file has already been closed
[+] Building 1.3s (8/8) FINISHED
=> [internal] load .dockerignore
=> => transferring context: 2B

reaPMS > m pom.xml
```

Creo el contenedor usando esta imagen

```
PS C:\Users\pablo\Downloads\tareaPMS\tareaPMS> docker run -p 8080:8080 --name springapp -d -t springbootapp:1.0
9ed537a6e61658f68c444b217774ef6a46aae272efcafd3641b4b1ff61a21323
PS C:\Users\pablo\Downloads\tareaPMS\tareaPMS>

reaPMS > m pom.xml
```

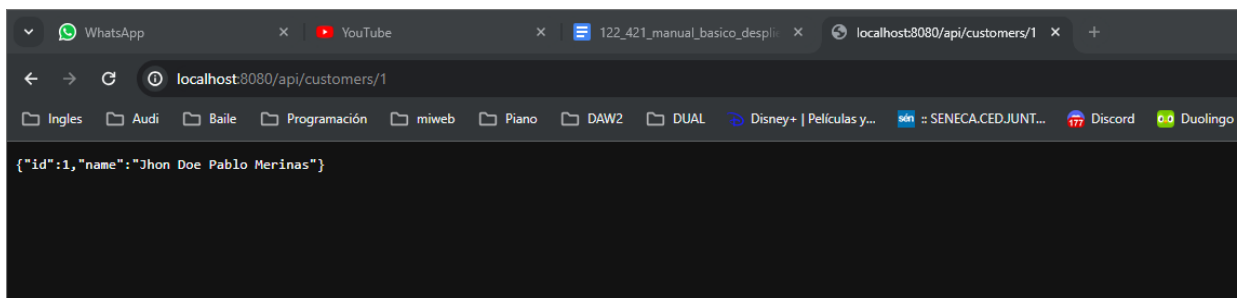
Comprobamos que se haya creado bien



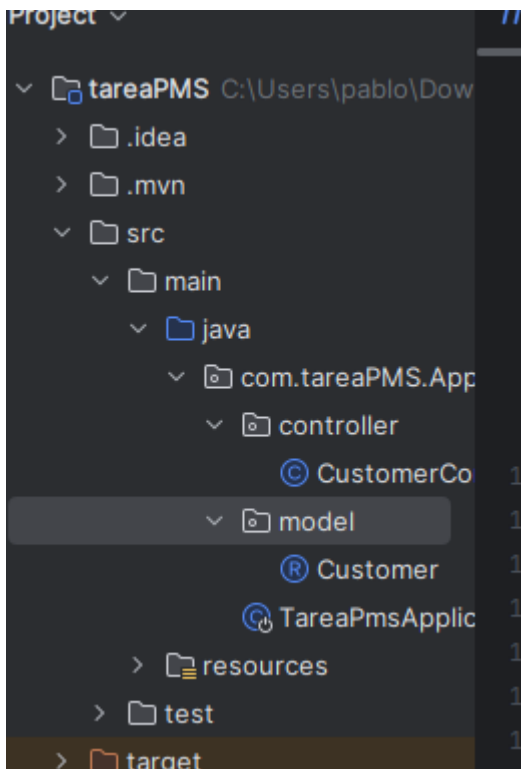
```
See 'docker --help'
PS C:\Users\pablo\Downloads\tareaPMS\tareaPMS> docker images
REPOSITORY          TAG         IMAGE ID      CREATED       SIZE
springbootapp       1.0        daf9b17af59d 43 seconds ago 286MB
PS C:\Users\pablo\Downloads\tareaPMS\tareaPMS> docker ps
CONTAINER ID   IMAGE          COMMAND                  CREATED        STATUS        PORTS                    NAMES
29115799bf59   springbootapp:1.0 "java -jar /app.jar"    35 seconds ago Up 33 seconds  0.0.0.0:8080->8080/tcp   springapp
PS C:\Users\pablo\Downloads\tareaPMS\tareaPMS>

reaPMS > m pom.xml
```

Comprobamos que se ejecute desde el navegador



Aquí tuve varios problemas que no me salía eso, al final tras investigar me di cuenta que había montado mal la estructura del proyecto, la corregí:



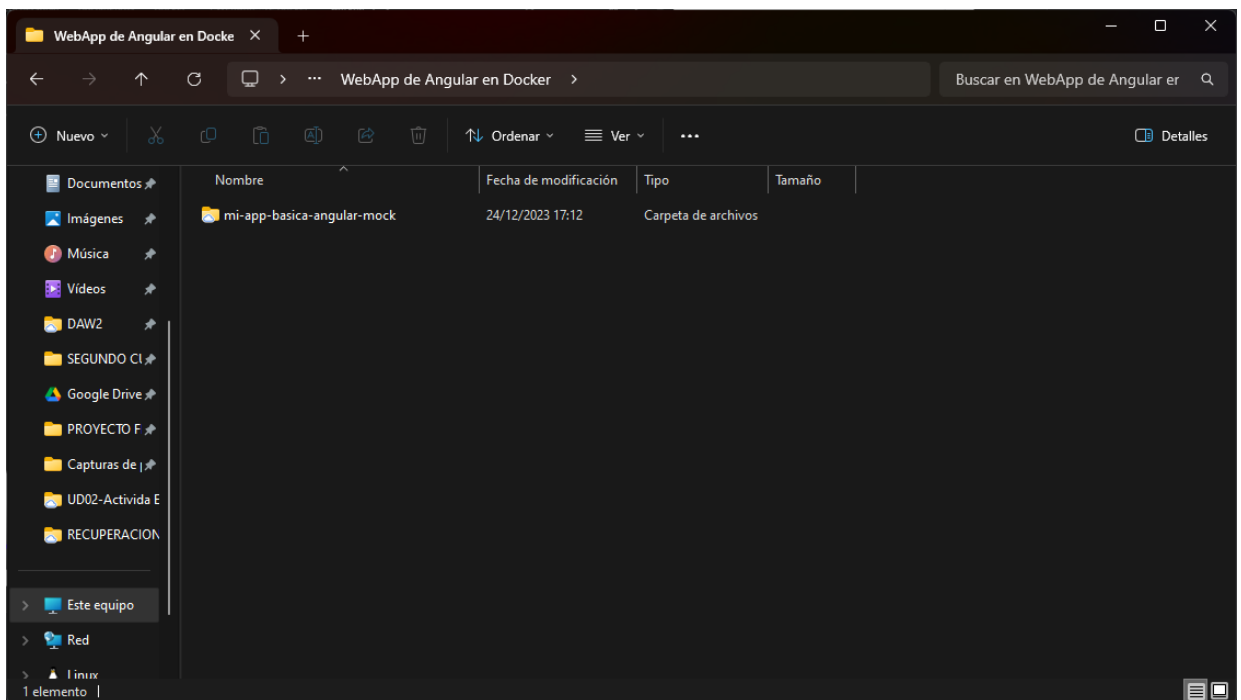
Volví a empaquetar y repetir todo el proceso y ya iba bien.



Y con esto ya queda desplegada la aplicación

## Montar tu WebApp de Angular en Docker

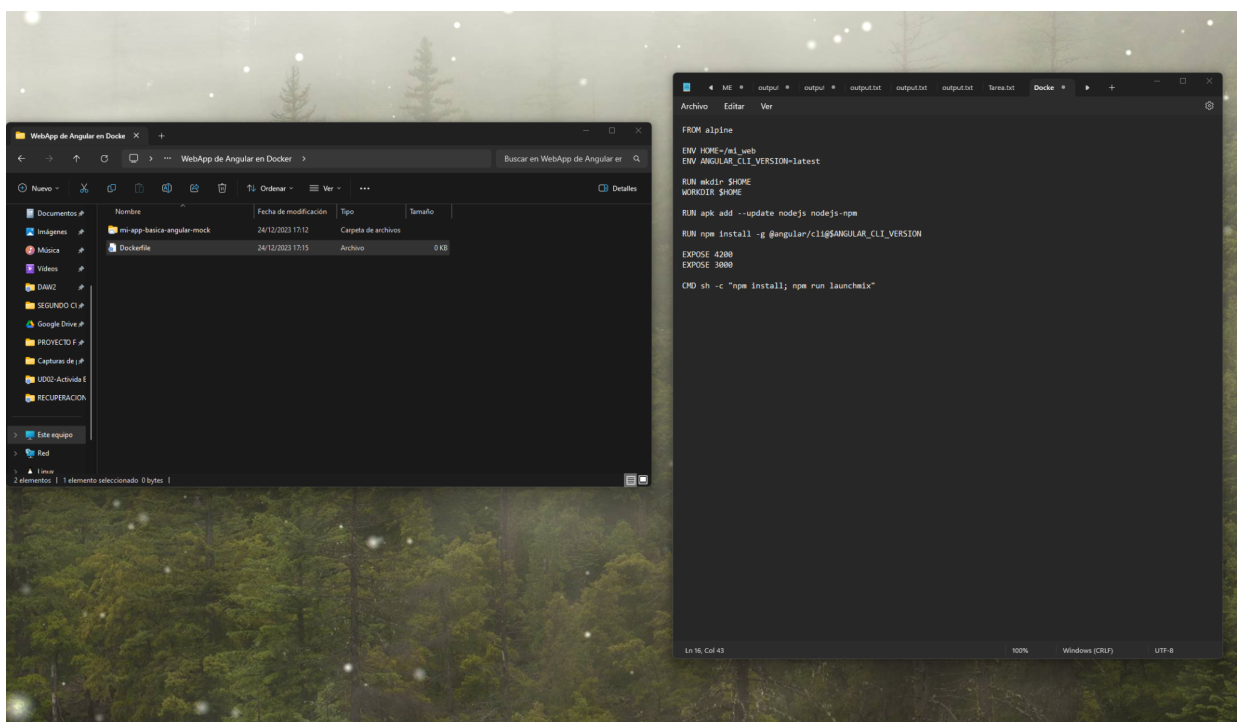
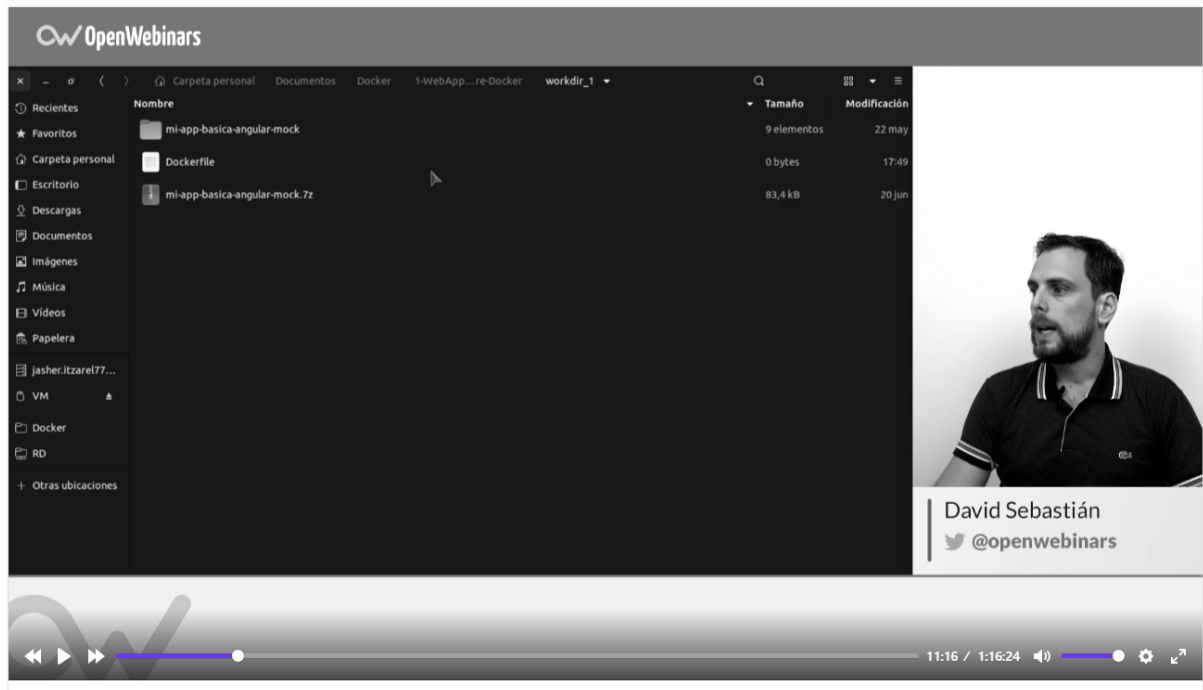
Primero descargamos el proyecto que facilita el curso a IntelliJ



Creamos el archivo dockerfile



## ← Cómo montar tu WebApp de Angular en Docker



Creamos la imagen de docker



```
C:\Windows\System32\cmd.e X + v
G:\Otros ordenadores\Mi portátil\DAW2\Despliegue de aplicaciones web\RECUPERACION\UD02-Activida Evaluable 01-Realización de Cursos y Talleres\WebApp de Angular en Docker>docker build -t example/existing_angular_app:0.0.1 .
[+] Building 2.0s (8/9)
=> [internal] load .dockerignore 0.0s
=> => transferring context: 2B 0.0s
=> [internal] load build definition from Dockerfile 0.0s
=> => transferring dockerfile: 308B 0.0s
=> [internal] load metadata for docker.io/library/alpine:latest 1.3s
=> [auth] library/alpine:pull token for registry-1.docker.io 0.0s
=> [1/5] FROM docker.io/library/alpine@sha256:51b67269f354137895d43f3b3d810bfacd3945438e94dc5ac55fdac340352f48 0.0s
=> CACHED [2/5] RUN mkdir /mi_web 0.0s
=> CACHED [3/5] WORKDIR /mi_web 0.0s
=> ERROR [4/5] RUN apk add --update nodejs nodejs-npm 0.7s
-----
> [4/5] RUN apk add --update nodejs nodejs-npm:
0.233 fetch https://dl-cdn.alpinelinux.org/alpine/v3.19/main/x86_64/APKINDEX.tar.gz
0.435 fetch https://dl-cdn.alpinelinux.org/alpine/v3.19/community/x86_64/APKINDEX.tar.gz
0.649 ERROR: unable to select packages:
0.649 nodejs-npm (no such package):
0.649   required by: world[nodejs-npm]
-----
Dockerfile:9
-----
7 | WORKDIR $HOME
8 |
9 | >>> RUN apk add --update nodejs nodejs-npm
10 |
11 | RUN npm install -g @angular/cli@$ANGULAR_CLI_VERSION
-----
ERROR: failed to solve: process "/bin/sh -c apk add --update nodejs nodejs-npm" did not complete successfully: exit code
: 1

G:\Otros ordenadores\Mi portátil\DAW2\Despliegue de aplicaciones web\RECUPERACION\UD02-Activida Evaluable 01-Realización de Cursos y Talleres\WebApp de Angular en Docker>
```

Aquí tuve el problema de que estaba usando windows, y el curso y comandos estaban hecho con linux, intente cambiar el Dockerfile para hacerlo compatible pero no fui capaz, así que instale una MV de linux para seguir con la tarea





```
*Dockerfile
~/Escritorio/WebApp Angular PMS

1 FROM alpine
2
3 ENV HOME=/mi_web
4 ENV ANGULAR_CLI_VERSION=latest
5
6 RUN mkdir $HOME
7 WORKDIR $HOME
8
9 RUN apk add --update nodejs
10 RUN apk add --update nodejs-npm
11
12 RUN npm install -g @angular/cli@$ANGULAR_CLI_VERSION
13
14 EXPOSE 4200
15 EXPOSE 3000
16
17 CMD sh -c "npm install; npm run launchmix"
```

Ejecutamos el comando para crear el docker



```
root@pablo-VirtualBox: /home/pablo/Escritorio/WebApp Angular PMS# docker build -t
example/existing_angular_app:0.0.1 .
[+] Building 15.4s (6/7)                                docker:default
=> [internal] load build definition from Dockerfile      0.1s
=> => transferring dockerfile: 264B                      0.0s
=> [internal] load .dockerignore                        0.0s
=> => transferring context: 2B                            0.0s
=> [internal] load metadata for docker.io/library/node:14-alpine 1.4s
=> [1/4] FROM docker.io/library/node:14-alpine@sha256:434215b487a329c9e8 2.4s
=> => resolve docker.io/library/node:14-alpine@sha256:434215b487a329c9e8 0.0s
=> => sha256:434215b487a329c9e867202ff89e704d3a75e554822 1.43kB / 1.43kB 0.0s
=> => sha256:4e84c956cd276af9ed14a8b2939a734364c2b004248 1.16kB / 1.16kB 0.0s
=> => sha256:0dac3dc27b1ad570e6c3a7f7cd29e88e7130ff0cad3 6.44kB / 6.44kB 0.0s
=> => sha256:f56be85fc22e46face30e2c3de3f7fe7c15f8fd7c4e 3.37MB / 3.37MB 0.3s
=> => sha256:8f665685b215c7daf9164545f1bbdd74d800af77d 37.17MB / 37.17MB 0.9s
=> => sha256:e5fca6c395a62ec277102af9e5283f6edb43b3e4f20 2.37MB / 2.37MB 0.5s
=> => extracting sha256:f56be85fc22e46face30e2c3de3f7fe7c15f8fd7c4e5add2 0.2s
=> => sha256:561cb69653d56a9725be56e02128e4e96fb434a8b4b4dec 448B / 448B 0.5s
=> => extracting sha256:8f665685b215c7daf9164545f1bbdd74d800af77d0d267db 1.1s
=> => extracting sha256:e5fca6c395a62ec277102af9e5283f6edb43b3e4f20f798e 0.1s
=> => extracting sha256:561cb69653d56a9725be56e02128e4e96fb434a8b4b4decf 0.0s
=> [2/4] RUN mkdir /mi_web                             0.3s
=> [3/4] WORKDIR /mi_web                               0.1s
=> [4/4] RUN npm install -g @angular/cli@latest        11.0s
```

Comprobamos

```
root@pablo-VirtualBox: /home/pablo/Escritorio/WebApp Angular PMS# docker images
REPOSITORY          TAG          IMAGE ID          CREATED          SIZE
example/existing_angular_app 0.0.1       fc3ad542ad69     16 seconds ago  169MB
root@pablo-VirtualBox: /home/pablo/Escritorio/WebApp Angular PMS#
```

Creamos la imagen basada en este contenedor

```
root@pablo-VirtualBox: /home/pablo/Escritorio/WebApp Angular PMS# docker images |
grep example
example/existing_angular_app 0.0.1       fc3ad542ad69     5 minutes ago  169MB
root@pablo-VirtualBox: /home/pablo/Escritorio/WebApp Angular PMS#
```



```
root@pablo-VirtualBox: /home/pablo/Escritorio/webapppms
root@pablo-VirtualBox:/home/pablo/Escritorio/webapppms# docker run -d --name mi_
existing_angular_app -v $(pwd)/mi-app-basica-angular-mock:/mi_web -p 4200:4200 -
p 3000:3000 example/existing_angular_app:0.0.1
20119715b8762b4a86a5f728e7e317cd2c33c3623ab3e5cb7a7a73384eeac1dd
root@pablo-VirtualBox:/home/pablo/Escritorio/webapppms#
```

Aquí tuve otro problema, el nombre de mi carpeta anterior era WebApp Angular PMS, y al poner \$(pwd) para que pusiera la ruta, no me lo generaba por los espacios del nombre de la carpeta, al darme cuenta cambie el nombre a webapppms sin espacios y ya funcionó sin problema.

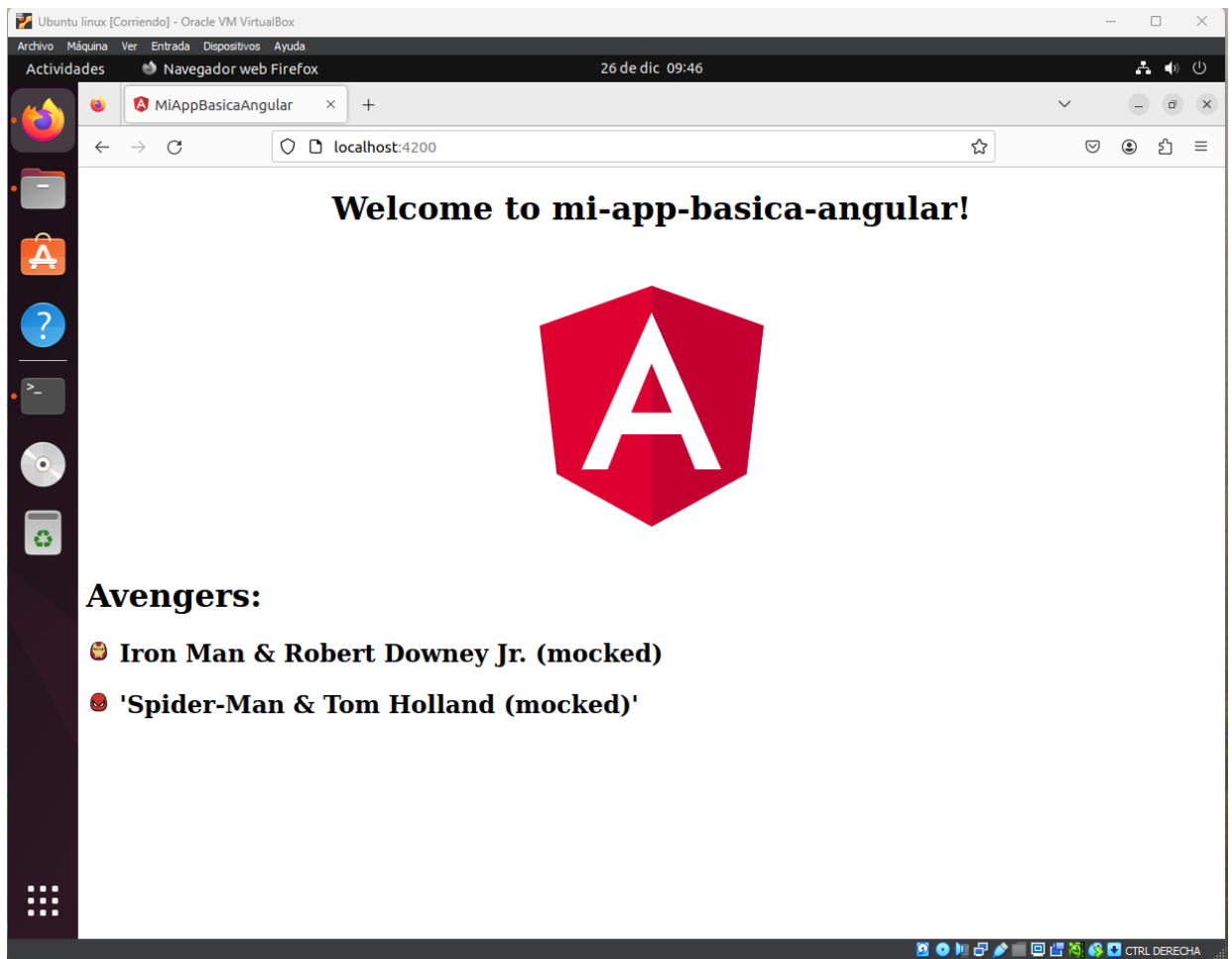
Y con un docker ps comprobamos su estado

```
root@pablo-VirtualBox:/home/pablo/Escritorio/webapppms# docker ps
CONTAINER ID   IMAGE                                COMMAND                  CREATED
STATUS        PORTS          NAMES
20119715b876   example/existing_angular_app:0.0.1  "docker-entrypoint.s..."  About a minute ago
Up About a minute   0.0.0.0:3000->3000/tcp, :::3000->3000/tcp,
0.0.0.0:4200->4200/tcp, :::4200->4200/tcp  mi_existing_angular_app
root@pablo-VirtualBox:/home/pablo/Escritorio/webapppms#
```

Por último comprobamos que la aplicación se haya desplegado correctamente.



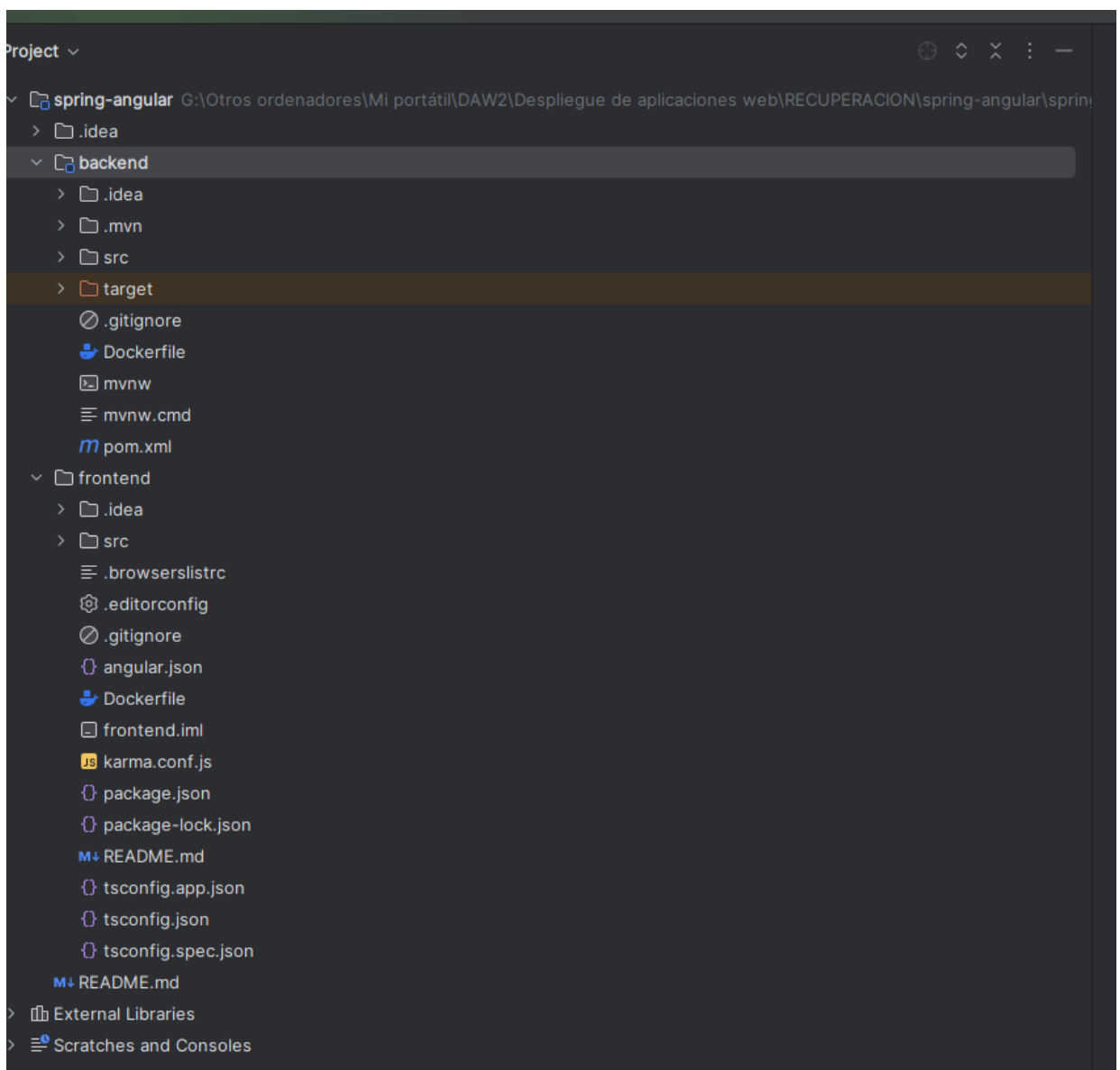
Pablo Merinas Soto  
Manual de despliegue





# Despliega con Docker Compose tu app de Spring Boot, Angular y PostgreSQL

Descargo el proyecto del curso y lo importo a IntelliJ y hacemos un clean y package del proyecto



Creo los Dockerfile del frontend y del backend ( Cada uno en su correspondiente carpeta de proyecto )



## Backend

```
docker-compose.yml  backend\Dockerfile  frontend\Dockerfile
1  FROM eclipse-temurin:18-jre-jammy
2
3  ARG JAR_FILE=target/*.jar
4
5  COPY ${JAR_FILE} app.jar
6
7  ENTRYPOINT ["java", "-jar", "/app.jar"]
```

## Frontend

```
docker-compose.yml  backend\Dockerfile  frontend\Dockerfile
1  FROM node:16.16.0 as build-step
2
3  RUN mkdir -p /usr/local/app
4
5  WORKDIR /usr/local/app
6
7  COPY ./ /usr/local/app
8
9  RUN npm install
10
11  RUN npm run build --prod
12
13  FROM nginx:1.23.1
14
15  RUN rm -rf /usr/share/nginx/html/*
16
17  COPY --from=build-step /usr/local/app/dist/frontend /usr/share/nginx/html
18
19  EXPOSE 80
20
```

Creo las imágenes del frontend y del backend ( Cada uno en su correspondiente carpeta de proyecto )

## Backend



```
PS G:\Otros ordenadores\Mi portátil\DAW2\Despliegue de aplicaciones web\RECUPERACION\spring-angular\spring-angular\backend> docker build -t backend:1.0 .
2023/12/26 11:03:08 http2: server: error reading preface from client //./pipe/docker_engine: file has already been closed
[+] Building 1.0s (8/8) FINISHED
=> [internal] load .dockerignore
=> => transferring context: 2B
=> [internal] load build definition from Dockerfile
=> => transferring dockerfile: 171B
=> [internal] load metadata for docker.io/library/eclipse-temurin:18-jre-jammy
=> [auth] library/eclipse-temurin:pull token for registry-1.docker.io
=> [internal] load build context
=> => transferring context: 3.32kB
=> [1/2] FROM docker.io/library/eclipse-temurin:18-jre-jammy@sha256:55373f54aacc86568b8427e7c1d5719e2ad08c2f66b66c2b86071cd82efd1632
=> CACHED [2/2] COPY app.jar
=> exporting to image
=> => exporting layers
=> => writing image sha256:1b2103d219772b6958b27b23e6f56bac1f76099b44ec9fd4bc11f1796b1a25d8
=> => naming to docker.io/library/backend:1.0

What's Next?
View a summary of image vulnerabilities and recommendations → docker scout quickview
PS G:\Otros ordenadores\Mi portátil\DAW2\Despliegue de aplicaciones web\RECUPERACION\spring-angular\spring-angular\backend>
spring-angular > docker-compose.yml
```

## Frontend

```
PS G:\Otros ordenadores\Mi portátil\DAW2\Despliegue de aplicaciones web\RECUPERACION\spring-angular\spring-angular\frontend> docker build -t frontend:1.0 .
2023/12/26 11:11:47 http2: server: error reading preface from client //./pipe/docker_engine: file has already been closed
[+] Building 24.7s (15/15) FINISHED
=> [internal] load build definition from Dockerfile
=> => transferring dockerfile: 352B
=> [internal] load .dockerignore
=> => transferring context: 2B
=> [internal] load metadata for docker.io/library/nginx:1.23.1
=> [internal] load metadata for docker.io/library/node:16.16.0
=> [build-step 1/6] FROM docker.io/library/node:16.16.0@sha256:1ed1e17ccabb09038cfb8a965337ebcda51ef9e9d32082164c502d44d9731a02
=> [internal] load build context
=> => transferring context: 2.20kB
=> [stage-1 1/3] FROM docker.io/library/nginx:1.23.1@sha256:2f77bd2fe27bc85f68fd7fe6a63900ef7076bc703022fe81b980377fe3d27b70
=> CACHED [stage-1 2/3] RUN rm -rf /usr/share/nginx/html/*
=> CACHED [build-step 2/6] RUN mkdir -p /usr/local/app
=> CACHED [build-step 3/6] WORKDIR /usr/local/app
=> [build-step 4/6] COPY ./ /usr/local/app
=> [build-step 5/6] RUN npm install
=> [build-step 6/6] RUN npm run build --prod
=> [stage-1 3/3] COPY --from=build-step /usr/local/app/dist/frontend /usr/share/nginx/html
=> exporting to image
=> => exporting layers
=> => writing image sha256:8ae3e89f2d7272d20cf382dd1fe343475749b8bda9efae28aa09d5e98274dbce
=> => naming to docker.io/library/frontend:1.0

What's Next?
View a summary of image vulnerabilities and recommendations → docker scout quickview
PS G:\Otros ordenadores\Mi portátil\DAW2\Despliegue de aplicaciones web\RECUPERACION\spring-angular\spring-angular\frontend>
```

## Comprobamos

```
View a summary of image vulnerabilities and recommendations → docker scout quickview
PS G:\Otros ordenadores\Mi portátil\DAW2\Despliegue de aplicaciones web\RECUPERACION\spring-angular\spring-angular\backend> docker images
REPOSITORY    TAG       IMAGE ID       CREATED        SIZE
backend       1.0      1b2103d21977   9 minutes ago  284MB
frontend      1.0      1b2103d21977   9 minutes ago  284MB
PS G:\Otros ordenadores\Mi portátil\DAW2\Despliegue de aplicaciones web\RECUPERACION\spring-angular\spring-angular\backend>
```

Creo el docker-compose y ejecuto el comando para levantarlo



The screenshot shows an IDE with a project named 'spring-angular'. The left sidebar displays the project structure, including 'backend' and 'frontend' folders. The main editor area shows the 'docker-compose.yml' file with the following configuration:

```
services:
  postgres:
    image: postgres:14.5
    container_name: postgres
    environment:
      - POSTGRES_USER=postgres
      - POSTGRES_PASSWORD=admin
      - POSTGRES_DB=spring_db
    ports:
      - 5432:5432
  backend:
    image: backend:1.0
    container_name: spring-backend
    ports:
      - 8080:8080
    depends_on:
      - postgres
  frontend:
    image: frontend:1.0
    container_name: spring-frontend
    ports:
      - 80:80
    depends_on:
      - backend
```

The terminal at the bottom shows the execution of 'docker-compose up -d' and 'docker ps'. The output indicates that the containers are successfully started and running.

```
PS 6:\Otros ordenadores\Mi portátil\DAW2\Despliegue de aplicaciones web\RECUPERACION\spring-angular\spring-angular> docker-compose up -d
[+] Running 14/1
  ✓ postgres 13 layers [#####] 0B/0B Pulled
[+] Running 4/4
  ✓ Network spring-angular_default Created
  ✓ Container postgres Started
  ✓ Container spring-backend Started
  ✓ Container spring-frontend Started
PS 6:\Otros ordenadores\Mi portátil\DAW2\Despliegue de aplicaciones web\RECUPERACION\spring-angular\spring-angular> docker ps
CONTAINER ID   IMAGE          COMMAND                  CREATED        STATUS        PORTS                               NAMES
76532ab4b16a   frontend:1.0   "/docker-entrypoint...." 8 seconds ago  Up 7 seconds  0.0.0.0:80->80/tcp                 spring-frontend
0257a5e353c9   backend:1.0    "java -jar /app.jar"      8 seconds ago  Up 7 seconds  0.0.0.0:8080->8080/tcp             spring-backend
74d7bf820981   postgres:14.5  "docker-entrypoint.s..." 9 seconds ago  Up 7 seconds  0.0.0.0:5432->5432/tcp             postgres
PS 6:\Otros ordenadores\Mi portátil\DAW2\Despliegue de aplicaciones web\RECUPERACION\spring-angular\spring-angular>
```

Comprobamos que se haya desplegado:





Pablo Merinas Soto  
Manual de despliegue

