**FORMATO DE DOCUMENTACIÓN**

|  |  |  |  |
| --- | --- | --- | --- |
| **Fecha:** | 2023-10-21 | **Grupo:** | Semillero Lab#1 |
| **Proyecto:** | DepGirPro v1 | **Squad:** | DevOps |
| **Elaboró:** | Squad de DevOps | **Aprobó:** |  |
| **Tipo:** | manual | | |
| **Descripción:** | Desplegar una aplicación de spring boot en Docker usando Linux | | |

**Instalación de java**

Primero, es importante actualizar el sistema antes de comenzar a instalar las herramientas, para ello, se ejecutan los siguientes comandos:

**sudo apt update**

**sudo apt upgrade**

Ahora, procederemos a agregar el repositorio de openjdk-17 para poder instalar el jdk-17 de java, ejecutamos los siguientes comandos:

**sudo add-apt-repository ppa:openjdk-r/ppa**

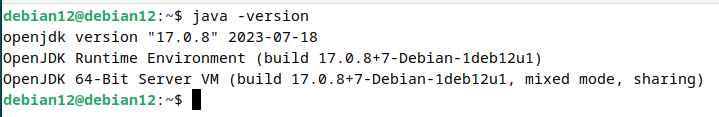
**sudo apt update**

Después, instalaremos java 17 ejecutando el siguiente comando:

**sudo apt install openjdk-17-jdk**

Finalmente, podemos verificar la versión de java para verificar su instalación, usamos el comando:

**java -version**



Una vez instalado java, instalaremos el gestor de dependencias de Maven para poder trabajar de forma mas optimizada los proyectos de spring, ejecutamos el comando:

**sudo apt install maven**

verificamos la instalación revisando la versión:

**mvn -version**

**Descargar IntelliJ IDEA – IDE de JAVA**

Vamos a instalar un IDE de java (IntelliJ) para poder comenzar a desarrollar directamente desde Linux, nos vamos al siguiente link en el navegador: <https://www.jetbrains.com/idea/download/?section=linux>

Descargaremos la versión Community, es decir, la versión gratuita:

Texto

Descripción generada automáticamente

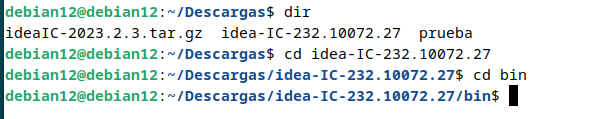
Daremos click en Download para descargar el .tar.gz

Una vez descargado y almacenado en la carpeta de Descargas, descomprimiremos el tar usando el siguiente comando:

**tar -xzf nombre\_del\_archivo.tar.gz**

Debemos estar en la carpeta descargas o donde se haya descargado el archivo.

Una vez descomprimido, solo queda iniciar IntelliJ, en la carpeta que nos quedó se encuentra el directorio bin, nos dirigiremos a ese directorio usando cd, por ejemplo:



Una vez dentro del directorio bin, iniciamos el IDE ejecutando el comando:

**./idea.sh**

IntelliJ iniciará y podremos comenzar a trabajar con un proyecto de Spring normalmente, podemos generar la plantilla usando Spring Initializr a través de la url <https://start.spring.io/>, descargamos el zip de la misma y lo descomprimimos a través del comando:

**unzip mi\_proyecto.zip**

tener en cuenta que “mi\_proyecto.zip” debe ser cambiado por el nombre que le dimos al proyecto cuando lo generamos en spring.io.

una vez descomprimido, en el IntelliJ podremos abrirlo y trabajarlo como un proyecto normal de Spring Boot, por ejemplo:

Interfaz de usuario gráfica, Texto

Descripción generada automáticamente

Una vez creado un proyecto simple en Spring boot, vamos a guardarlo con ctrol-s y vamos a instalar Docker:

Para ello, ejecutamos los siguientes comandos en orden:

**$ sudo apt install apt-transport-https ca-certificates curl gnupg-agent software-properties-common**

**$ curl -fsSL https://download.docker.com/linux/debian/gpg | sudo gpg --dearmor -o /usr/share/keyrings/docker-archive-keyring.gpg**

**$ echo "deb [signed-by=/usr/share/keyrings/docker-archive-keyring.gpg] https://download.docker.com/linux/debian $(lsb\_release -cs) stable" | sudo tee /etc/apt/sources.list.d/docker.list**

**$ sudo apt update**

**$ sudo apt install docker-ce docker-ce-cli containerd.io**

**$ docker –version**

Este último comando permite verificar la versión de Docker que ha sido instalada.

El siguiente paso es añadir un usuario en Docker para permitir tener permisos al ejecutar los comandos esenciales de Docker, ejecutamos la siguiente línea:

**sudo usermod -aG docker $(whoami)**

**Importante:** cerrar la consola, reiniciar y volver a ingresar, en caso de que se esté utilizando una VM, apagarla y volverla a encender.

**Dockerizar una aplicación Spring boot**

Volvemos a ingresar al IDE de intelliJ donde ya teníamos el proyecto de prueba realizado de Spring boot, y vamos a proceder a crear una imagen para desplegarlo desde Docker.

En la carpeta del proyecto, desde el IntelliJ IDEA crearemos un nuevo archivo llamado “Dockerfile” exactamente igual el nombre. Su contenido debe ser el siguiente:

**FROM openjdk:17**

**VOLUME /tmp**

**EXPOSE 8080**

**ADD ./target/project-name-0.0.1-SNAPSHOT.jar app.jar**

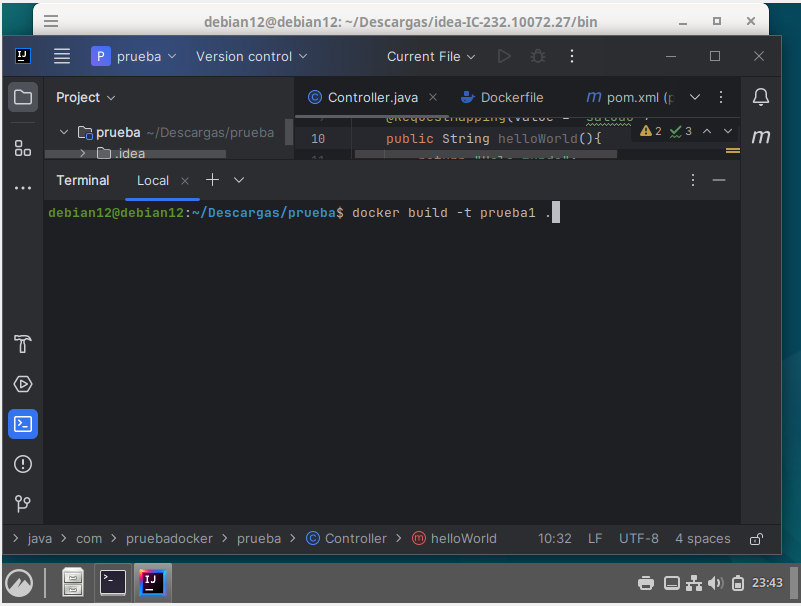
**ENTRYPOINT ["java", "-jar", "/app.jar"]**

Reemplazar la parte que dice “project-name” por el nombre de su proyecto de Spring sobre el cual están trabajando.

Una vez tengamos el Dockerfile, vamos a abrir una nueva consola de comandos en IntelliJ para que se abra justo en el directorio raíz del proyecto, a la altura del Dockerfile, y vamos a ejecutar el siguiente comando:

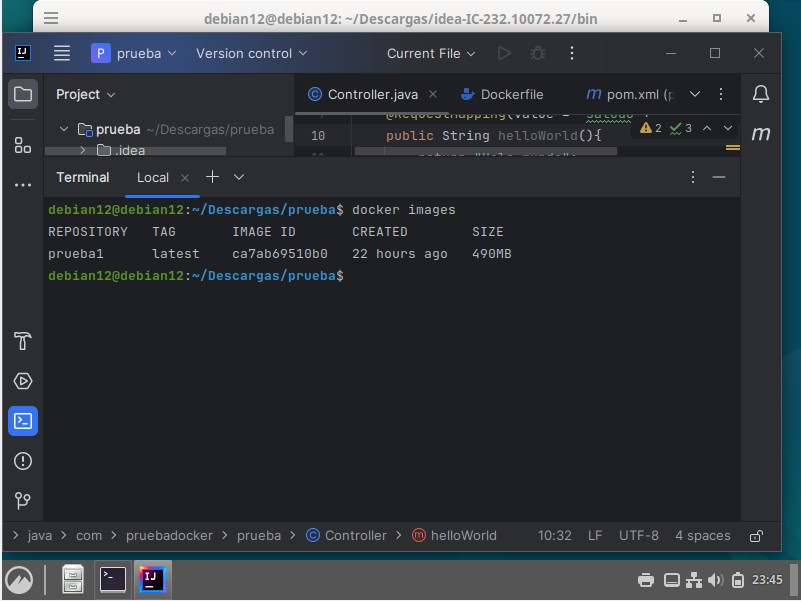
**docker build -t nombre\_de\_la\_imagen .**

Reemplazar “nombre\_de\_la\_imagen” por el nombre que deseemos darle a la imagen de Docker, por ejemplo:



Esperaremos a que Docker cree la imagen de nuestra aplicación, al finalizar podemos confirmar que realmente se ha creado correctamente a través del comando:

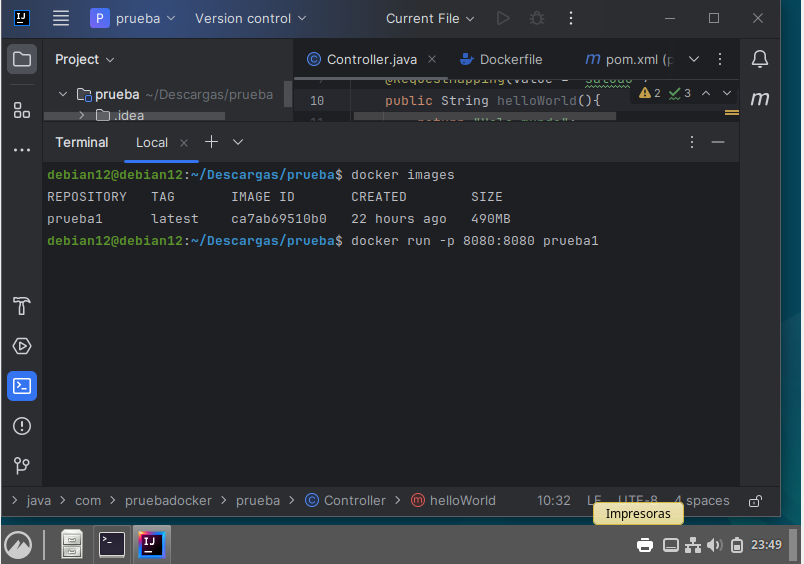
**docker images**



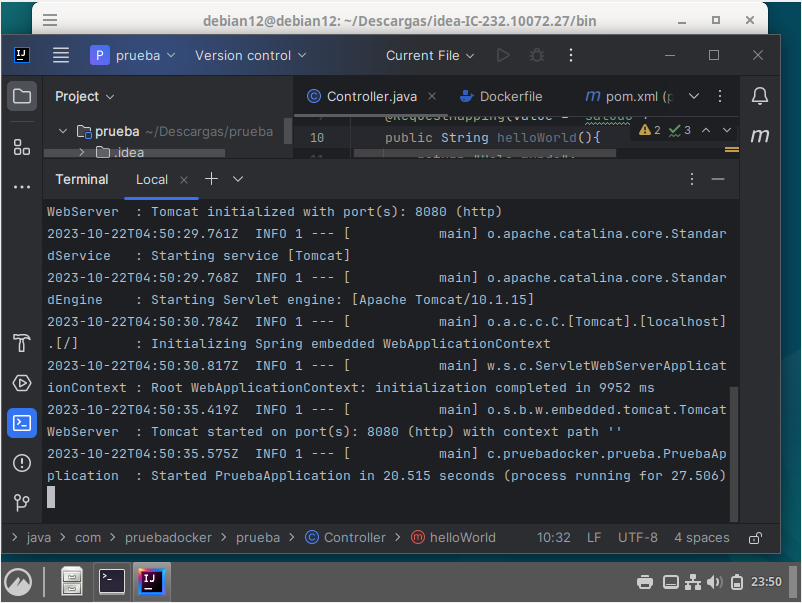
Al confirmar que se ha creado correctamente nuestra imagen, en mi caso, prueba1, procederemos a ejecutarla desde Docker, se hace a través del comando:

**docker run -p 8080:8080 nombre\_de\_la\_imagen**

Reemplazar “nombre\_de\_la\_imagen” por el nombre de la imagen que queremos ejecutar, en este caso, es la que acabamos de crear, llamada “prueba1”



Listo. Ya podemos llamar al recurso del proyecto que creamos a través del localhost:



Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente