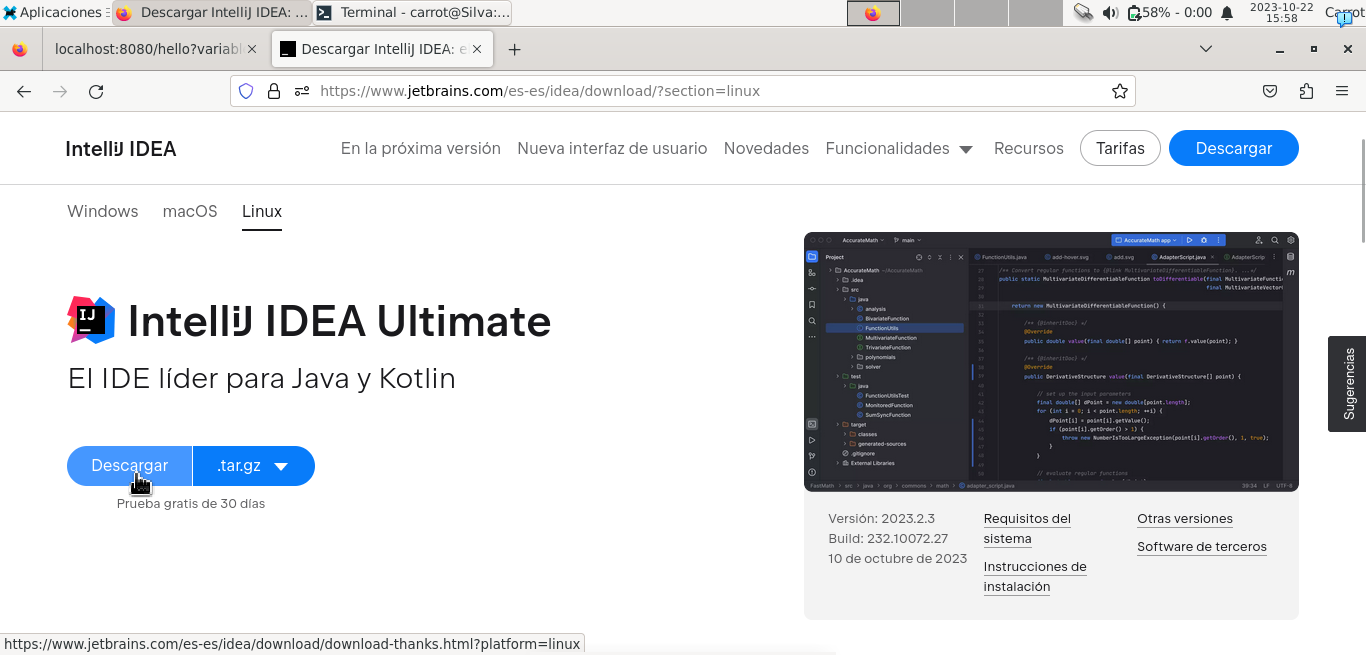
**FORMATO DE DOCUMENTACIÓN**

|  |  |  |  |
| --- | --- | --- | --- |
| **Fecha:** | 2023-10-21 | **Grupo:** | Semillero Lab#1 |
| **Proyecto:** | DepGirPro v1 | **Squad:** | DevOps |
| **Elaboró:** | Squad de DevOps | **Aprobó:** |  |
| **Tipo:** | Manual | | |
| **Descripción:** | Despliegue de una plantilla de aplicación web del framework Spring Boot de Java utilizando Docker en Debian 12 | | |

**Instalación de IDE (IntellIj Idea Ultimate)**

Se utilizará un IDE que tiene como propósito crear aplicación con Java con mayor facilidad, para instalarlo deberemos de instalarlo desde la página oficial de Jetbrains [www.jetbrains.com/es-es/idea/download](http://www.jetbrains.com/es-es/idea/download).



Deberemos de instalarlo con la extensión para poder descomprimirlo más adelante, al descargarlo deberemos de ir al terminal y dirigirnos a la carpeta de Descargas de nuestro usuario de Linux.

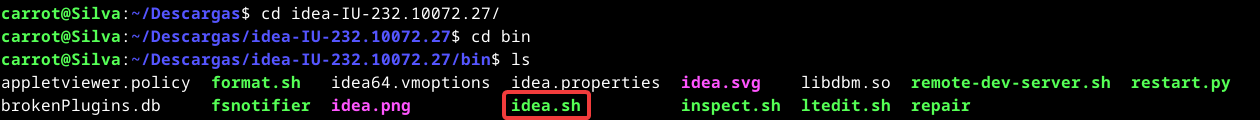
Interfaz de usuario gráfica

Descripción generada automáticamente con confianza baja

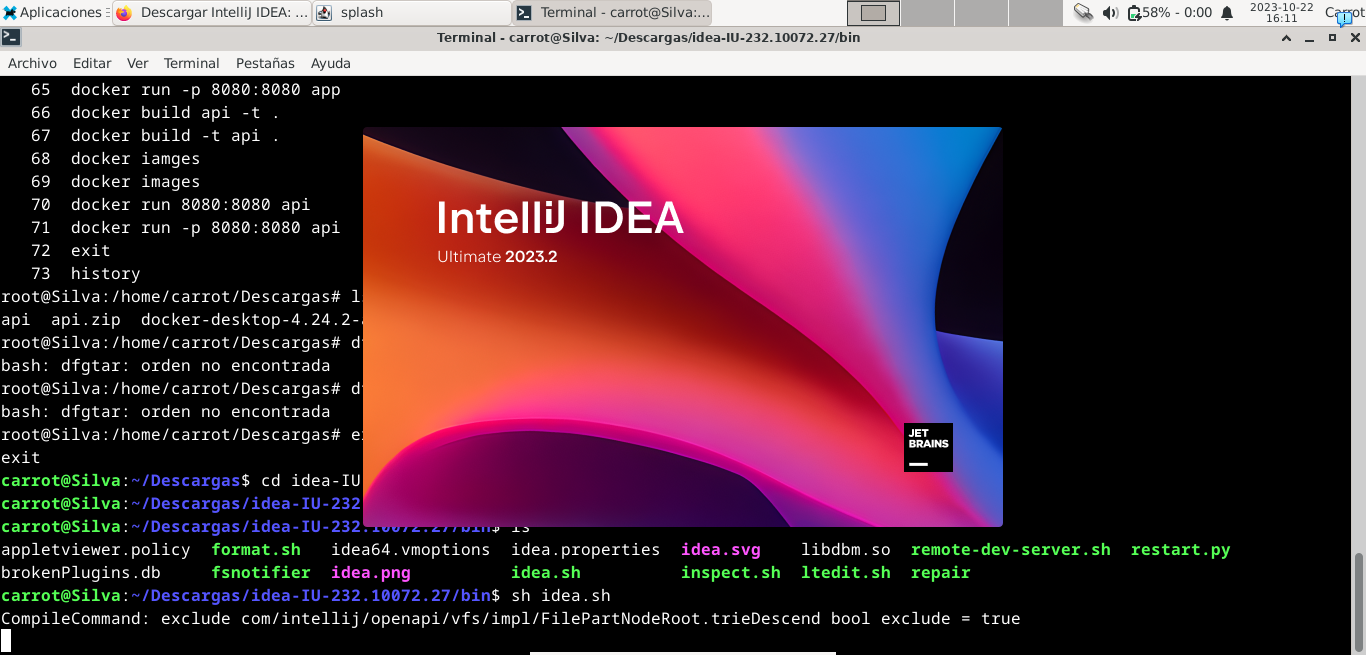
Por consiguiente, deberemos de descomprimir el archivo descargado, por defecto el nombre del archivo se conoce como , para descomprimirlo ejecutaremos el comando:



Al ejecutarlo empezara a mostrar en pantalla todos los archivos que descomprime el archivo, al terminar deberemos de abrir IntellIj Idea Ultimate desde su ejecutable , para ello deberemos de dirigirnos a la siguiente ruta.



Ejecutando las 3 líneas de comando en la terminal nos habremos dirigido a la carpeta raíz de donde se guarda la configuración del IDE, para poder abrirlo debimos habernos metido a su carpeta , y ejecutar un archivo de script llamado con el siguiente comando:



Al abrirlo empezara a abrir el IDE correctamente.

**Instalación de una plantilla de Spring Boot**

Ahora deberemos de instalar una plantilla común y corriente de Spring Boot, para ello deberemos de dirigirnos al siguiente enlace a través del navegador:

<https://start.spring.io/>

Al entrar deberemos de configurar los parámetros con el que se hará la aplicación del , en este caso utilizaremos los siguientes parámetros

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

Inicialmente utilizaremos el lenguaje Java utilizando un proyecto de Maven, utilizaremos la última versión estable siendo la 3.1.5, y como información del proyecto puede ser a gusto de cualquier persona, pero en este caso se utilizara un grupo llamado , y el se llamara , la descripción puede ser cualquiera y el sera automatizado en los parámetros elegidos, como se comprimirá al construir el proyecto en este caso será de un utilizando Java 17.

Después deberemos de instalarle una dependencia que es la que se encargara de darle sentido a la funcionalidad para una aplicación WEB, para eso deberemos de dirigirnos a las dependencias a la creación del proyecto y darle a añadir una nueva dependencia.

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

Dentro podemos elegir múltiples dependencias que tendrá el proyecto, el que nos interesa sera “Spring Web”, que para buscarlo debemos poner “web” en la búsqueda de dependencias.

Interfaz de usuario gráfica, Texto, Aplicación

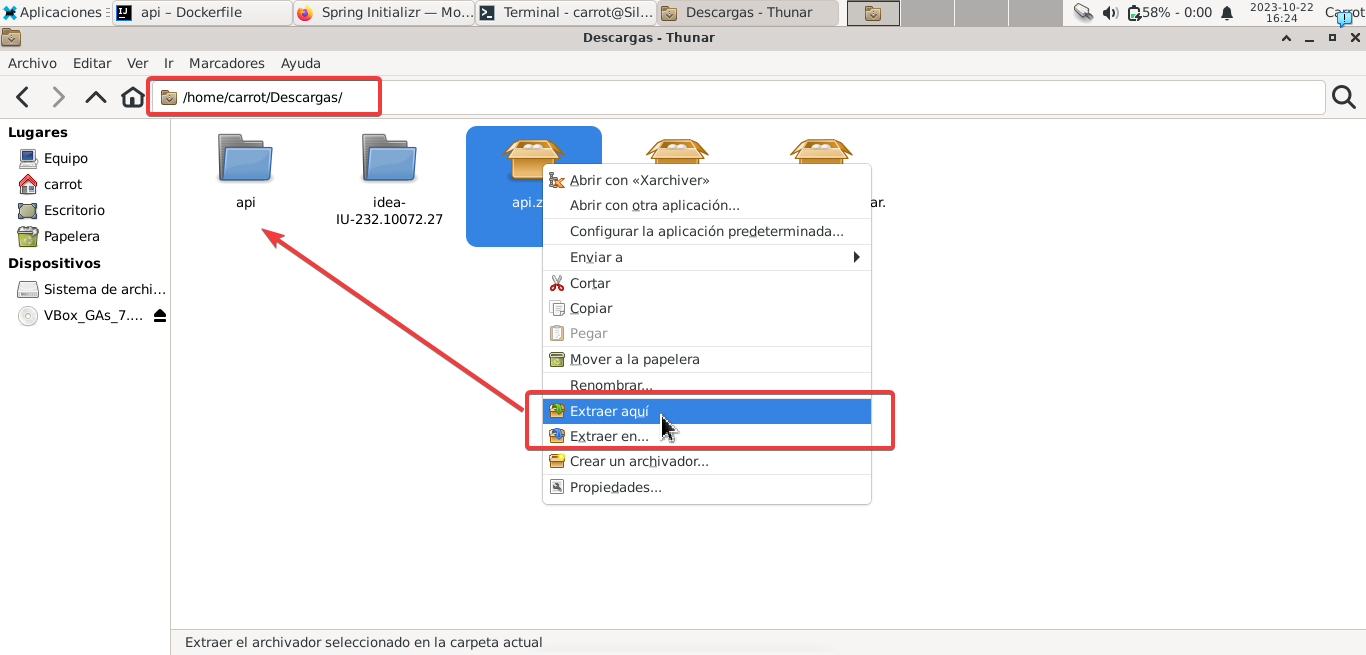
Descripción generada automáticamente

Ya con ello podremos descargar nuestra plantilla de proyecto para la creación de nuestra plantilla de aplicación web que utilizaremos para el despliegue utilizando Docker, para eso lo descargaremos dándole clic a Generate.

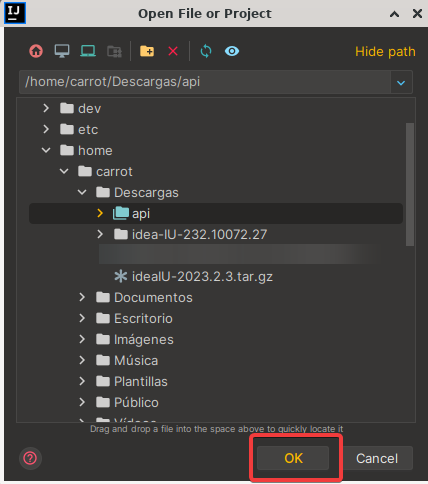
Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

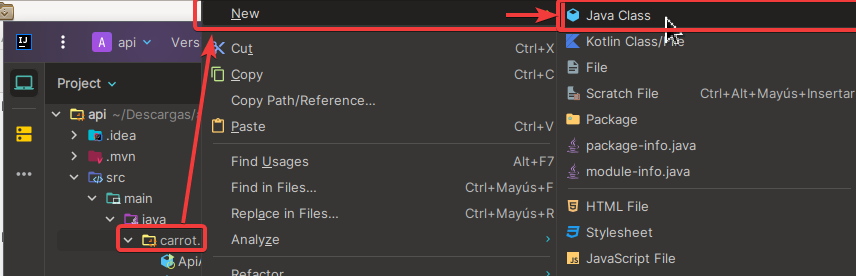
Hecho esto se descargara la plantilla que hemos creado en formato “.zip”, se descomprime a través del gestor de archivos de Linux



Se habría creado una carpeta para poder acceder desde a nuestro proyecto con el IDE descargado previamente, para ello abriremos el IDE y buscaremos la ruta del proyecto para poder abrirlo como proyecto Maven.



Empezara a abrirse el proyecto y empezaría a descargar las dependencias que solicitamos en los pasos anteriores, hecho esto procederemos a crear un Controller que sera una prueba para comprobar más adelante el funcionamiento de peticiones a través de Docker, para ello crearemos una clase llamado ApiHelloWorld, creandolo a través desde la ruta del package.



Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

**Instalación de Java 17**

Dentro del archivo, el IDE nos proporciona una utilidad para poder implementar un SDK, significando que podemos adicionar un interprete de Java que podemos descargar ahí mismo.

Interfaz de usuario gráfica

Descripción generada automáticamente

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

Con esto ya se descargaría el Java 17 para que el IDE pueda interpretar nuestro código de forma más fácil.

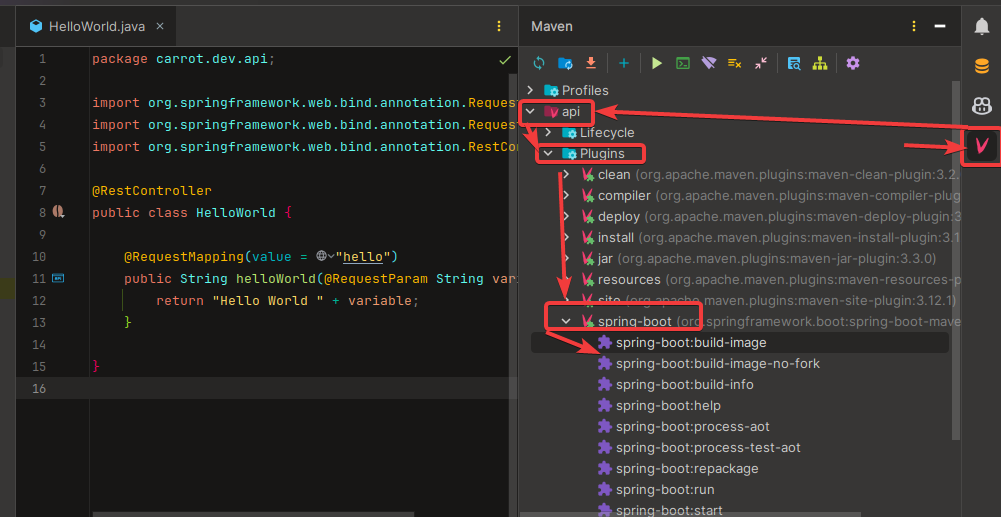
**Creación de la funcionalidad de la aplicación WEB.**

Dentro de la clase copiaremos el siguiente código, esto nos indica que utilizaremos un controlador Rest como anotación que indica la línea 7, y dentro de la clase del archivo crearemos una función que devolverá un String que tendrá como parámetros un String que se solicita como un parámetro llamado “variable” como recurso en la url que se identifica como un String y devuelve una concatenación de la variable junto a un “Hello World”, además de ello se adiciona una anotación de Request Mapping que tiene como función de implementarlo como posible ruta en la aplicación.

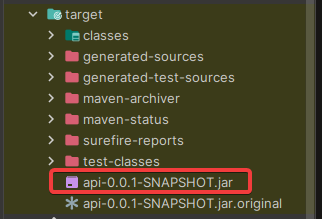
Texto

Descripción generada automáticamente

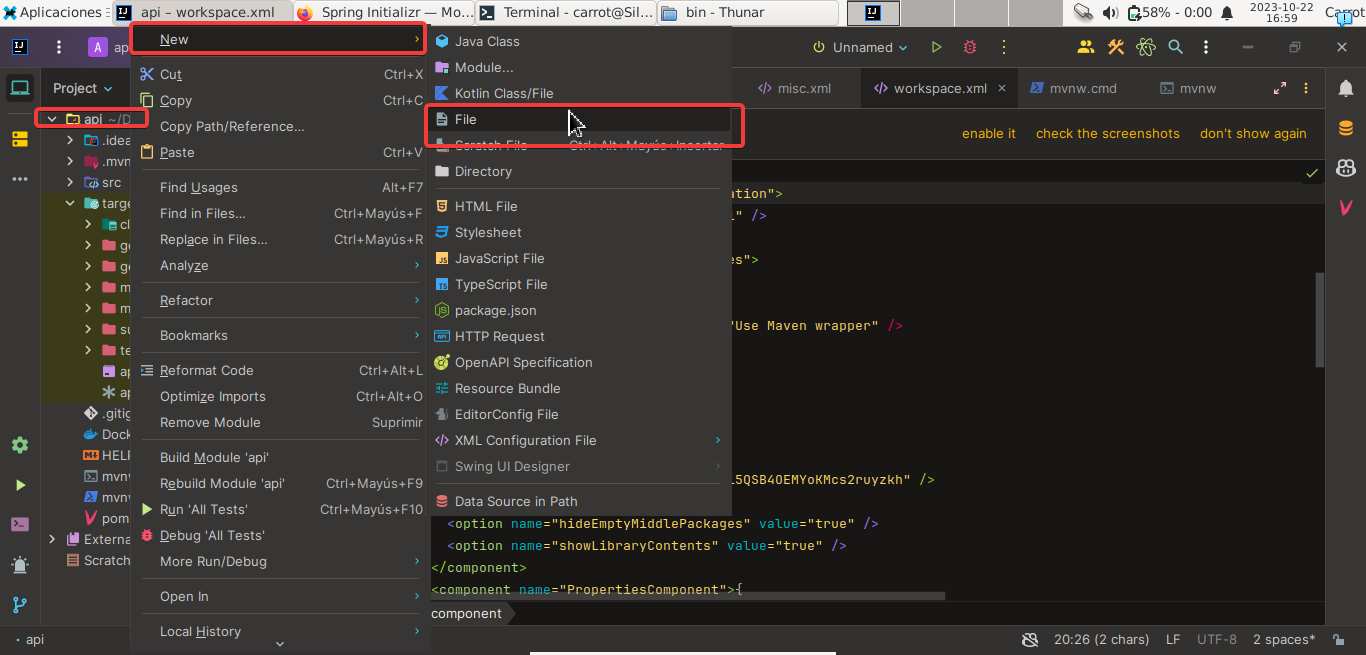
Hecho esto ya habríamos creado la primera funcionalidad de la página, como siguiente paso deberemos de crear una build del proyecto, para ello deberemos de hacer los siguiente.



Ya con esto, se creará una ruta llamada “target” en la ruta del proyecto y ahí se encontrará el archivo que hemos creado con la clase que hemos creado anteriormente



Ahora deberemos de desplegarlo en Docker, para ello deberemos de crear un archivo llamado “Dockerfile” en los archivos del proyecto.



Interfaz de usuario gráfica, Texto, Sitio web

Descripción generada automáticamente

Dentro de esa carpeta deberemos de pegar lo siguiente

Captura de pantalla de un celular

Descripción generada automáticamente

Ojo, en la línea 4 es donde debimos haber puesto la ruta anterior donde se buildeo nuestro proyecto con el nombre del archivo.

**Instalación de Docker**

Antes de desplegar nuestro proyecto deberemos de tener Docker instalado en nuestro sistema de Debian, para ello deberemos de seguir los siguientes pasos.

# Añadir la llave GPG oficial de Docker.

sudo apt-get update

sudo apt-get install ca-certificates curl gnupg

sudo install -m 0755 -d /etc/apt/keyrings

curl -fsSL https://download.docker.com/linux/debian/gpg | sudo gpg --dearmor -o /etc/apt/keyrings/docker.gpg

sudo chmod a+r /etc/apt/keyrings/docker.gpg

# Añadir el repositorio de Docker

echo \

"deb [arch="$(dpkg --print-architecture)" signed-by=/etc/apt/keyrings/docker.gpg] https://download.docker.com/linux/debian \

"$(. /etc/os-release && echo "$VERSION\_CODENAME")" stable" | \

sudo tee /etc/apt/sources.list.d/docker.list > /dev/null

sudo apt-get update

Hecho los pasos anteriores deberemos de instalar Docker con el siguiente comando

sudo apt-get install docker-ce docker-ce-cli containerd.io docker-buildx-plugin docker-compose-plugin

Con ello ya se habría instalado Docker correctamente para comprobar su funcionamiento.



Con el ultimo paso deberemos de añadirle permisos a nuestro usuario para poder usar Docker de forma que no dependamos de ser usuarios root, para eso deberemos de ejecutar el siguiente comando.

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

Después de eso deberemos de reiniciar el sistema de Debian.

**Despliegue de aplicación en Docker**

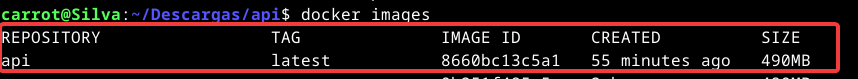
Una vez desplegado, deberemos de ir a la ruta del proyecto.



Dentro de la carpeta deberemos de crear una imagen para el Docker que nos permita ejecutar la build del proyecto utilizando el .jar generado previamente.

docker build -t api .

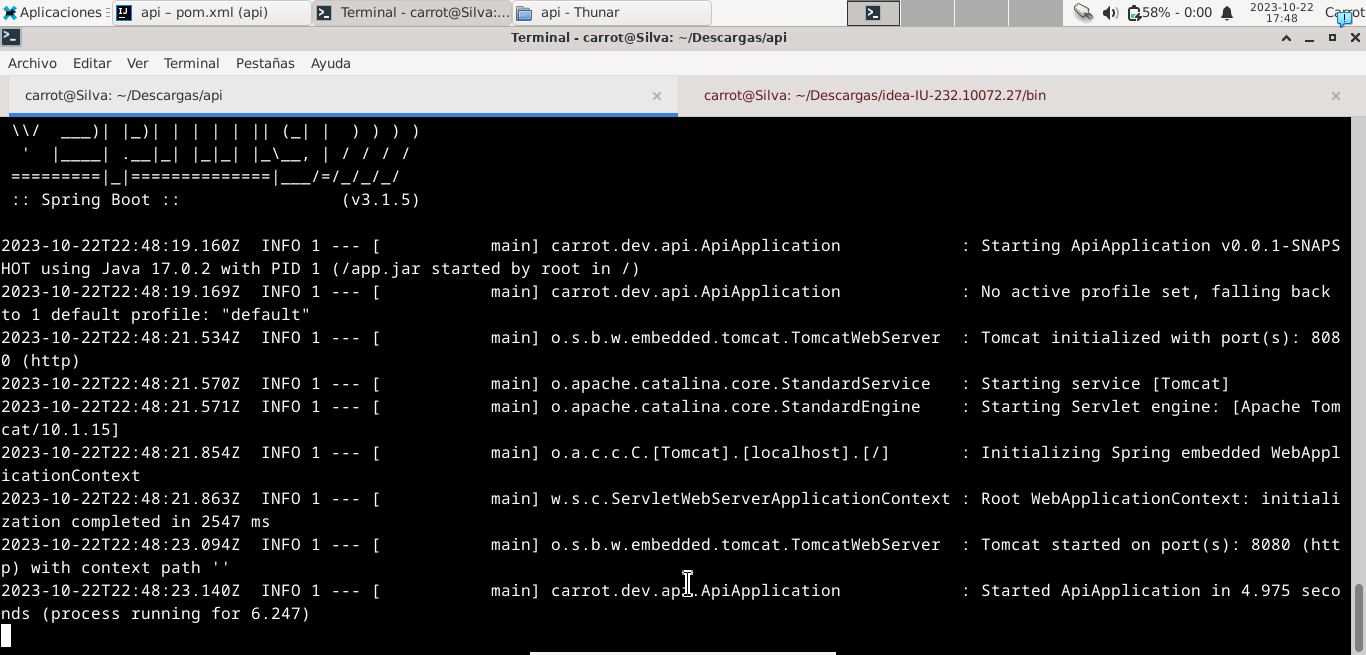
Indica que se crea una imagen para docker llamado api, que utiliza todos los directorios de la carpeta raíz, realizado el comando comprobaremos si se realizo correctamente ejecutando el comando .



Al realizar la comprobación ya estaría listo para poder desplegar la aplicación, para ello ejecutaremos el siguiente comando.

docker run -p 8080:8080 api

Al hacer esto le indicamos a Docker que corra la imagen api que creamos anteriormente en los pasos anteriores en un servidor local en el puerto 8080, habiendo ejecutado el comando nos encontramos que el servidor ha sido ejecutado de forma correcta.



Si entramos al navegador del sistema, en la ruta , nos encontraremos con una petición exitosa con la clase creada previamente.

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

Con estos pasos ya estaríamos preparados para desplegar una aplicación en Docker de forma exitosa.

Si quisiéramos volver a desplegar nuevos cambios, tendríamos que volver a repetir los pasos anteriores, incluyendo haciendo una build en el IDE, y haciendo una build en el Docker y volviéndolo a iniciar.