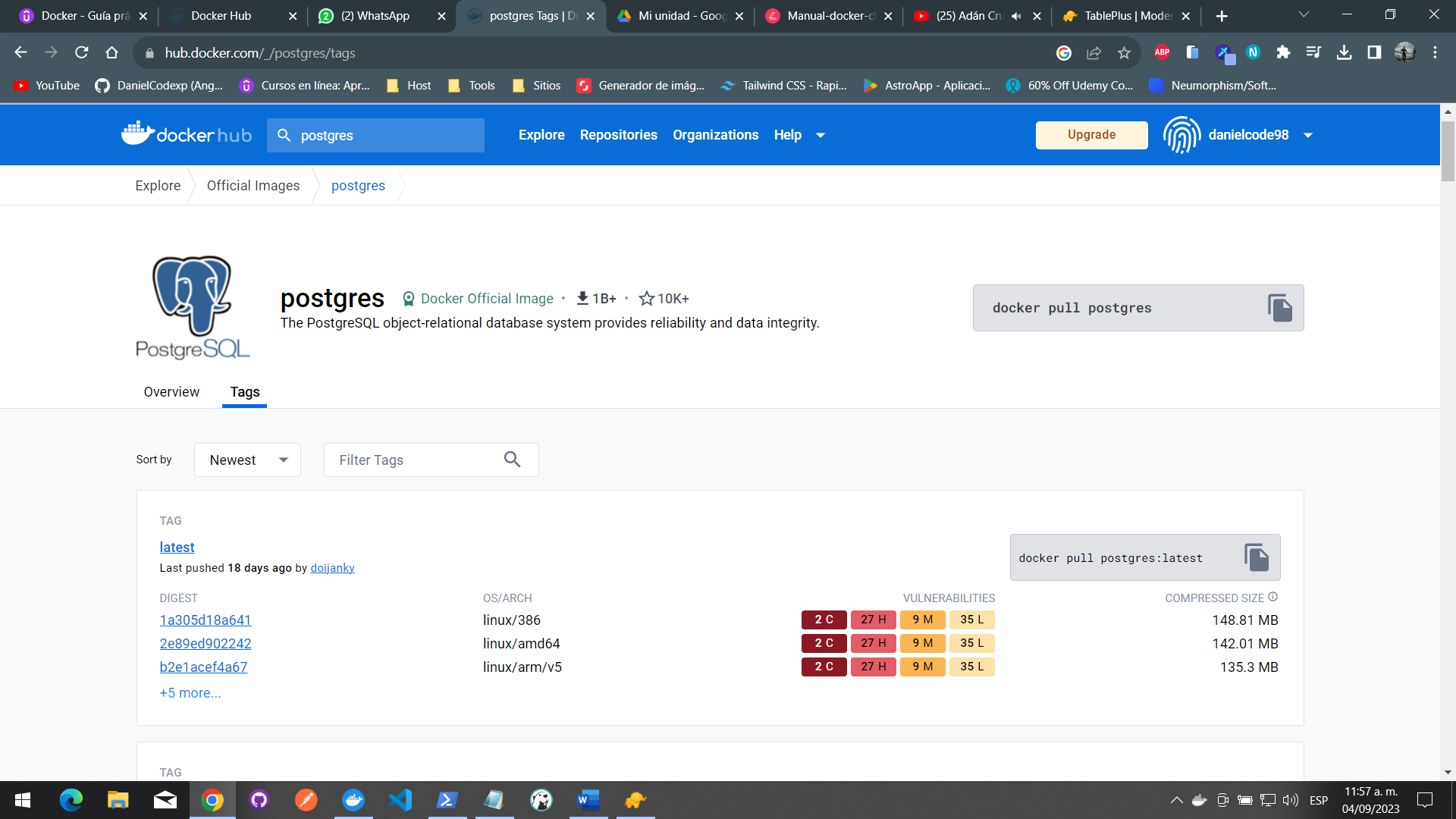
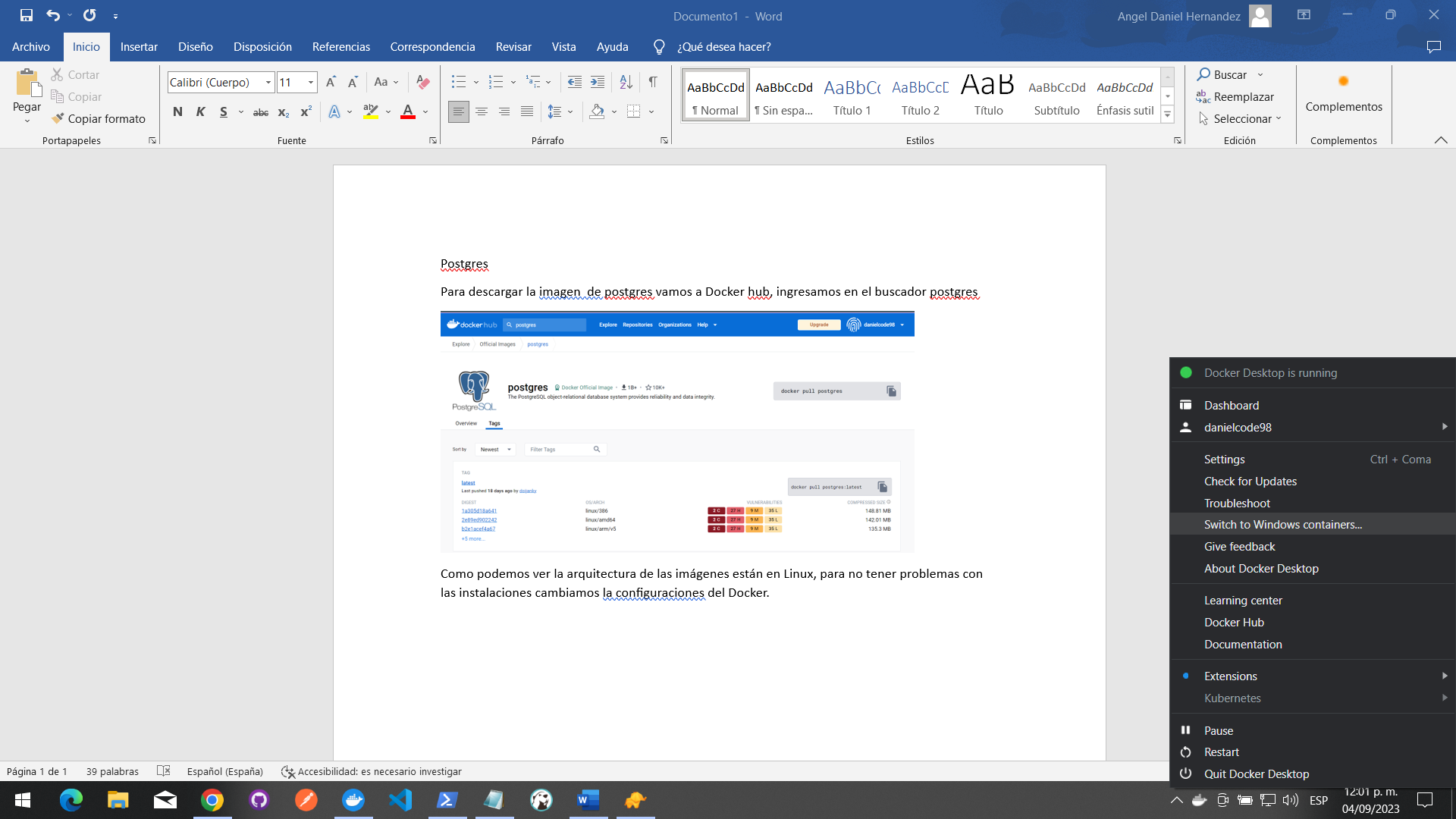
PostgreSQL

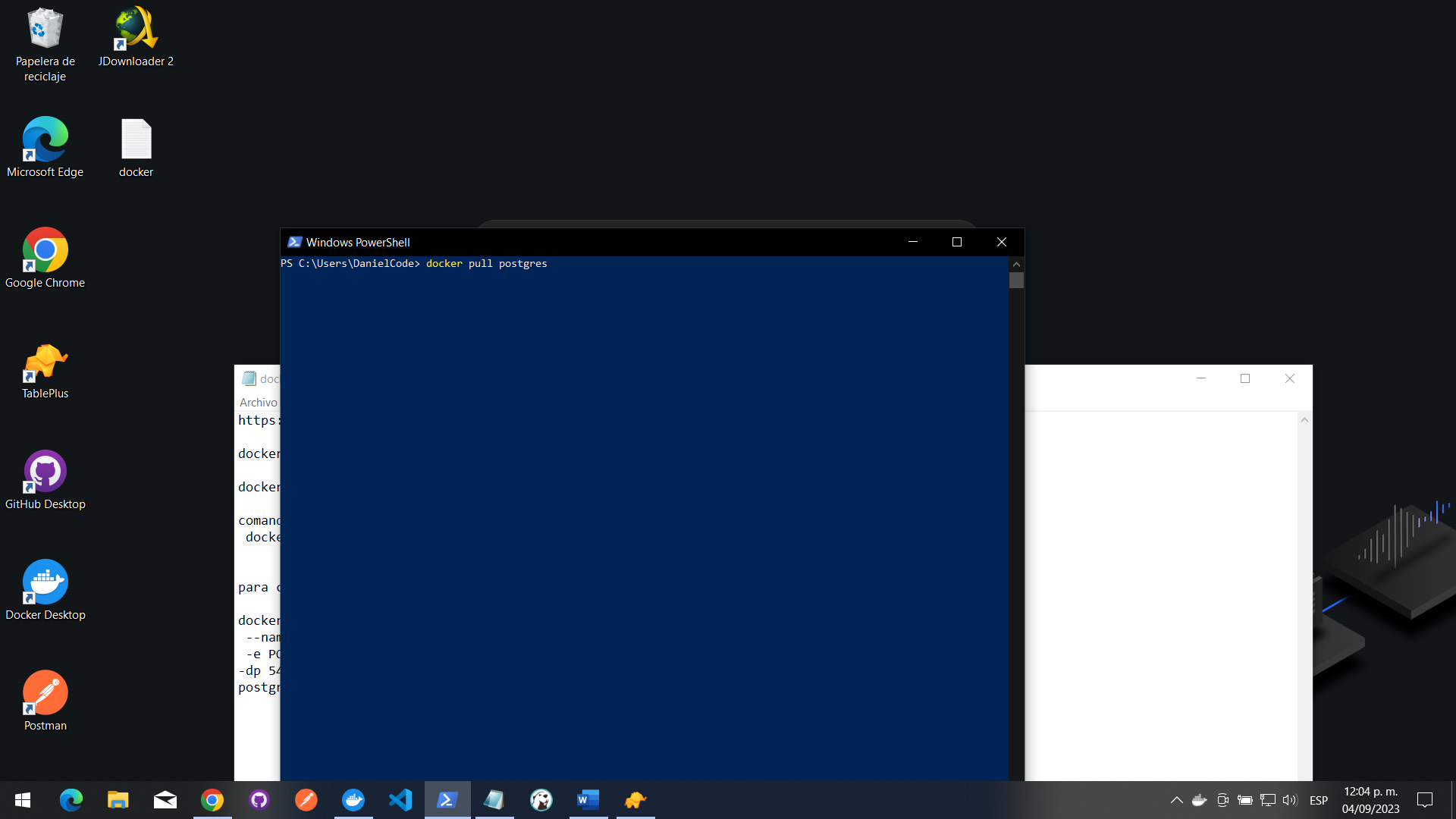
Para obtener la imagen de PostgreSQL, dirigimos nuestra atención a Docker Hub y procedemos a buscar "PostgreSQL" en el campo de búsqueda.



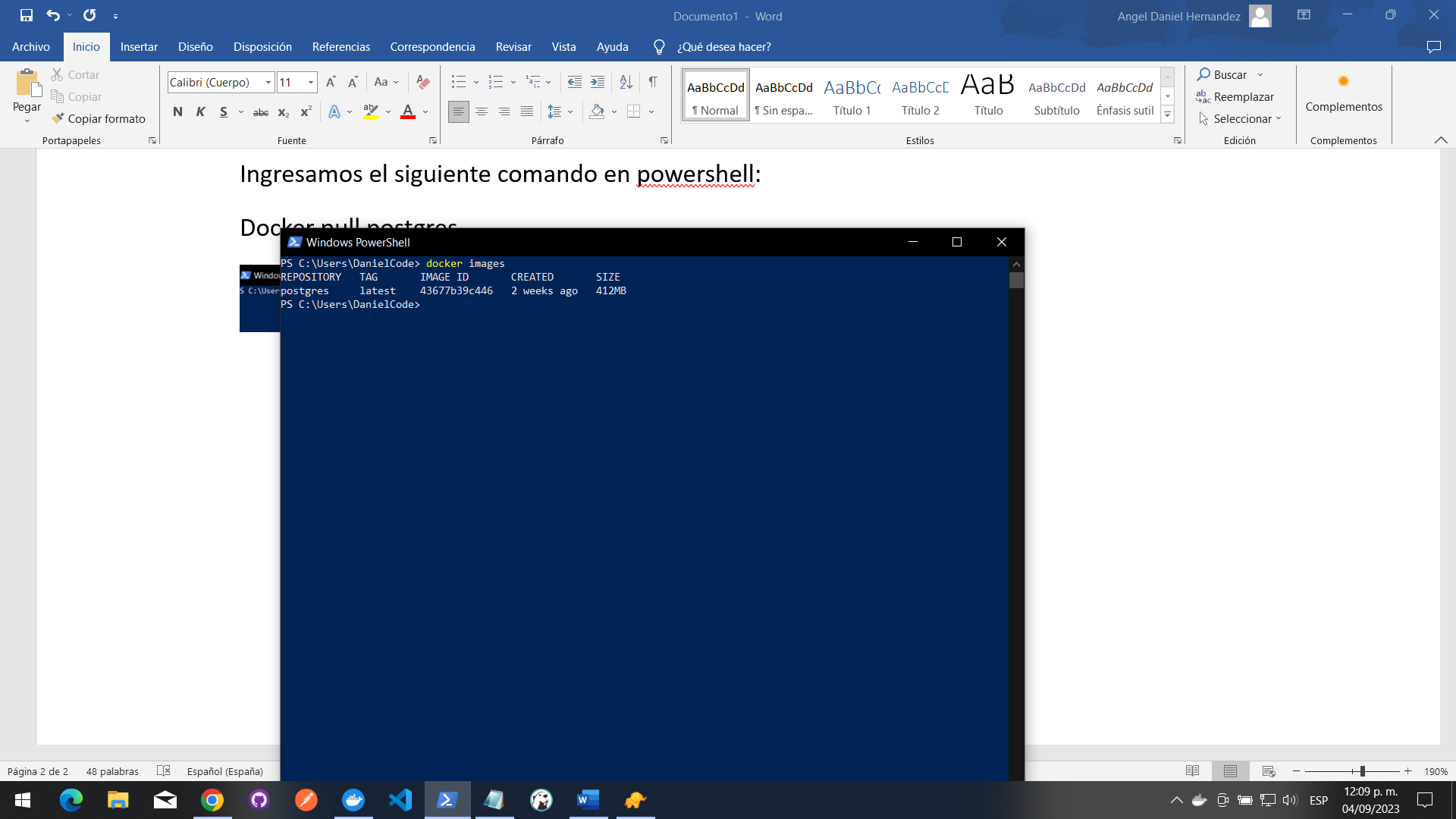
Para garantizar una instalación sin problemas en sistemas distintos de Linux, es importante ajustar la configuración de Docker. Esto se debe a que la arquitectura de las imágenes de Docker generalmente se basa en Linux. A continuación, se presenta una forma de expresar esta idea de manera más clara:

Dado que las imágenes de Docker están diseñadas principalmente para arquitecturas basadas en Linux, es esencial modificar la configuración de Docker para asegurar una instalación sin inconvenientes en otros sistemas operativos.

En PowerShell, ejecutamos el siguiente comando para descargar la imagen oficial de PostgreSQL desde Docker Hub:



Después de que la descarga haya finalizado, podemos verificar si la imagen se descargó correctamente ejecutando el siguiente comando:



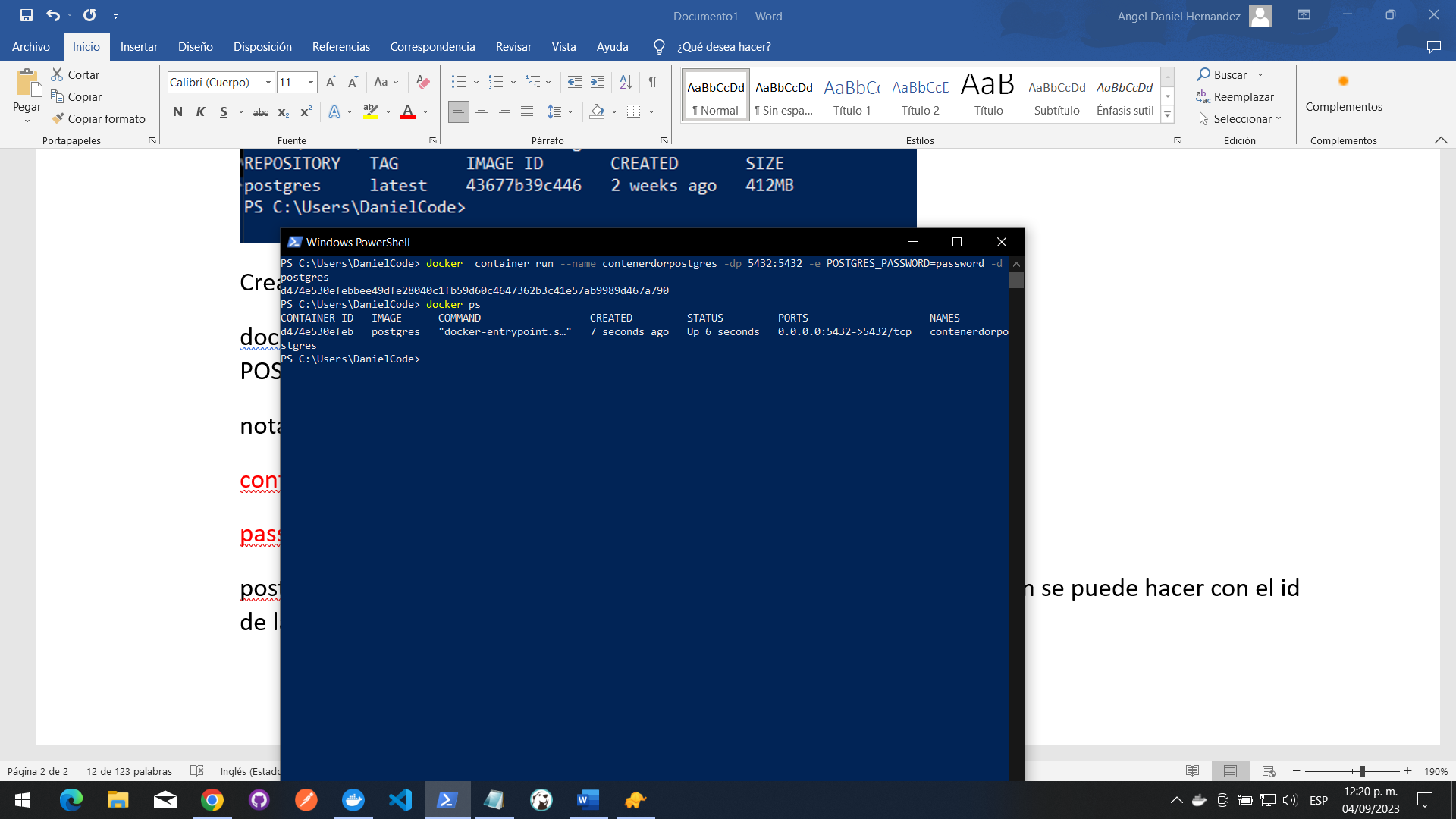
Para crear un contenedor de prueba de PostgreSQL, utiliza el siguiente comando en PowerShell:

docker container run --name contenedorpostgres -dp 5432:5432 -e POSTGRES\_PASSWORD=password -d postgres

* **--name contenedorpostgres**: Esto asigna el nombre "contenedorpostgres" al contenedor que estás creando.
* **-dp 5432:5432**: Esto mapea el puerto 5432 del contenedor al puerto 5432 de tu sistema local, lo que te permitirá acceder a PostgreSQL desde fuera del contenedor.
* **-e POSTGRES\_PASSWORD=password**: Establece la contraseña "password" para el usuario "postgres" en PostgreSQL. Asegúrate de cambiar "password" por una contraseña segura en un entorno de producción.
* **-d postgres**: Esto indica que se utilizará la imagen de PostgreSQL para crear el contenedor.

Una vez que hayas ejecutado el comando, puedes verificar si el contenedor se ha creado correctamente escribiendo:

docker ps

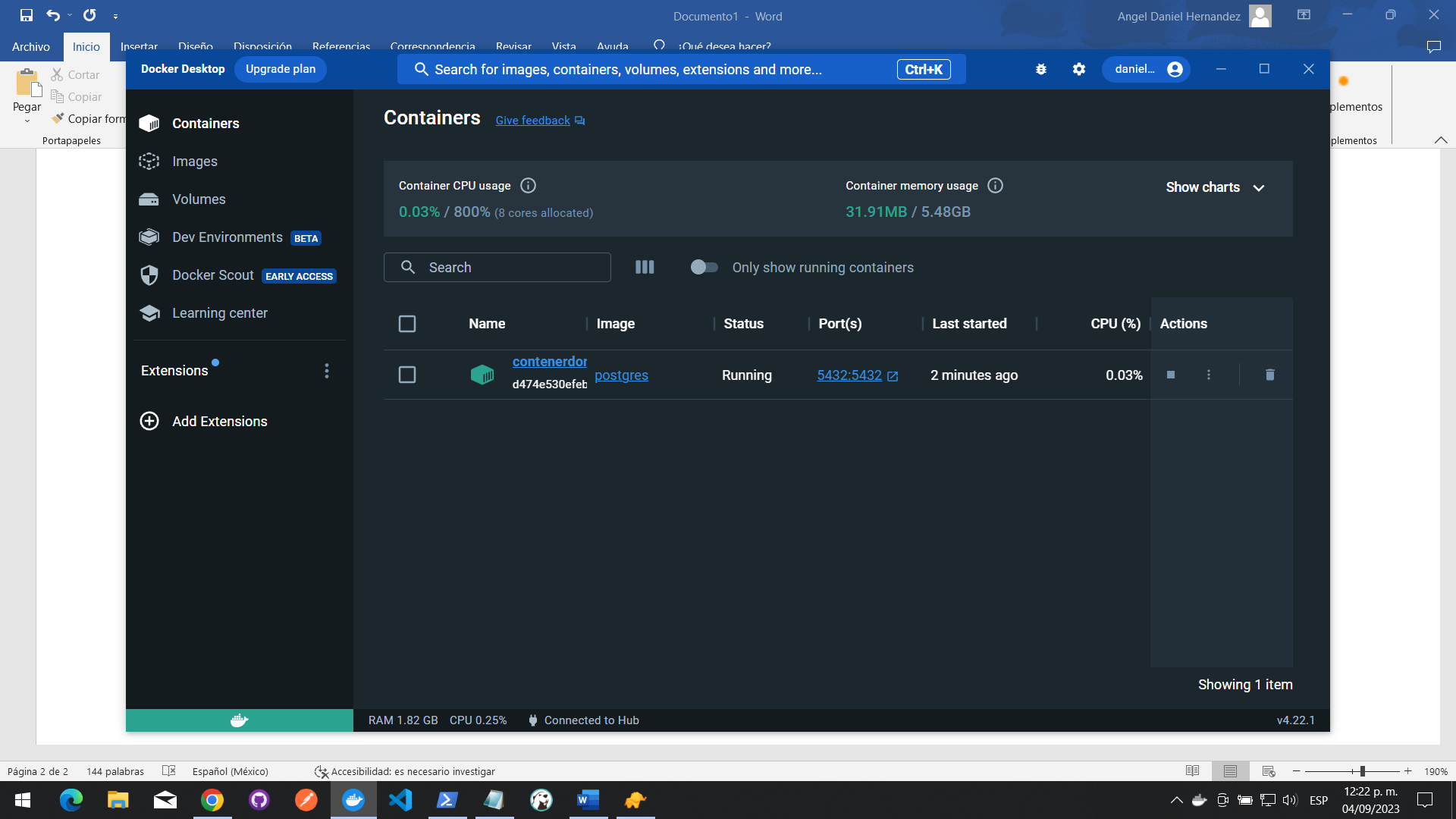


puedes verificar visualmente si tu contenedor de PostgreSQL se ha creado correctamente utilizando herramientas gráficas como Docker Desktop si estás en un entorno de Windows o Docker for Mac si estás en un entorno macOS.

Sigue estos pasos:

1. Abre la aplicación Docker Desktop (Windows) o Docker for Mac (macOS).
2. Una vez que esté en ejecución, verás el icono de Docker en la barra de tareas o la barra de menú, respectivamente. Haz clic derecho en el icono y selecciona "Dashboard" o "Ver contenedores" (puede variar según la versión).
3. Esto abrirá la interfaz gráfica de Docker que muestra una lista de tus contenedores. Si tu contenedor de PostgreSQL se ha creado correctamente, deberías verlo listado en la interfaz junto con su nombre ("contenedorpostgres") y otros detalles como el estado, los puertos mapeados, etc.
4. Puedes hacer clic en el nombre del contenedor para obtener más detalles sobre él y verificar su estado actual.

Esta interfaz gráfica te proporcionará una forma más visual y amigable de verificar y gestionar tus contenedores Docker.



Conectar tu contenedor PostgreSQL a un gestor de bases de datos como TablePlus es un paso importante para administrar y trabajar con tu base de datos de manera más eficiente. Aquí hay una guía general sobre cómo hacerlo:

1. Abre TablePlus en tu computadora.
2. En TablePlus, crea una nueva conexión o abre una existente si ya tienes una configuración previa. Para crear una nueva conexión, sigue estos pasos:

a. Haz clic en "File" (Archivo) en la parte superior izquierda de la ventana.

b. Selecciona "New Connection" (Nueva Conexión).

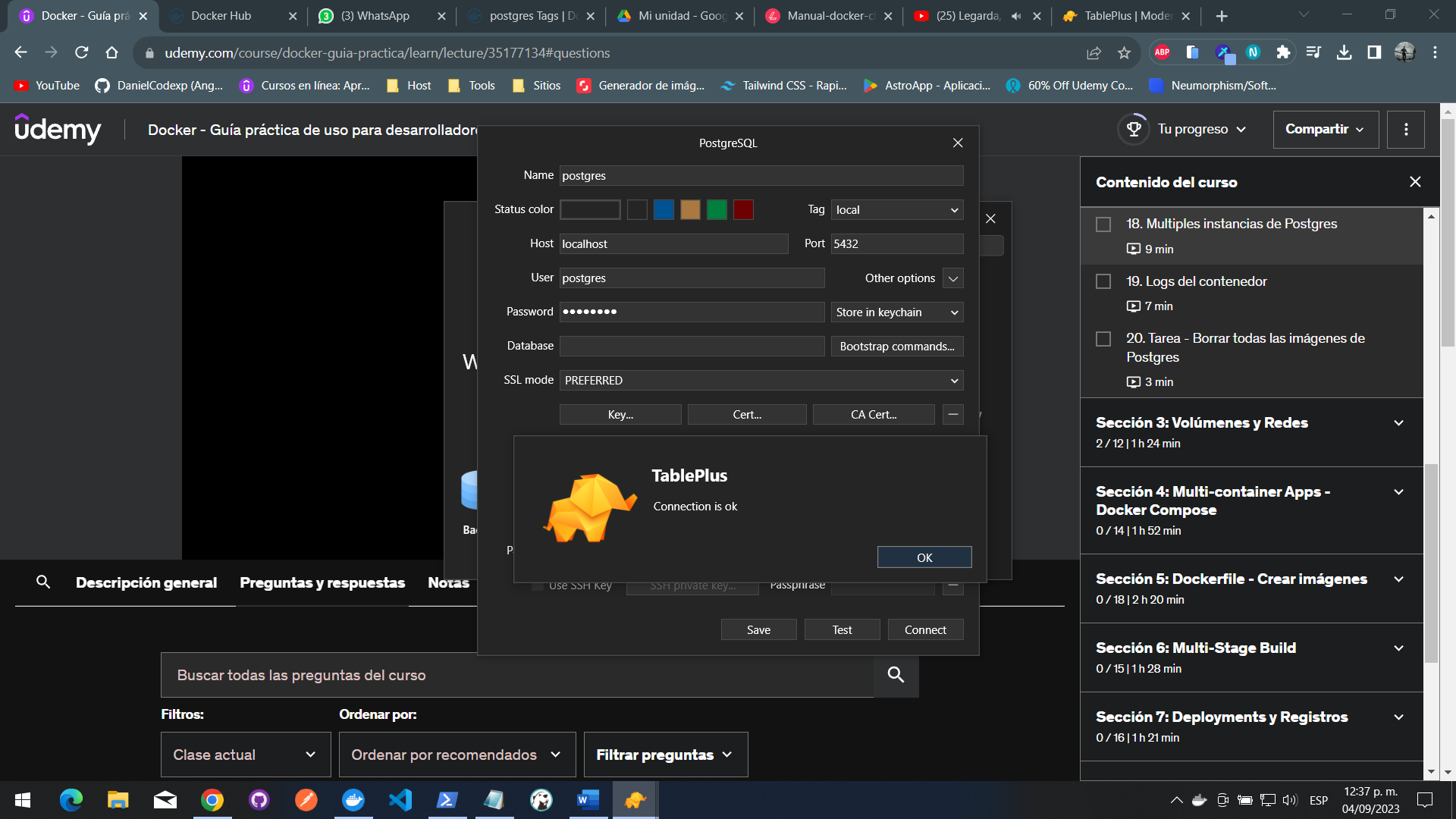
1. Configura la conexión a tu contenedor PostgreSQL:

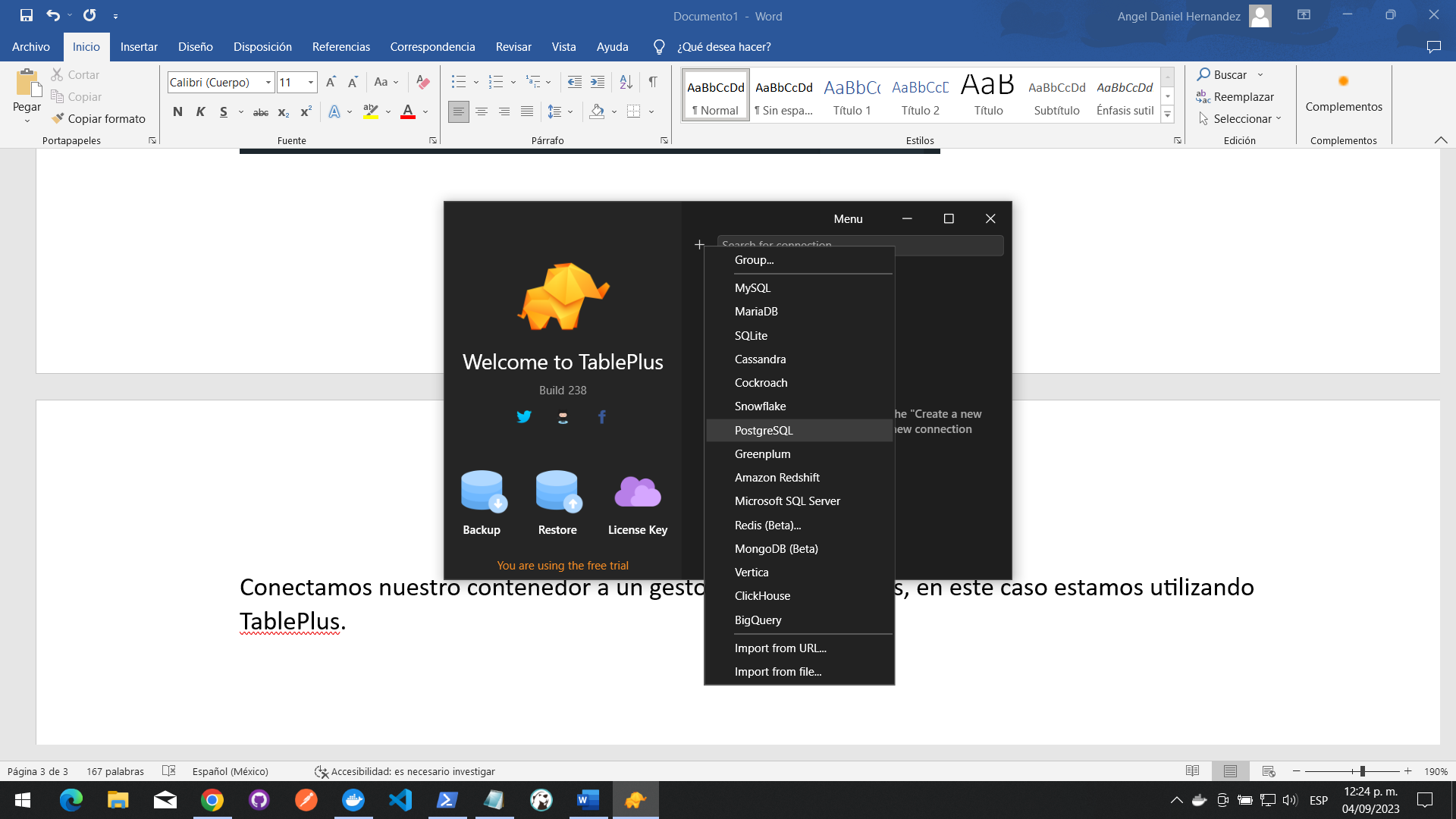
a. Selecciona el tipo de base de datos, que en este caso es "PostgreSQL".

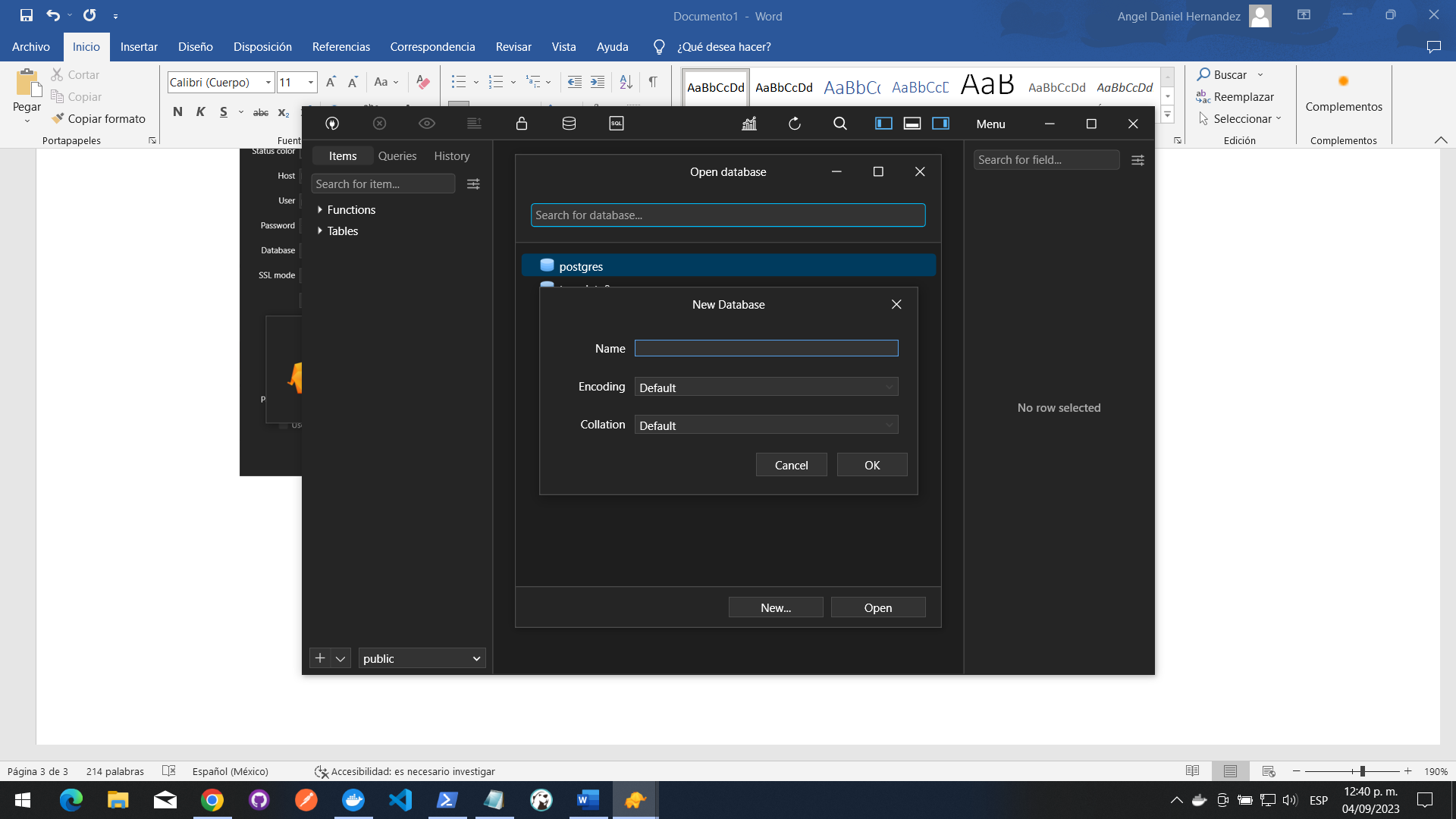
b. Completa los siguientes campos:

* + **Host**: Puedes utilizar la dirección IP de tu máquina local o "localhost" si estás ejecutando TablePlus en la misma máquina donde está Docker.
  + **Port**: Deja el puerto como "5432" si no has cambiado la configuración del puerto en tu contenedor.
  + **Database**: Puedes especificar la base de datos a la que deseas conectarte.
  + **Username**: Utiliza "postgres" u otro nombre de usuario si lo has configurado de manera diferente.
  + **Password**: Utiliza la contraseña que configuraste al crear el contenedor.

1. Haz clic en "Test" (Probar) para asegurarte de que TablePlus puede conectarse correctamente al contenedor.
2. Si la prueba es exitosa, haz clic en "Connect" (Conectar) para establecer la conexión.

Ahora deberías estar conectado a tu base de datos PostgreSQL que se ejecuta en el contenedor desde TablePlus. Puedes utilizar TablePlus para administrar y consultar la base de datos de manera gráfica y eficiente.

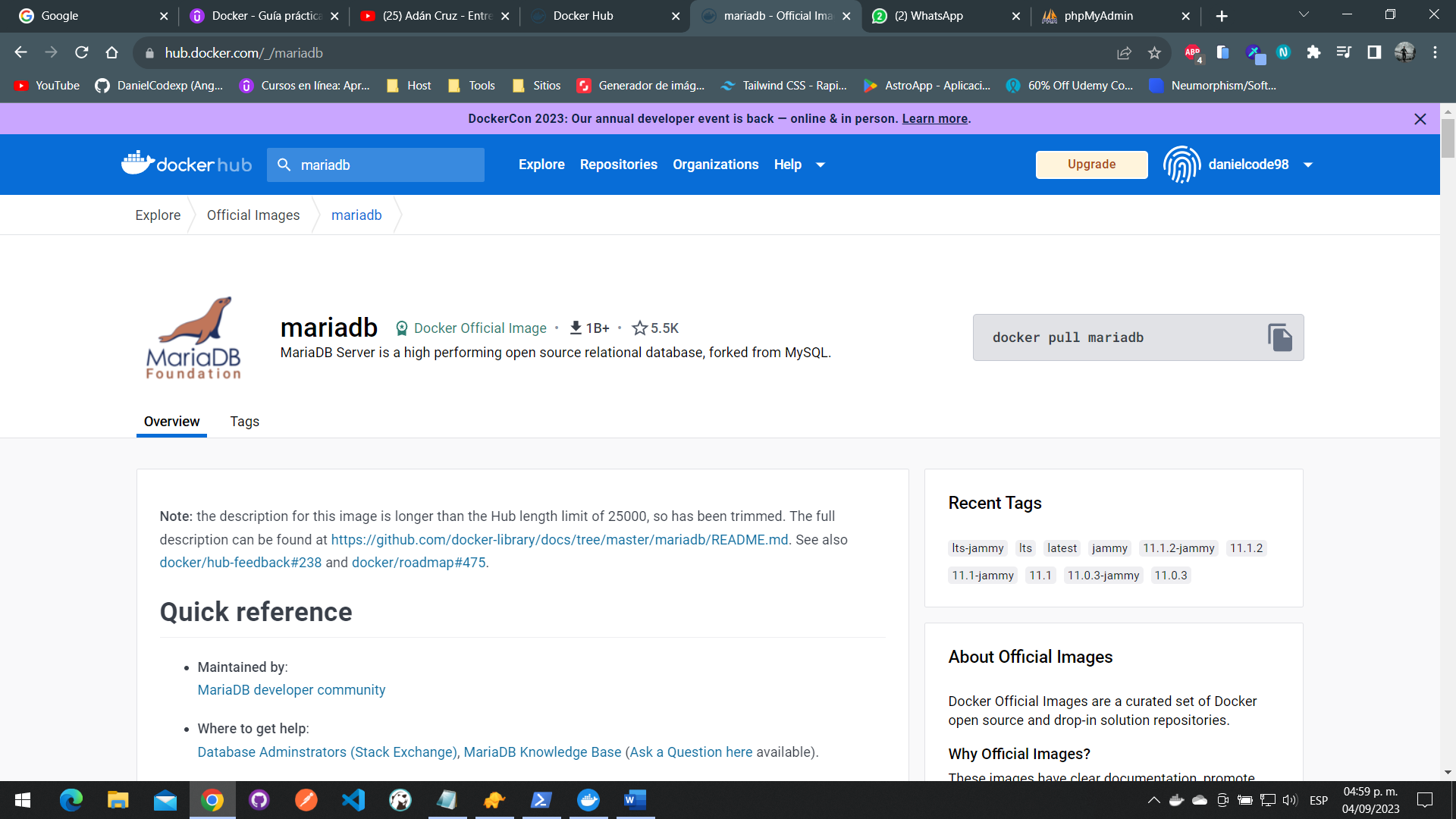




MariaDB y phpMyAdmin

Descargar la imagen de MariaDB desde Docker Hub:

Dirígete a Docker Hub y utiliza el buscador para encontrar la imagen de MariaDB. Puedes buscarla escribiendo "mariadb" en el campo de búsqueda.

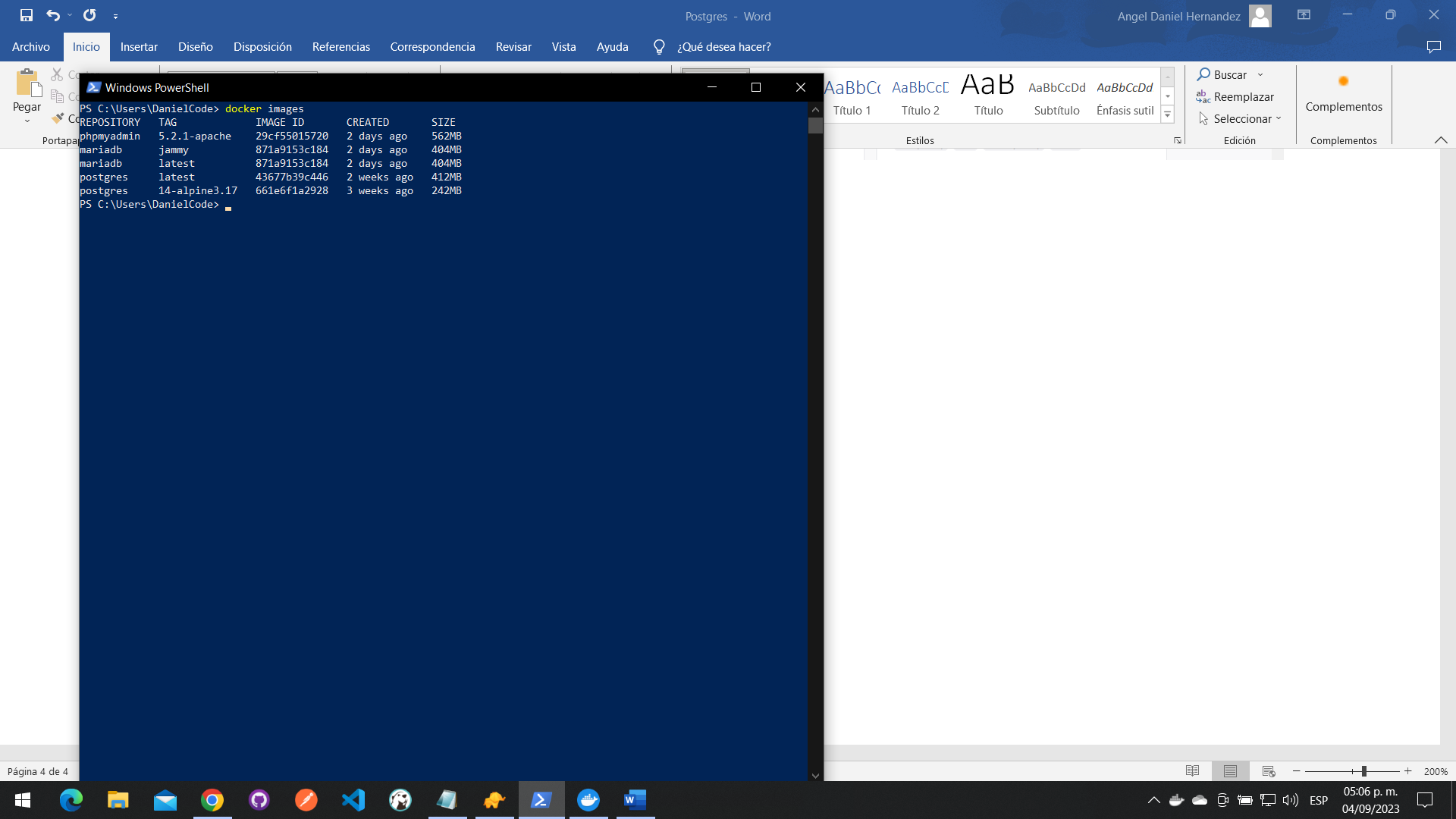


Descargar la imagen de MariaDB utilizando el siguiente comando en PowerShell:



Para comprobar si la imagen de MariaDB se descargó correctamente, puedes ejecutar el siguiente

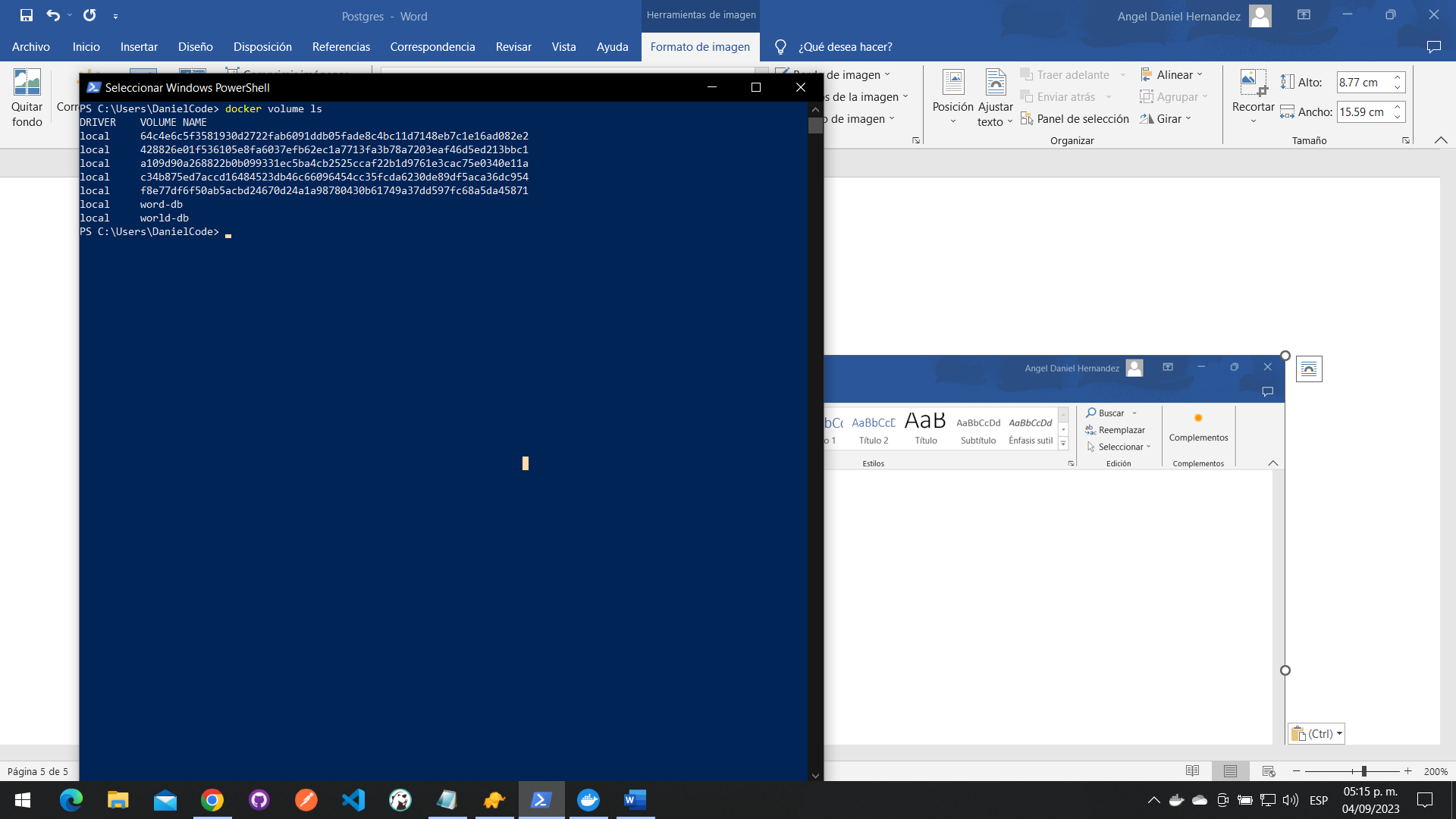
comando en PowerShell:



Después de descargar la imagen, procedemos a crear un volumen en Docker.

Un volumen es esencial para preservar los datos de un contenedor, permitiendo que la información persista incluso si el contenedor se elimina. Para llevar a cabo esta tarea, ejecutamos el siguiente comando:



Para visualizar la lista de volúmenes creados en Docker, puedes utilizar el siguiente comando:

A continuación, procederemos a crear un nuevo contenedor, aprovechando el volumen previamente creado. Utiliza el siguiente comando:



Este comando tiene los siguientes propósitos:

* **-d**: Ejecuta el contenedor en segundo plano, en modo daemon.
* **-p 3306:3306**: Realiza un mapeo del puerto 3306 del contenedor al puerto 3306 del sistema local.
* **--name world-db**: Asigna al contenedor el nombre "world-db".
* **--env MARIADB\_USER=example-user**, **--env MARIADB\_PASSWORD=user-password**, **--env MARIADB\_ROOT\_PASSWORD=root-secret-password**, **--env MARIADB\_DATABASE=world-db**: Define variables de entorno que configuran la base de datos MariaDB con un usuario, contraseña, contraseña de root y nombre de la base de datos específicos.
* **--volume world-db:/var/lib/mysql**: Vincula el volumen "world-db" que creamos previamente al directorio "/var/lib/mysql" del contenedor. Esto garantiza que los datos de la base de datos se almacenen en el volumen, lo que proporciona persistencia incluso si el contenedor se elimina.
* **mariadb**: Utiliza la imagen de MariaDB para crear el contenedor.

Con este comando, hemos configurado un contenedor de MariaDB que utiliza el volumen "world-db" para almacenar los datos de la base de datos, y hemos personalizado la configuración de MariaDB mediante variables de entorno.

Comprobamos si se creo el contenedor ingresando el siguiente comando:



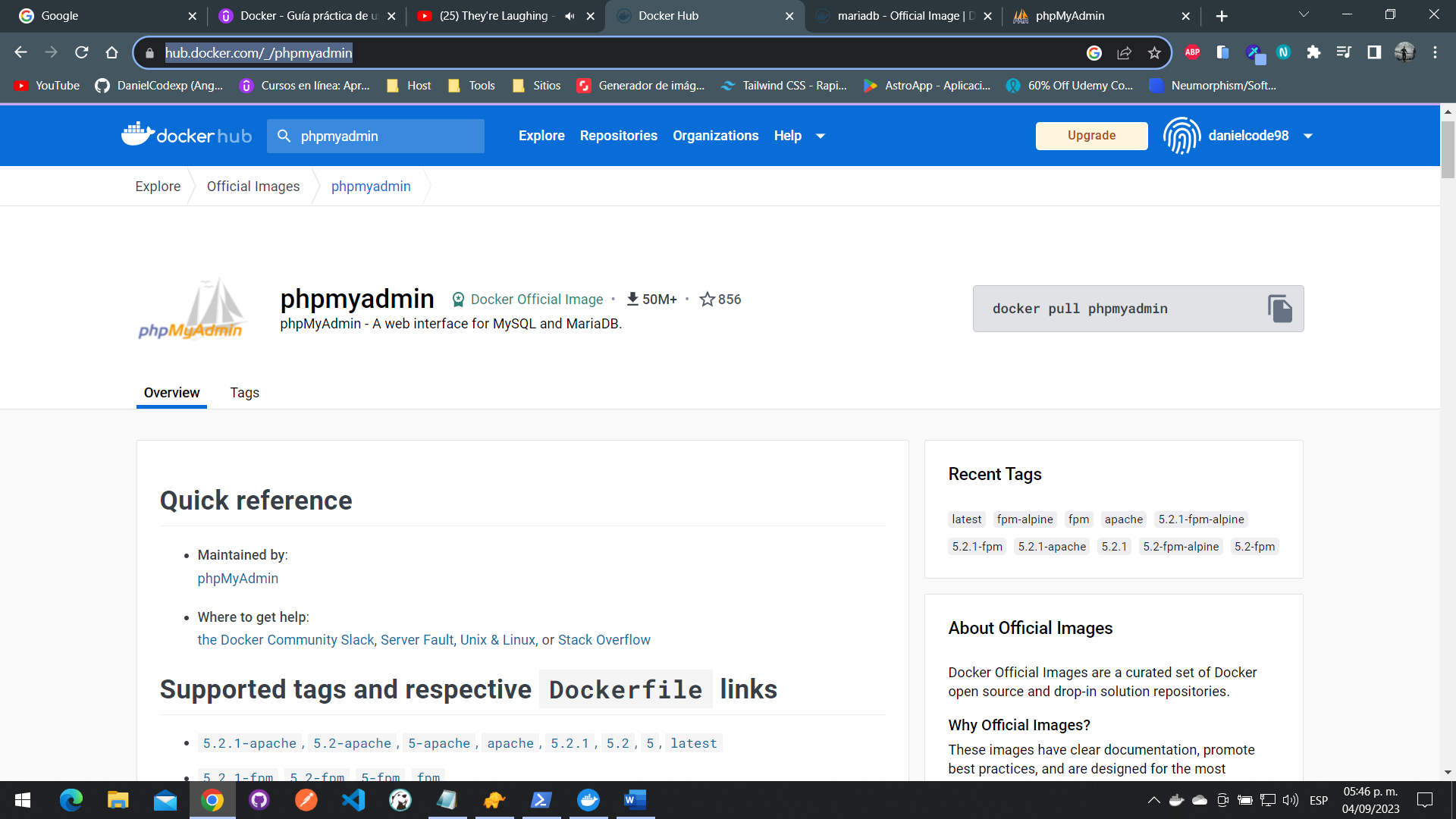
Establecemos una conexión utilizando TablePlus.



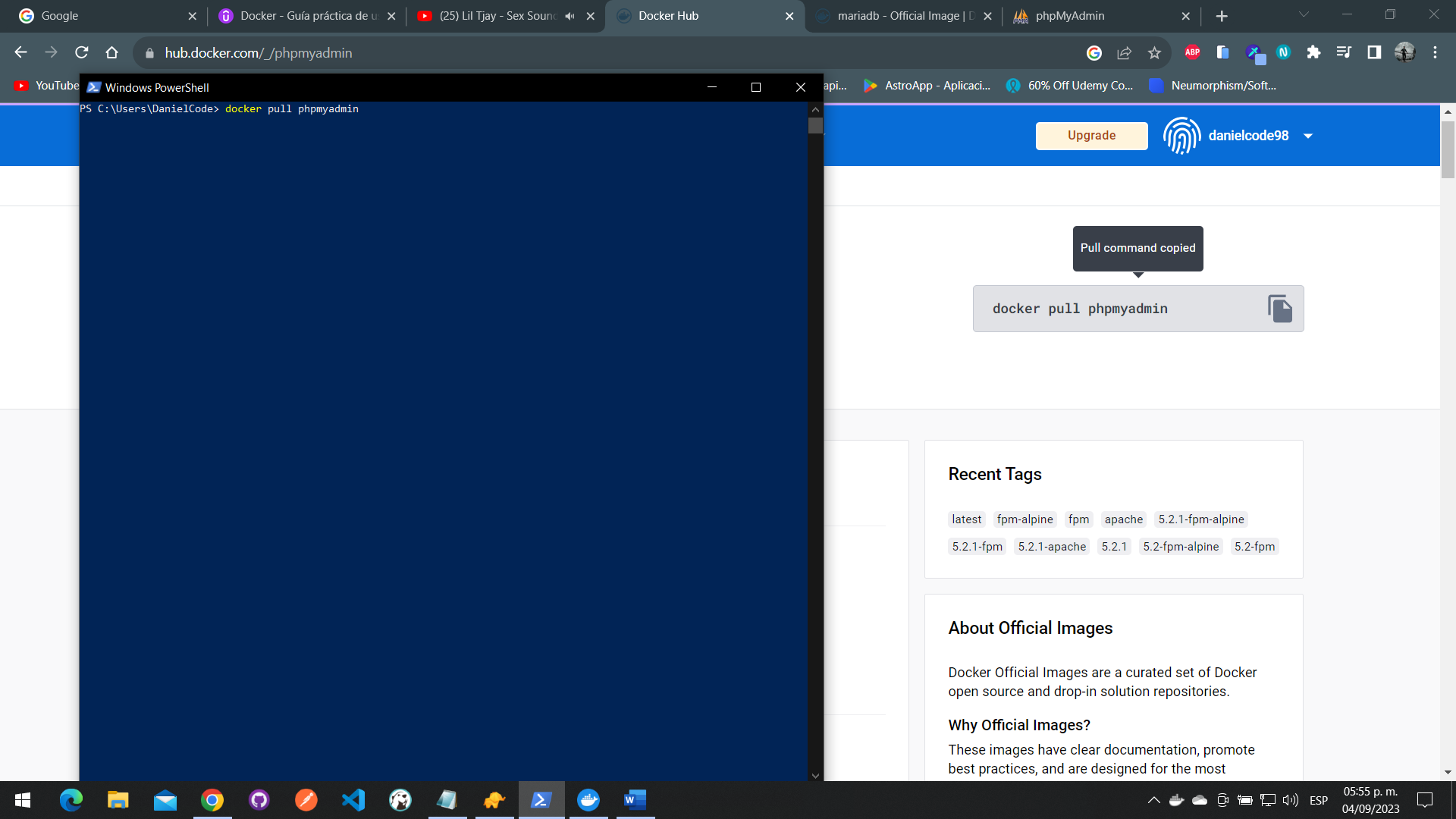
phpMyAdmin

Descargar la imagen de phpMyAdmin desde Docker Hub:

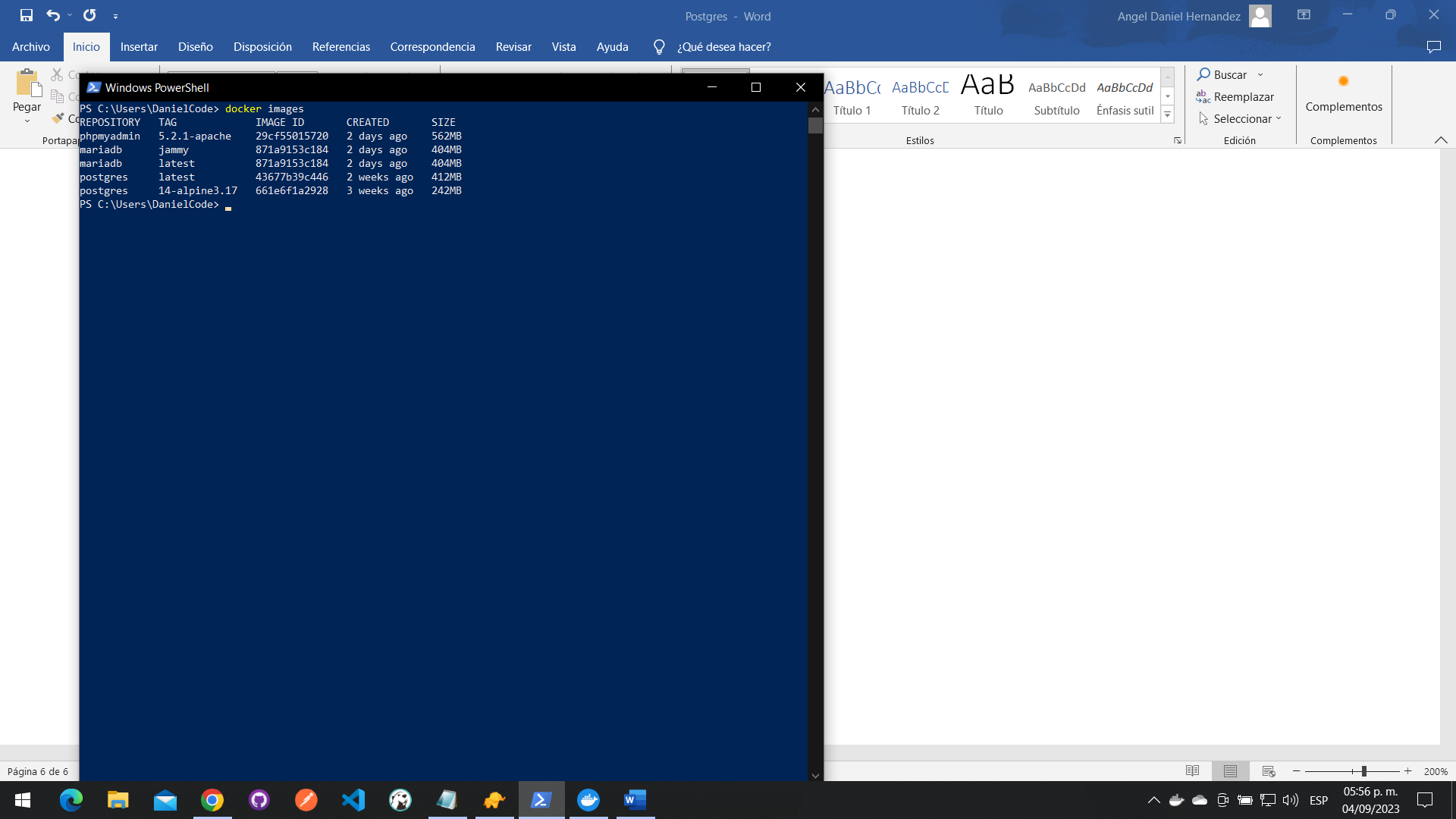
Dirígete a Docker Hub y utiliza el buscador para encontrar la imagen de phpMyAdmin. Puedes buscarla escribiendo " phpMyAdmin" en el campo de búsqueda.



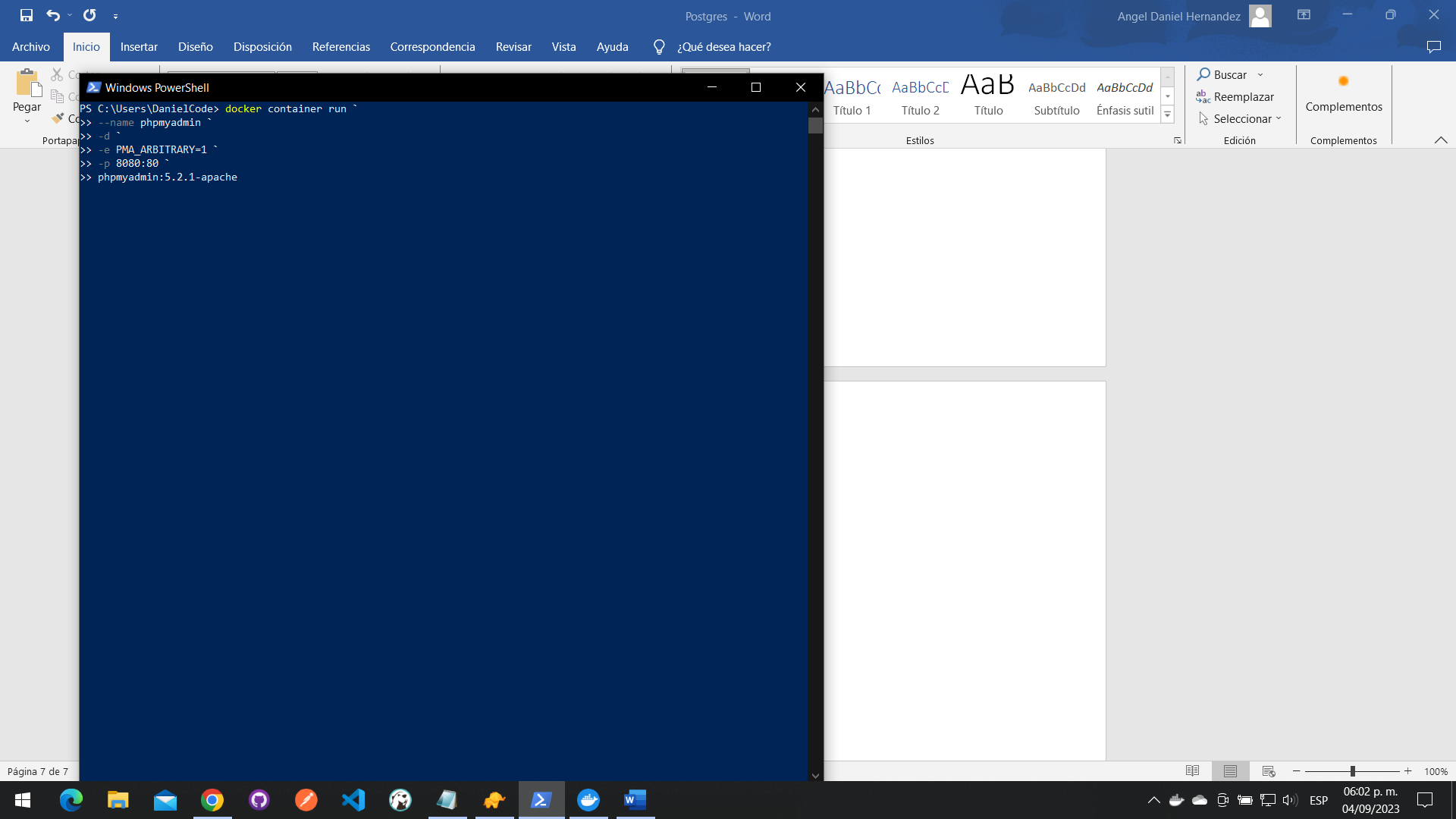
Ingresamos el siguiente comando en powershell:



Comprobamos si la imagen se descargó correctamente Ingresando el siguiente comando en powershell



Creamos el contenedor con el siguiente comando:



Este comando realiza lo siguiente:

**--name phpmyadmin**: Asigna el nombre "phpmyadmin" al contenedor que estás creando.

**-d**: Ejecuta el contenedor en segundo plano (modo daemon).

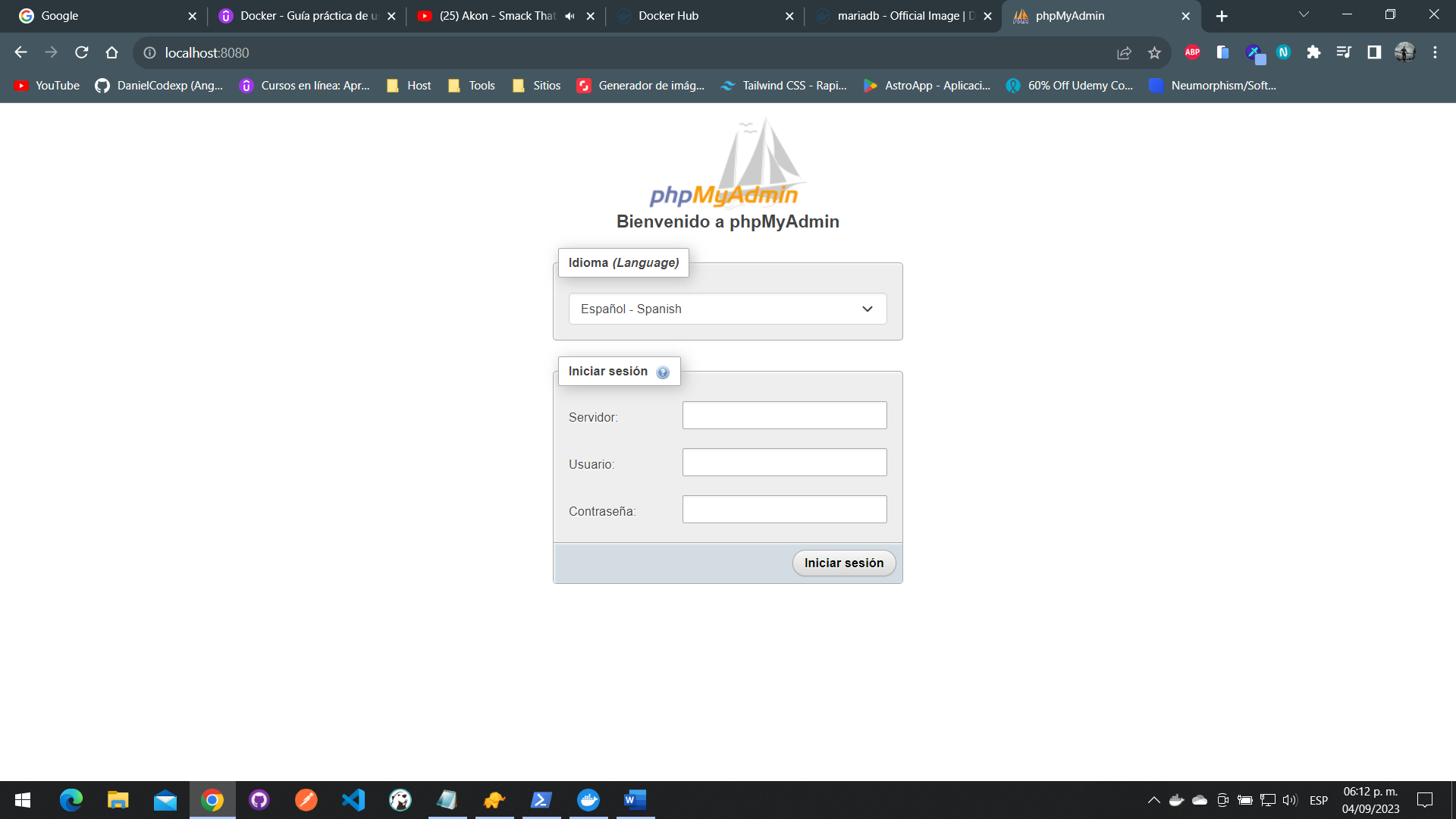
**-e PMA\_ARBITRARY=1**: Configura una variable de entorno **PMA\_ARBITRARY** en el contenedor con el valor "1". Esta variable permite configurar phpMyAdmin para conectarse a cualquier servidor de base de datos, lo que puede ser útil en situaciones donde no deseas limitar las conexiones a un servidor específico.

**-p 8080:80**: Mapea el puerto 8080 del sistema local al puerto 80 del contenedor. Esto permite acceder a phpMyAdmin desde un navegador web en tu sistema local utilizando el puerto 8080.

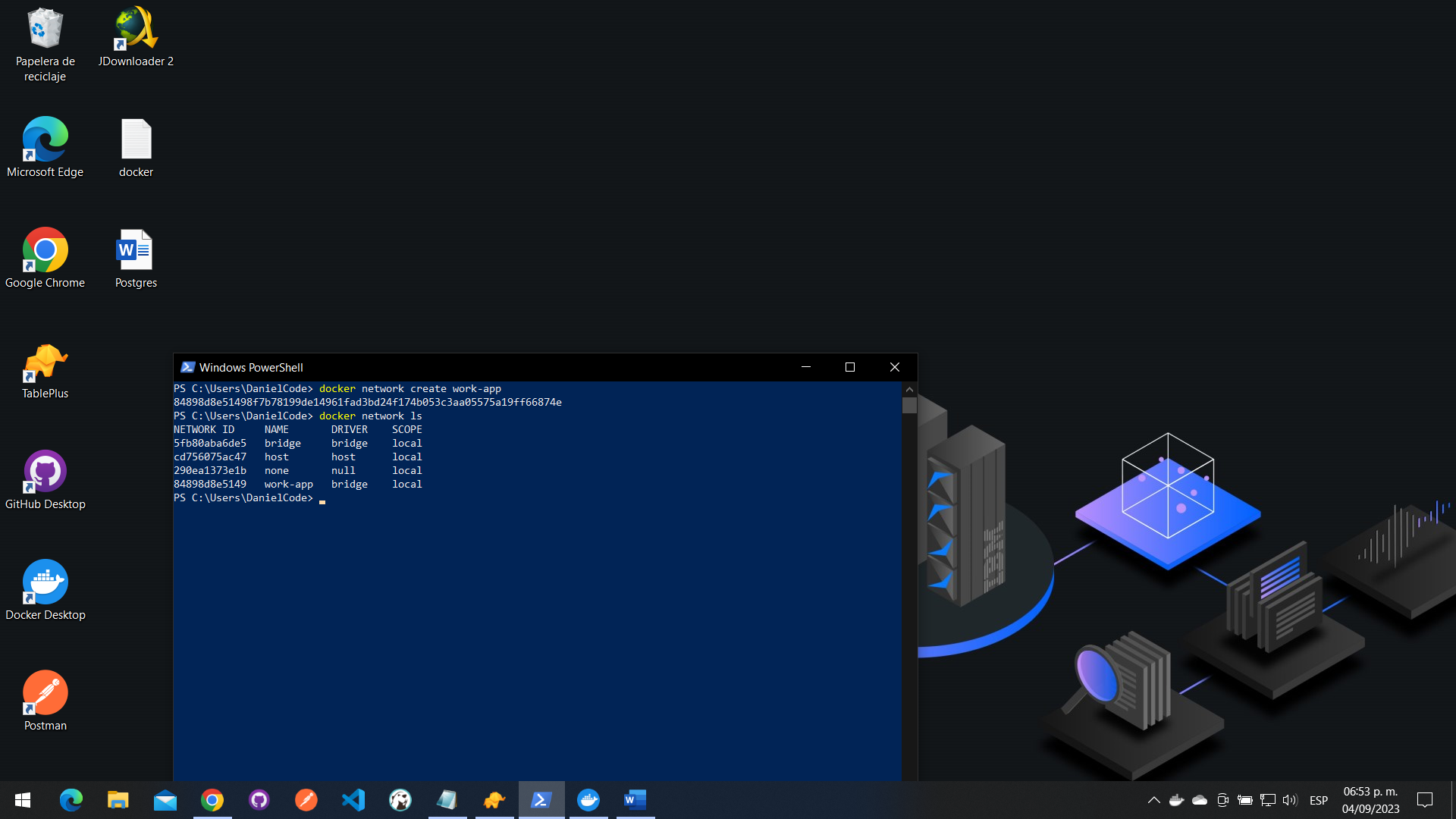
**phpmyadmin:5.2.1-apache**: Utiliza la imagen de phpMyAdmin con la etiqueta específica "5.2.1-apache" para crear el contenedor. Esta imagen contiene phpMyAdmin ejecutándose en un servidor web Apache.

Una vez que hayas ejecutado este comando, tendrás un contenedor de phpMyAdmin en funcionamiento y podrás acceder a él a través de un navegador web utilizando la dirección **http://localhost:8080** o la dirección IP de tu sistema si corresponde. Esto te permitirá administrar tus bases de datos MySQL o MariaDB de manera gráfica utilizando la interfaz de phpMyAdmin.

Principio del formulario

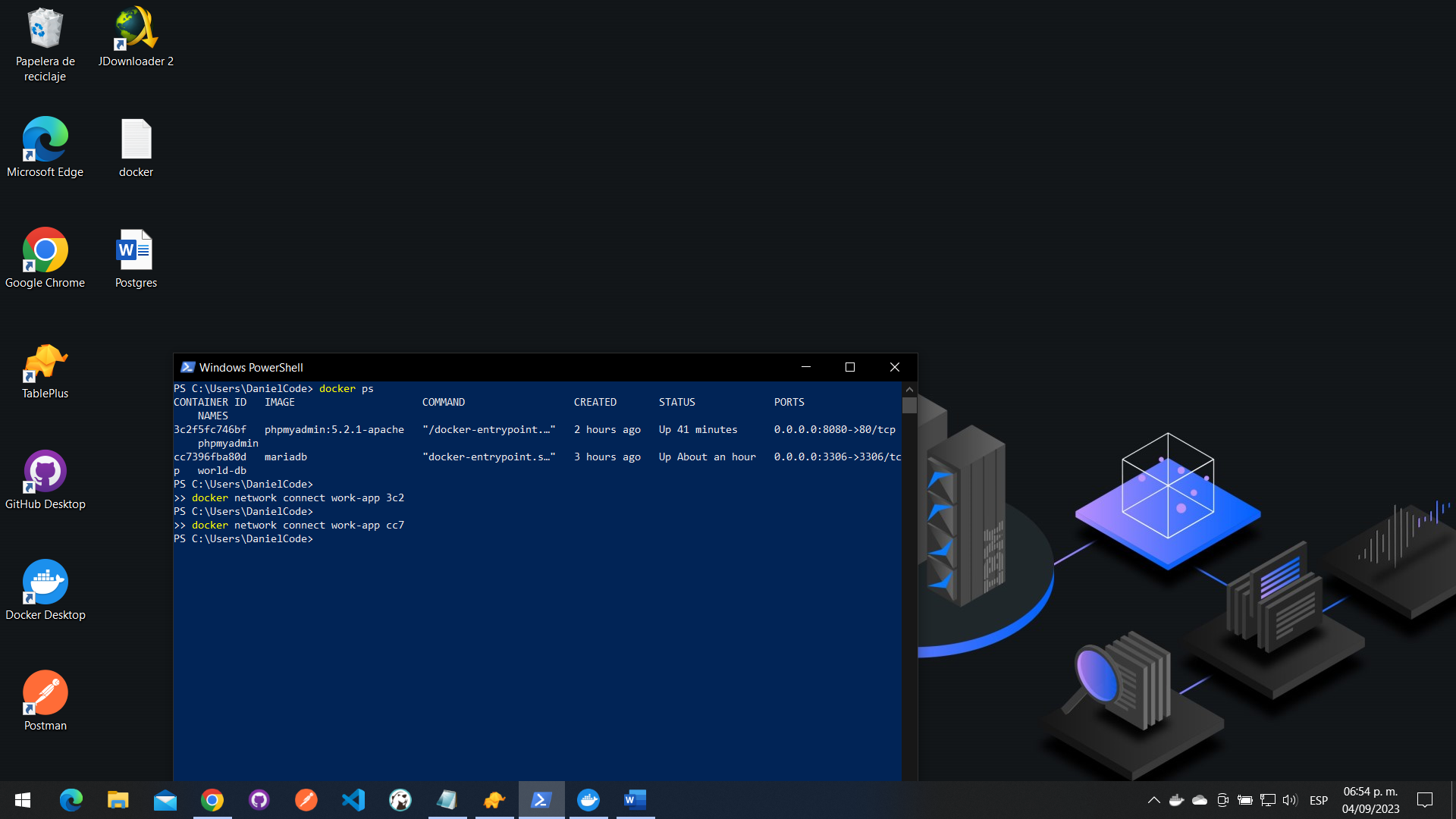


Para facilitar la interconexión de los dos contenedores previamente creados, vamos a establecer una red utilizando el siguiente comando:



Este comando creará una red llamada "work-app" que permitirá que los contenedores se comuniquen eficazmente entre sí. Esto es esencial para garantizar la interacción adecuada entre los contenedores, como el contenedor de la base de datos y el contenedor de phpMyAdmin.

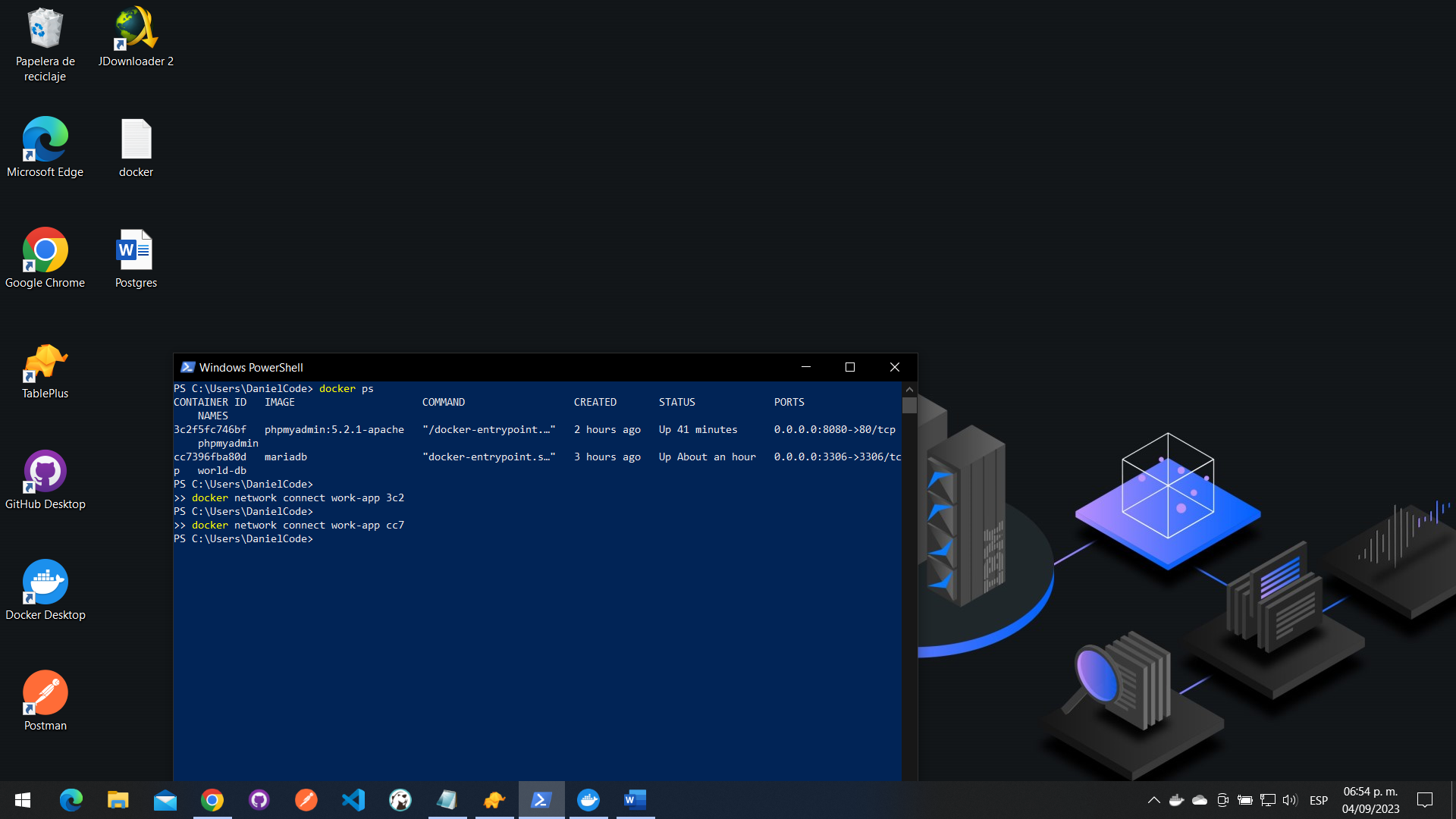
Ingresamos a la terminal de PowerShell y ejecutamos el siguiente comando:



Seguido de: docker network connect work-app <idcontainer>

Este comando se utiliza para conectar un contenedor específico a la red llamada "work-app". Debes reemplazar **<idcontainer>** con el ID del contenedor que deseas conectar a la red "work-app".

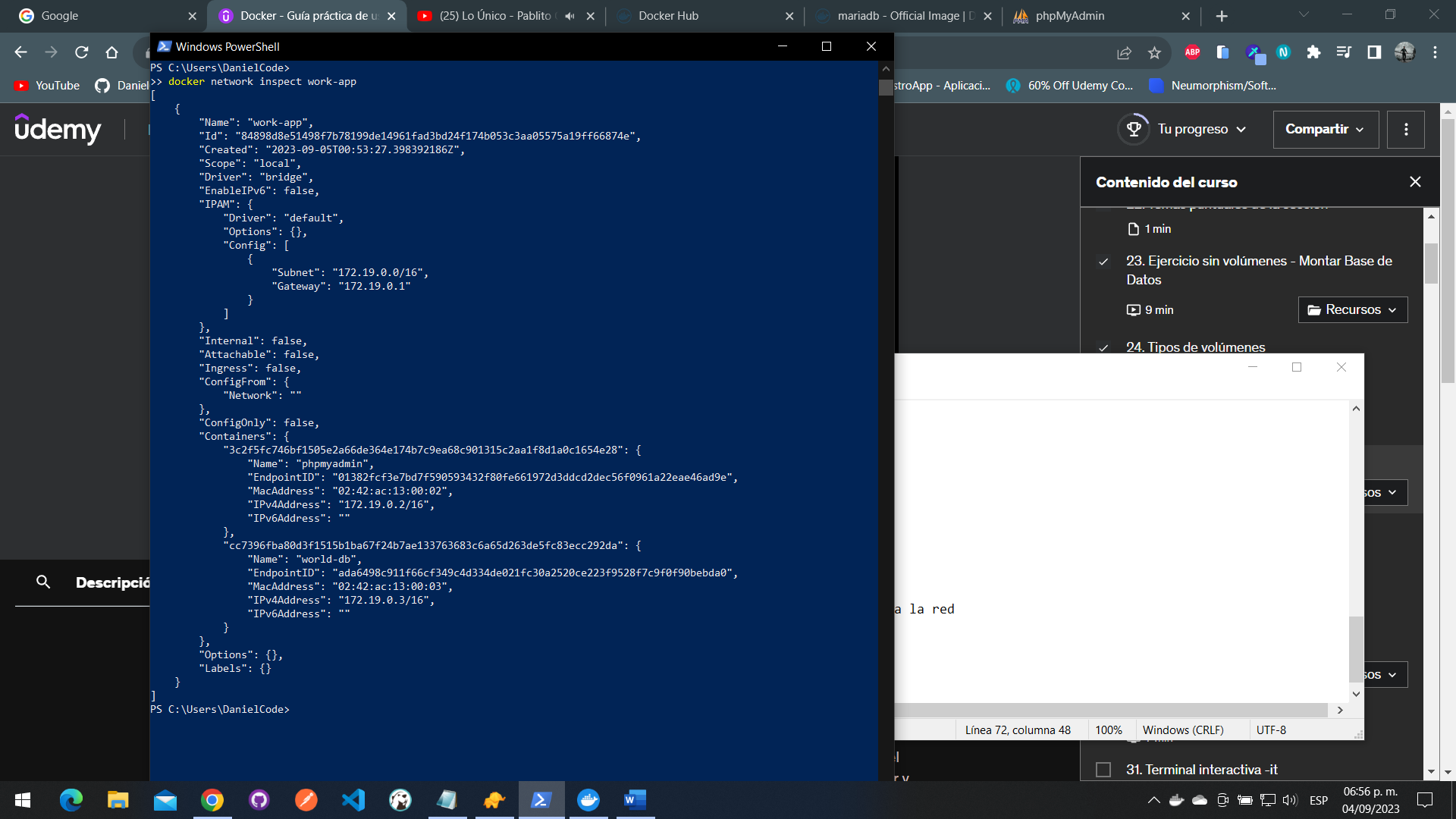
Por ejemplo, si tienes un contenedor con ID **abcd1234** y deseas conectarlo a la red "work-app", el comando se vería así:



Para inspeccionar la red, ejecutamos el siguiente comando:

docker network inspect work-app

Este comando nos proporciona detalles y configuraciones específicas de la red llamada "work-app".



utilizando la dirección [**http://localhost:8080**](http://localhost:8080)

