TAREA 2

Implementación de una arquitectura de tres capas en la nube

Zurita Cariño Emmanuel Einar

Contenido

Introducción

La implementación de una arquitectura de tres capas en la nube es un enfoque que permite desarrollar aplicaciones robustas y escalables, dividiendo las responsabilidades en capas independientes que interactúan entre sí. En esta tarea, se construirá una aplicación SPA (Single Page Application) como la capa de presentación, la cual interactúa con un servicio web REST alojado en un servidor Tomcat, constituyendo la capa de lógica de negocios. Finalmente, se emplea MySQL como el sistema de gestión de bases de datos (DBMS) para la capa de almacenamiento de datos. Esta separación de responsabilidades facilita el mantenimiento, escalabilidad y seguridad del sistema, al tiempo que permite una mejor distribución de cargas de trabajo y la integración de tecnologías específicas en cada capa. La instalación de Tomcat será la primera fase del proceso, proporcionando el entorno adecuado para ejecutar servicios web REST. Para ello, se utilizará JAX-RS, una biblioteca que permite exponer servicios web de estilo REST sobre el servidor Tomcat, junto con la integración de Jersey, que es una implementación popular de esta especificación. Además, se asegurará la compatibilidad de las bibliotecas necesarias y se integrará el soporte para JSON mediante la biblioteca Gson, permitiendo la comunicación eficiente entre el cliente SPA y el servicio REST. Una vez configurado el entorno de servidor y los servicios web, MySQL servirá como el motor de base de datos que gestionará la información de la aplicación, asegurando una capa de persistencia eficiente y estructurada. Los usuarios y permisos serán configurados de manera que solo los usuarios autorizados puedan acceder y modificar los datos. Además, se construirán y probarán tablas relacionales, como la de usuarios y sus fotos, lo que refuerza la integridad de los datos y la estructura de la base de datos. Finalmente, la configuración automática de Tomcat garantizará que el servidor web esté siempre en línea al iniciar la máquina virtual, asegurando la disponibilidad constante de los servicios. Este enfoque modular y distribuido es esencial para aprovechar las ventajas de la nube y las tecnologías web modernas, optimizando tanto el rendimiento como la seguridad en la aplicación.

Desarrollo

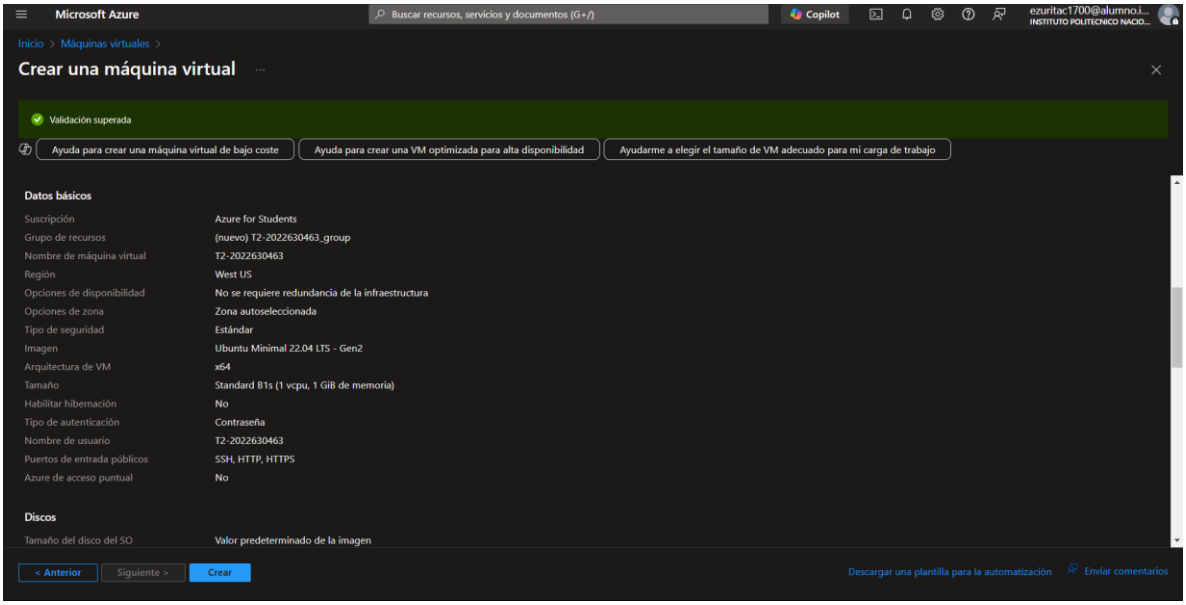
En esta tarea se va a implementar una arquitectura de tres capas en la nube, consistente en una aplicación SPA (Single Page Application), un servicio web REST para Tomcat y el DBMS MySQL.

Primeramente se instalará Tomcat y la biblioteca para el soporte de servicios web REST (JAX-RS).

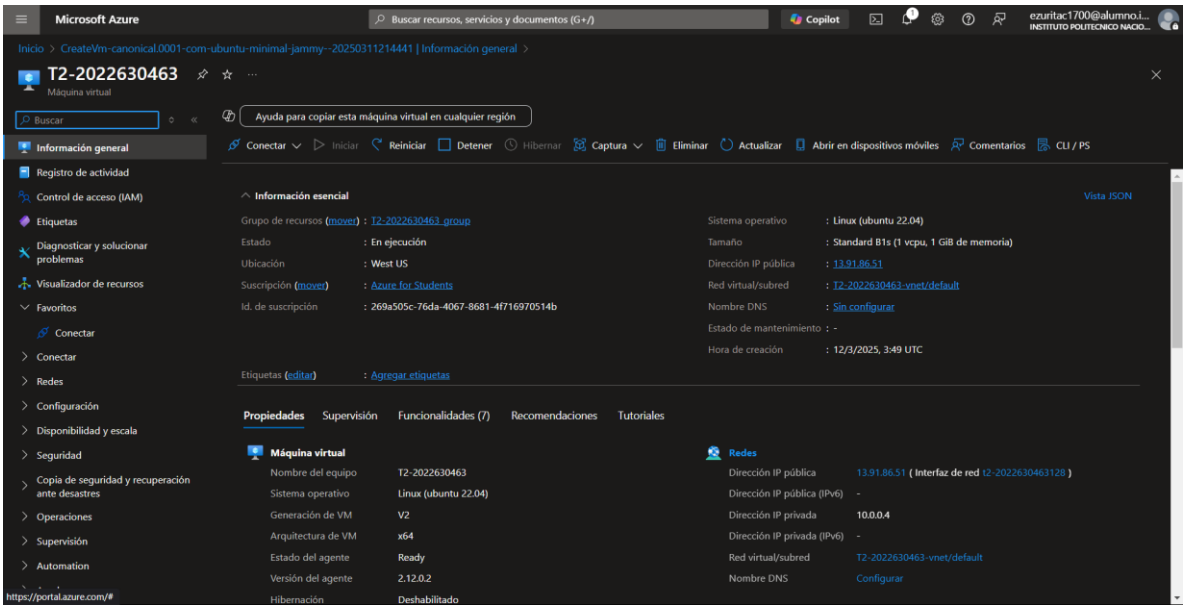
Instalación de Tomcat con soporte REST

1. Crear una máquina virtual con Ubuntu con al menos 1GB de memoria RAM. Abrir los puertos 80 y 8080 para el protocolo TCP.

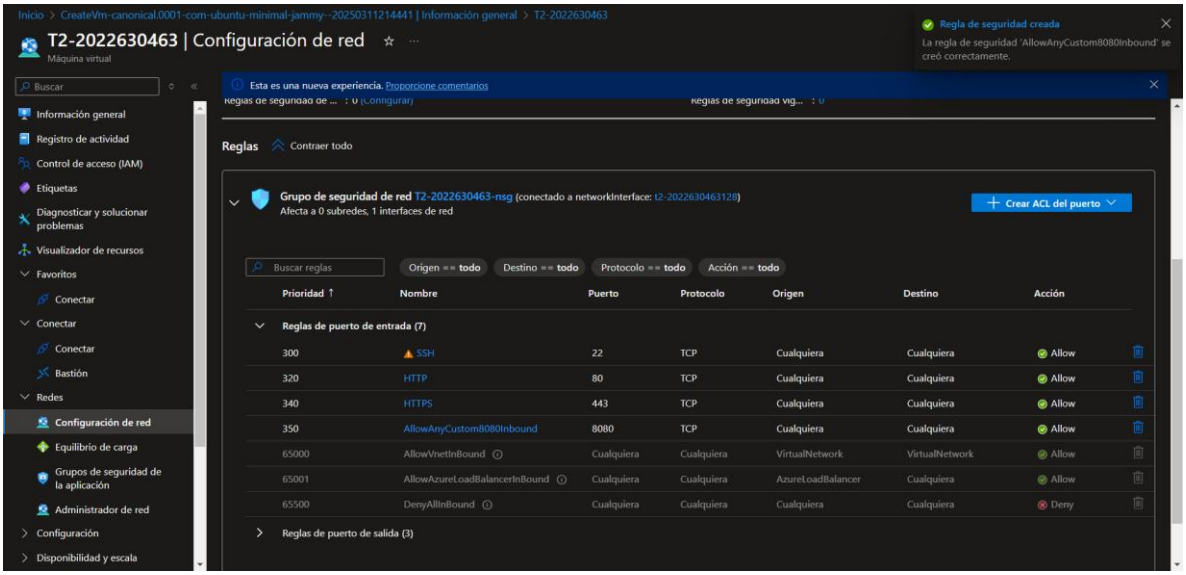
El nombre de la máquina virtual deberá iniciar con "T2-" y el número de boleta del alumno o alumna, si el número de boleta es 12345678, entonces la máquina virtual deberá llamarse: T2-12345678.



Entrada inicial



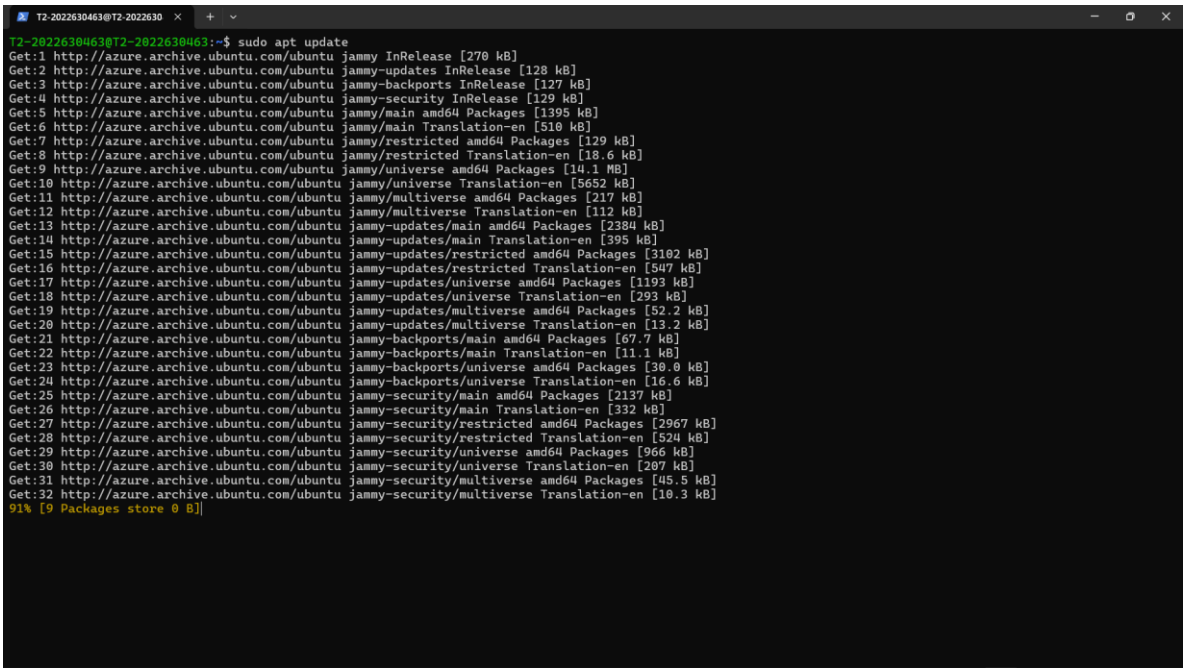
Abrir los puertos 80 y 8080



2. Instalar JDK8 ejecutando los siguientes comandos en la máquina virtual:

sudo apt update

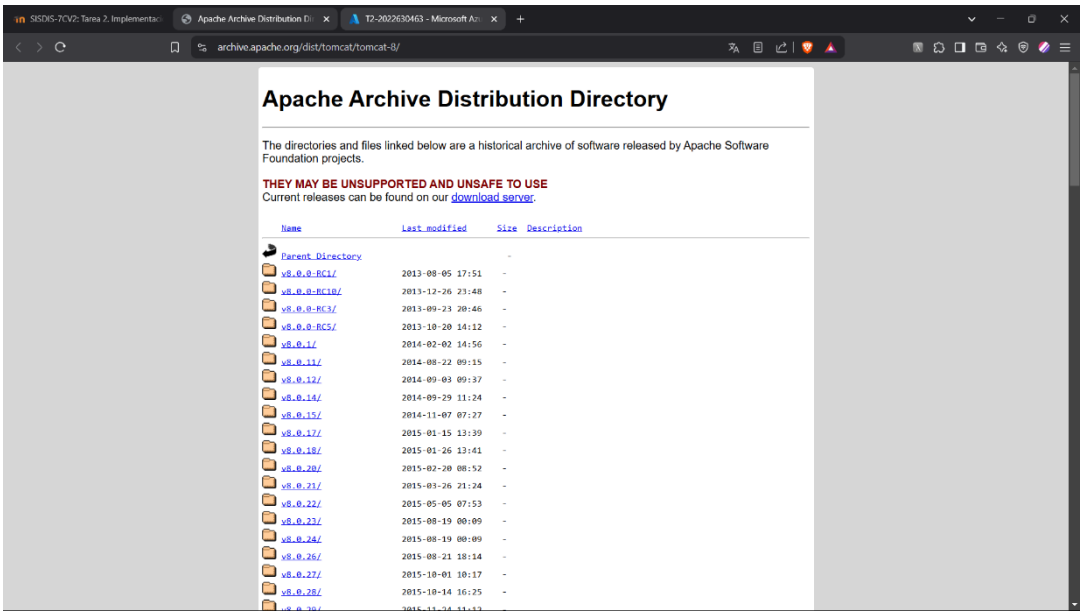
sudo apt install openjdk-8-jdk-headless



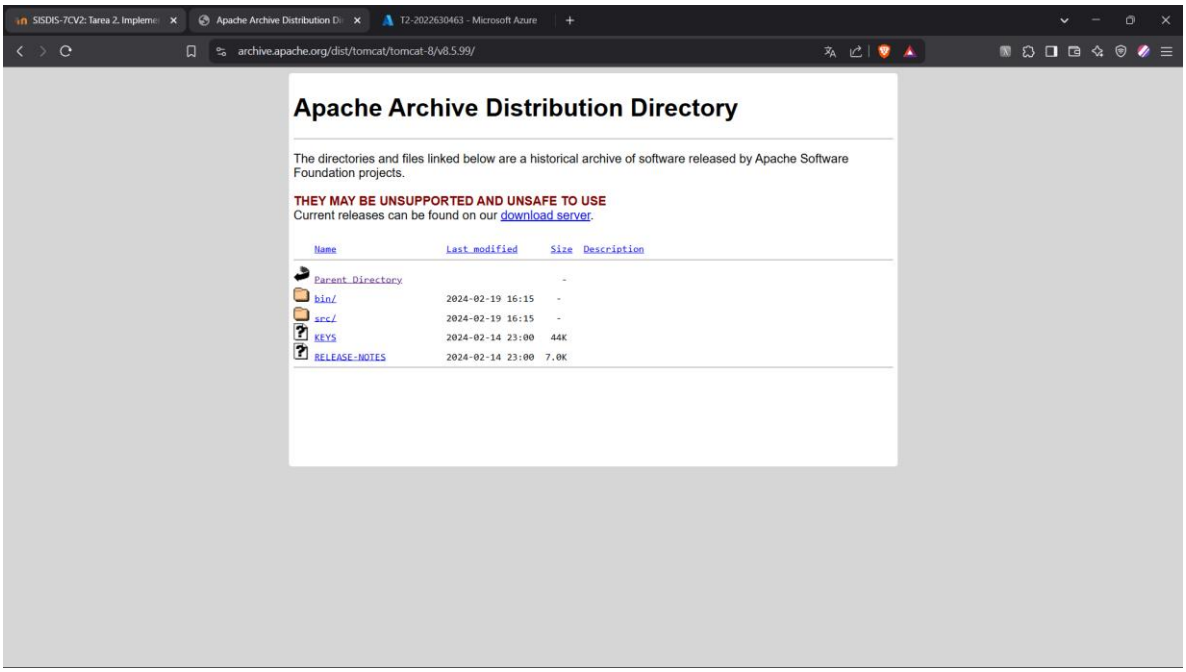
```
T2-2022630463@T2-2022630463:~$ sudo apt install openjdk-8-jdk-headless
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  ca-certificates-java fontconfig-config fonts-dejavu-core java-common libavahi-client3 libavahi-common-data libavahi-common3 libcups2 libfontconfig1
  libjpeg-turbo8 libjpeg8 liblcms2-2 libnspr4 libnss3 libpcsclite1 libx11-6 libx11-data libxau6 libxcb1 libxdmcp6 libxext6 libxi6 libxrender1 libxtst6
  openjdk-8-jre-headless x11-common
Suggested packages:
  default-jre cups-common liblcms2-utils pcscd openjdk-8-demo openjdk-8-source libnss-mdns fonts-dejavu-extra fonts-nanum fonts-ipafont-gothic
  fonts-ipafont-mincho fonts-wqy-microhei fonts-wqy-zenhei fonts-indic
The following NEW packages will be installed:
  ca-certificates-java fontconfig-config fonts-dejavu-core java-common libavahi-client3 libavahi-common-data libavahi-common3 libcups2 libfontconfig1
  libjpeg-turbo8 libjpeg8 liblcms2-2 libnspr4 libnss3 libpcsclite1 libx11-6 libx11-data libxau6 libxcb1 libxdmcp6 libxext6 libxi6 libxrender1 libxtst6
  openjdk-8-jdk-headless openjdk-8-jre-headless x11-common
0 upgraded, 27 newly installed, 0 to remove and 11 not upgraded.
Need to get 44.0 MB of archives.
After this operation, 158 MB of additional disk space will be used.
Do you want to continue? [Y/n]
```

```
T2-2022630463@T2-2022630
update-alternatives: using /usr/lib/jvm/java-8-openjdk-amd64/bin/jcmd to provide /usr/bin/jcmd (jcmd) in auto mode
update-alternatives: using /usr/lib/jvm/java-8-openjdk-amd64/bin/jdb to provide /usr/bin/jdb (jdb) in auto mode
update-alternatives: using /usr/lib/jvm/java-8-openjdk-amd64/bin/jdeps to provide /usr/bin/jdeps (jdeps) in auto mode
update-alternatives: using /usr/lib/jvm/java-8-openjdk-amd64/bin/jfr to provide /usr/bin/jfr (jfr) in auto mode
update-alternatives: using /usr/lib/jvm/java-8-openjdk-amd64/bin/jhat to provide /usr/bin/jhat (jhat) in auto mode
update-alternatives: using /usr/lib/jvm/java-8-openjdk-amd64/bin/jinfo to provide /usr/bin/jinfo (jinfo) in auto mode
update-alternatives: using /usr/lib/jvm/java-8-openjdk-amd64/bin/jmap to provide /usr/bin/jmap (jmap) in auto mode
update-alternatives: using /usr/lib/jvm/java-8-openjdk-amd64/bin/jps to provide /usr/bin/jps (jps) in auto mode
update-alternatives: using /usr/lib/jvm/java-8-openjdk-amd64/bin/jrunscript to provide /usr/bin/jrunscript (jrunscript) in auto mode
update-alternatives: using /usr/lib/jvm/java-8-openjdk-amd64/bin/jsadebugd to provide /usr/bin/jsadebugd (jsadebugd) in auto mode
update-alternatives: using /usr/lib/jvm/java-8-openjdk-amd64/bin/jstack to provide /usr/bin/jstack (jstack) in auto mode
update-alternatives: using /usr/lib/jvm/java-8-openjdk-amd64/bin/jstat to provide /usr/bin/jstat (jstat) in auto mode
update-alternatives: using /usr/lib/jvm/java-8-openjdk-amd64/bin/jstatd to provide /usr/bin/jstatd (jstatd) in auto mode
update-alternatives: using /usr/lib/jvm/java-8-openjdk-amd64/bin/native2ascii to provide /usr/bin/native2ascii (native2ascii) in auto mode
update-alternatives: using /usr/lib/jvm/java-8-openjdk-amd64/bin/rmic to provide /usr/bin/rmic (rmic) in auto mode
update-alternatives: using /usr/lib/jvm/java-8-openjdk-amd64/bin/schemagen to provide /usr/bin/schemagen (schemagen) in auto mode
update-alternatives: using /usr/lib/jvm/java-8-openjdk-amd64/bin/serialver to provide /usr/bin/serialver (serialver) in auto mode
update-alternatives: using /usr/lib/jvm/java-8-openjdk-amd64/bin/wsgen to provide /usr/bin/wsgen (wsgen) in auto mode
update-alternatives: using /usr/lib/jvm/java-8-openjdk-amd64/bin/wsimport to provide /usr/bin/wsimport (wsimport) in auto mode
update-alternatives: using /usr/lib/jvm/java-8-openjdk-amd64/bin/xjc to provide /usr/bin/xjc (xjc) in auto mode
Processing triggers for libc-bin (2.35-0ubuntu3.9) ...
Processing triggers for ca-certificates (20240203~22.04.1) ...
Updating certificates in /etc/ssl/certs...
0 added, 0 removed; done.
Running hooks in /etc/ca-certificates/update.d...

done.
done.
debconf: unable to initialize frontend: Dialog
debconf: (No usable dialog-like program is installed, so the dialog based frontend cannot be used. at /usr/share/perl5/Debconf/FrontEnd/Dialog.pm line 78.)
debconf: falling back to frontend: Readline
Scanning processes...
Scanning linux images...

Running kernel seems to be up-to-date.

No services need to be restarted.

No containers need to be restarted.

No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this host.
T2-2022630463@T2-2022630463:~$
```

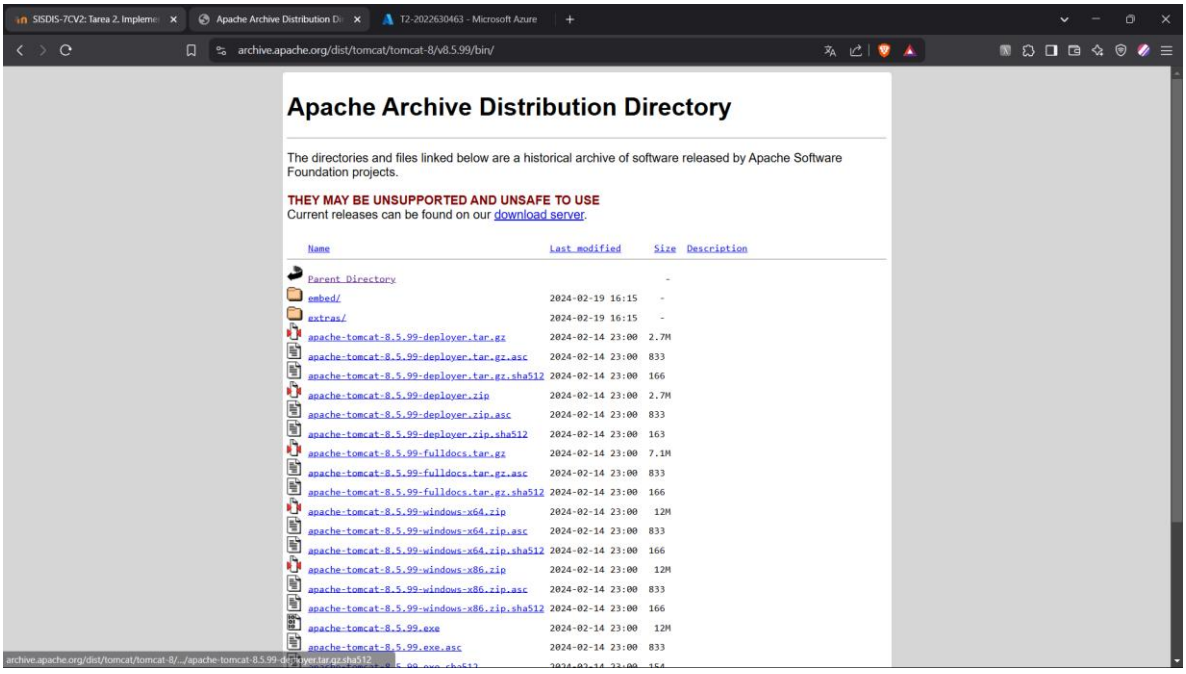
3. Descargar la distribución binaria de Tomcat 8 (p.e. apache-tomcat-8.5.99.zip en el directorio v8.5.99/bin) de la siguiente URL: https://archive.apache.org/dist/tomcat/tomcat-8/



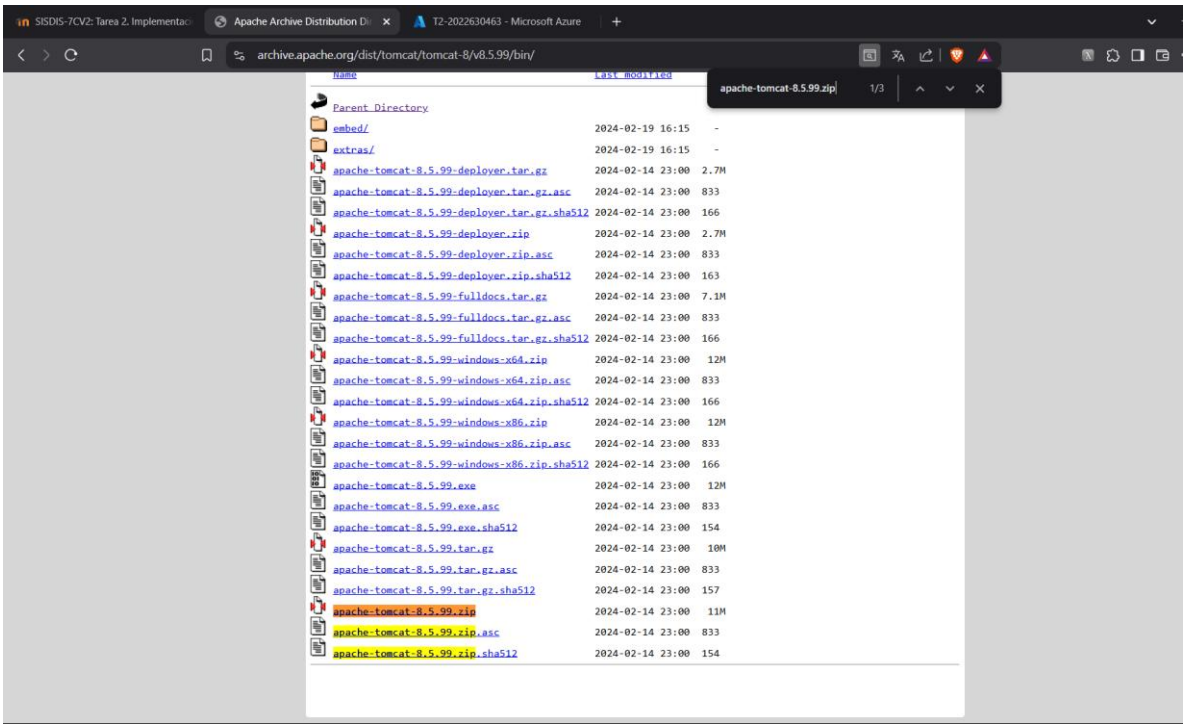
Al ingresar a la carpeta v8.5.99 encontramos estas carpetas:



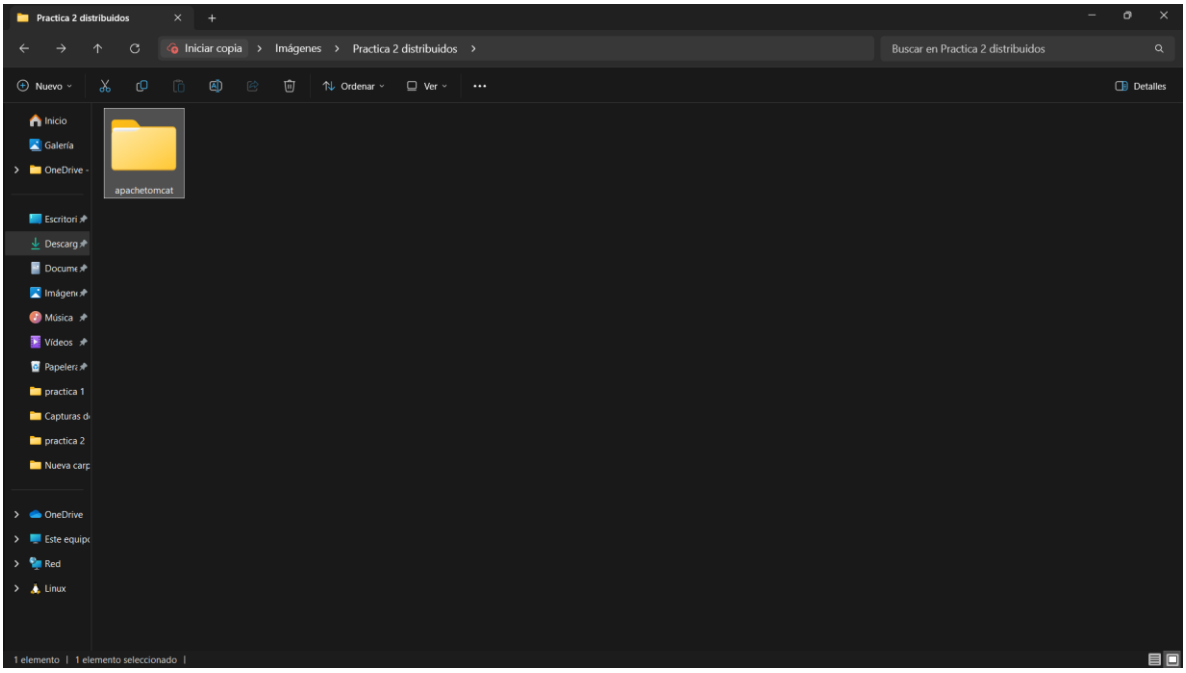
Accedemos a la carpeta bin:



Ya ingresando a la carpeta bin buscaremos un zip de nombre: apache-tomcat-8.5.99.zip y lo descargamos

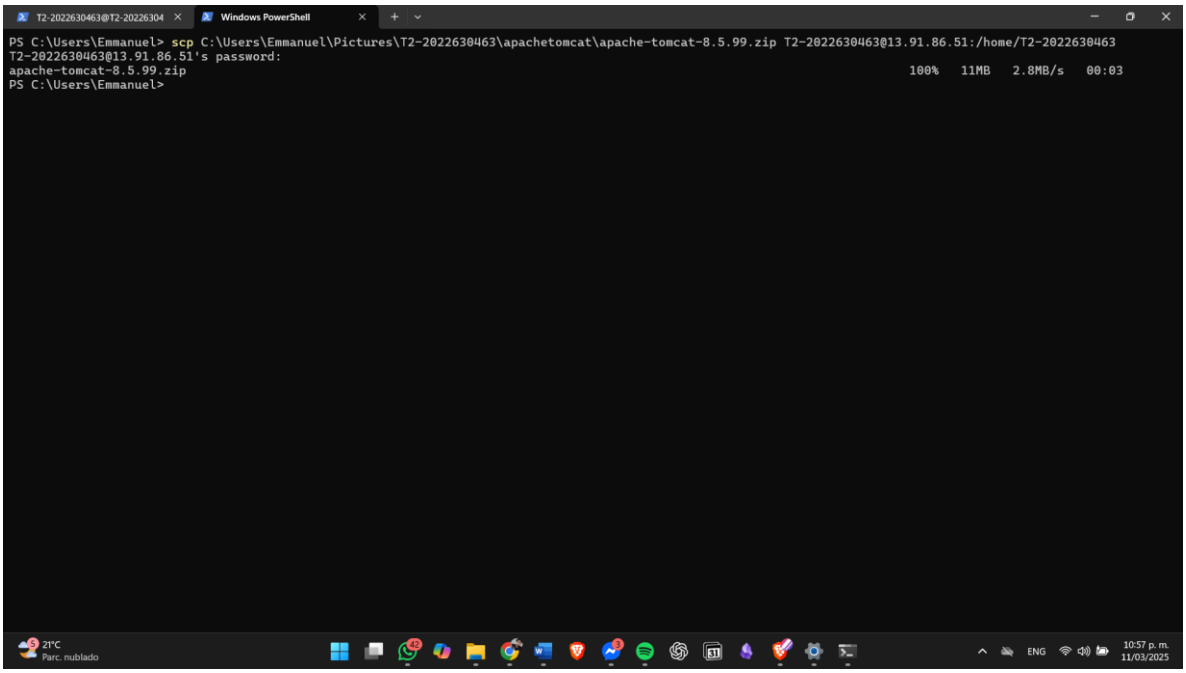


Lo ubicaré en una carpeta específica en mi máquina host (Windows)

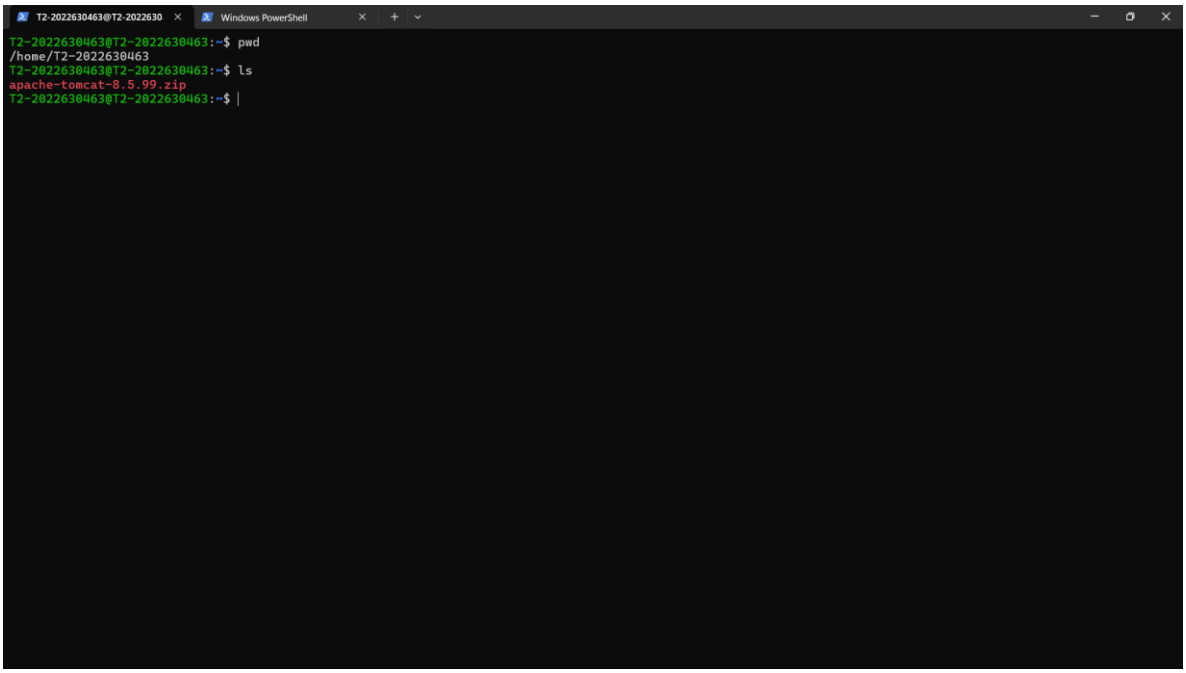


Ahora, copiaré el archivo zip ubicado en mi máquina host a la máquina virtual, esto se hará utilizando el comando SCP (secure copy) el cual utiliza ssh, la sintaxis del comando es:

scp C:\ruta\al\archivo usuario@IP-ubuntu:/ruta/destino

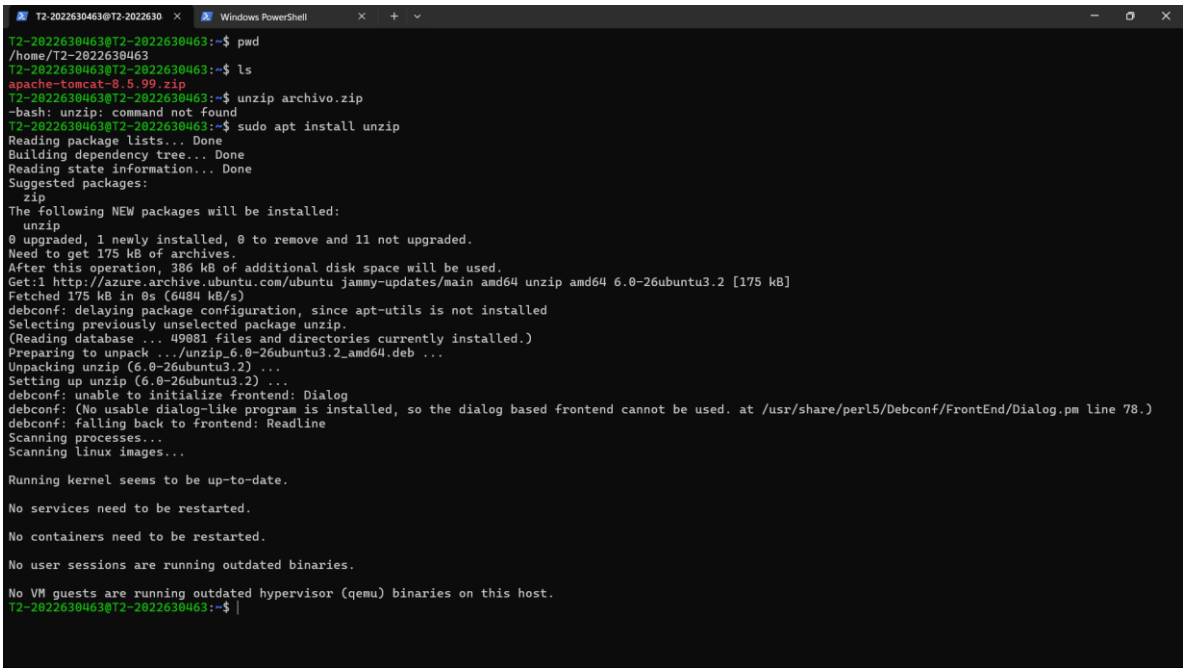


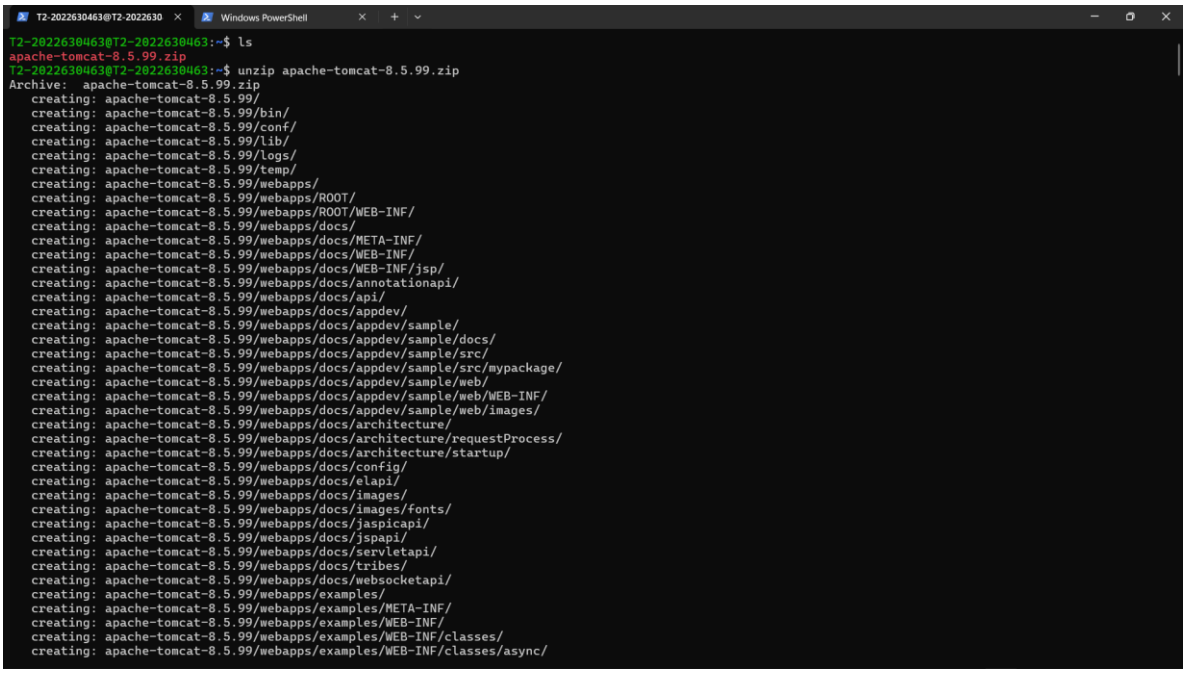
Ahora utilizamos **ls** para listar los archivos localizados en mi ruta actual. Y observamos que ya se encuentra el archivo .zip ubicado en mi máquina virtual.



Ahora utilizamos el comando **unzip** para descomprimir el archivo.

Primero instalamos los paquetes de unzip.

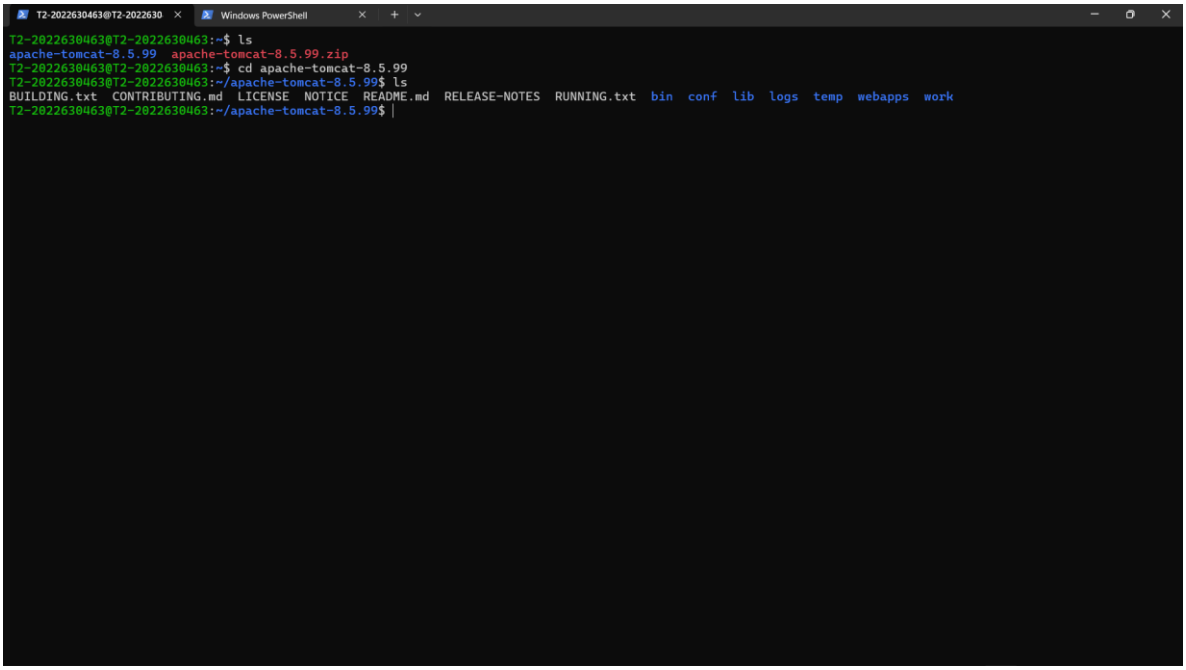




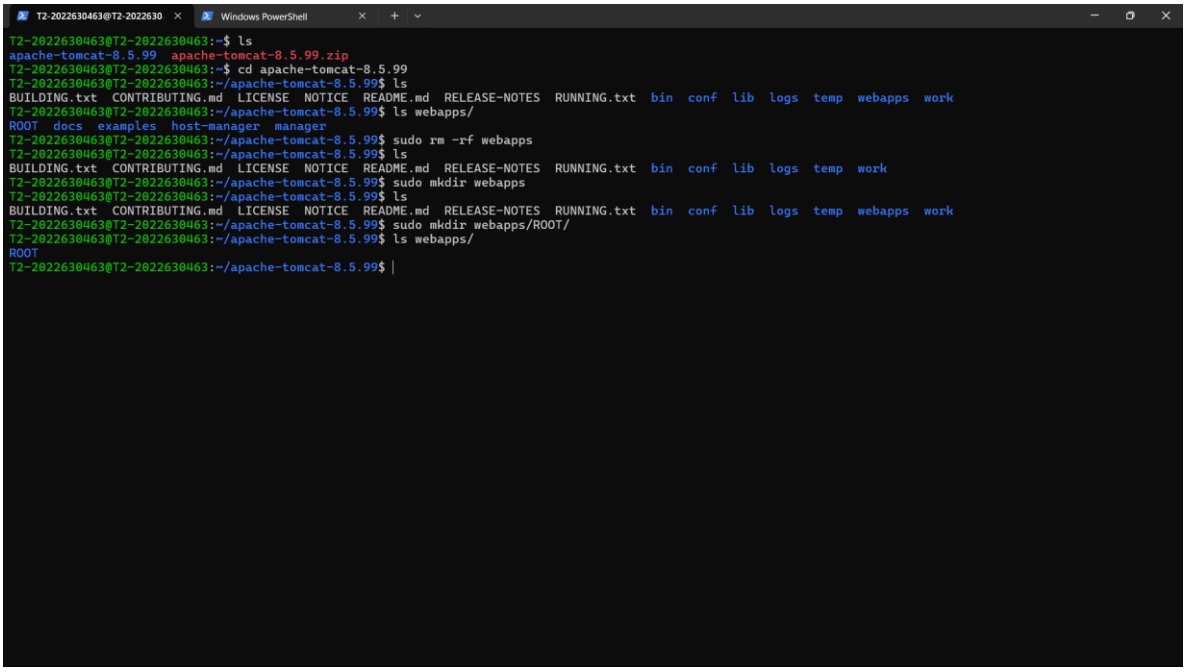
5. Eliminar el directorio webapps el cual se encuentra dentro del directorio de Tomcat. Crear un nuevo directorio webapps y dentro de éste crear el directorio ROOT.

NOTA DE SEGURIDAD: Lo anterior se recomienda debido a que se han detectado vulnerabilidades en algunas aplicaciones que vienen con Tomcat, estas aplicaciones se encuentran originalmente instaladas en los directorios webapps y webapps/ROOT.

Buscamos el directorio webapp dentro de la carpeta Tomcat que se generó al momento de descomprimirlo

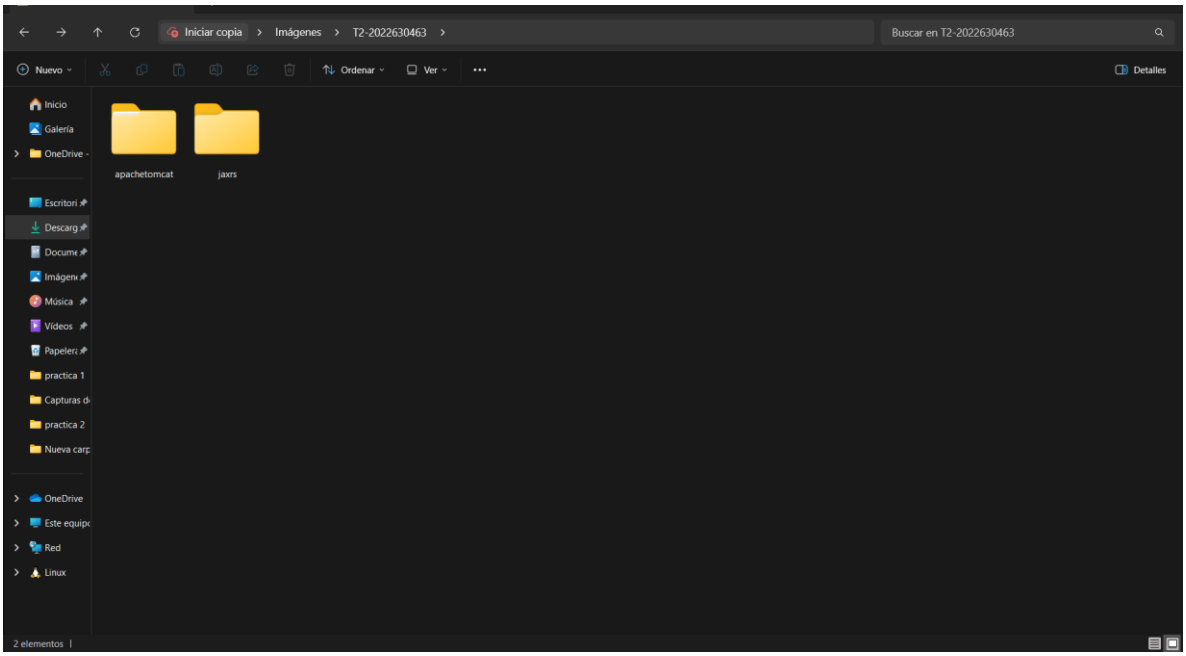


Accedemos al directorio webapps para comprobar si se encuentra el directorio de ROOT. Por lo que procederemos a borrar el directorio webapps, para luego crear el directorio Webapps y en su interior el directorio ROOT.
Utilizaré el comando **sudo rm -rf webapps** el cual eliminará el directorio webapps con todo lo que esté en su interior.
Con el comando **sudo mkdir** creamos los nuevos directorios.



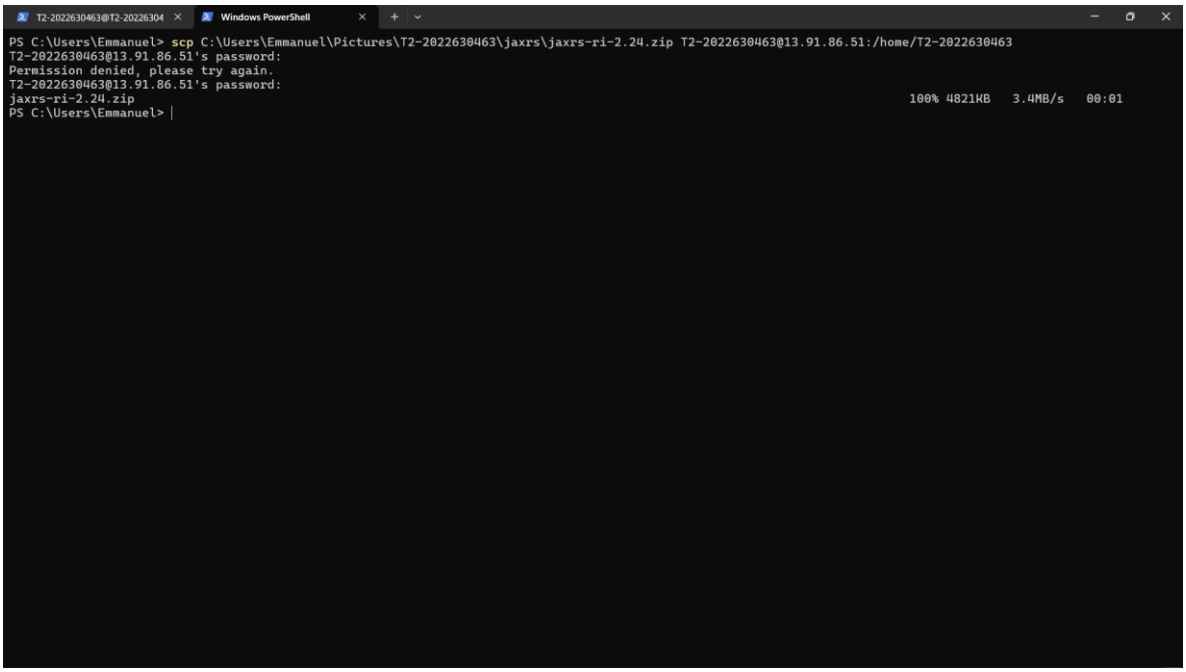
6. Descargar la biblioteca "Jersey" de la siguiente URL. Jersey es una implementación de JAX-RS la cual permite ejecutar servicios web estilo REST sobre Tomcat:

https://repo1.maven.org/maven2/org/glassfish/jersey/bundles/jaxrs-ri/2.24/jaxrs-ri-2.24.zip

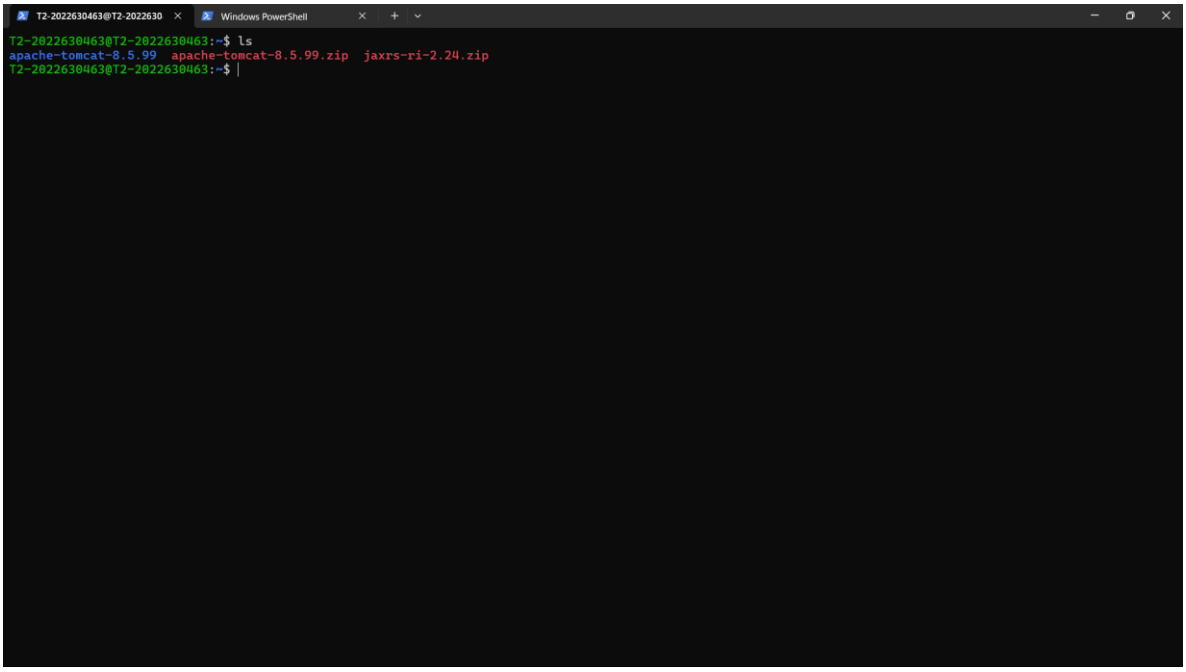


Lo almacenamos nuevamente en una ruta específica, en este caso yo lo almacené en la misma carpeta donde almacené tomcat.

Aplicaremos el mismo procedimiento para copiar el zip de jersey a la máquina virtual.

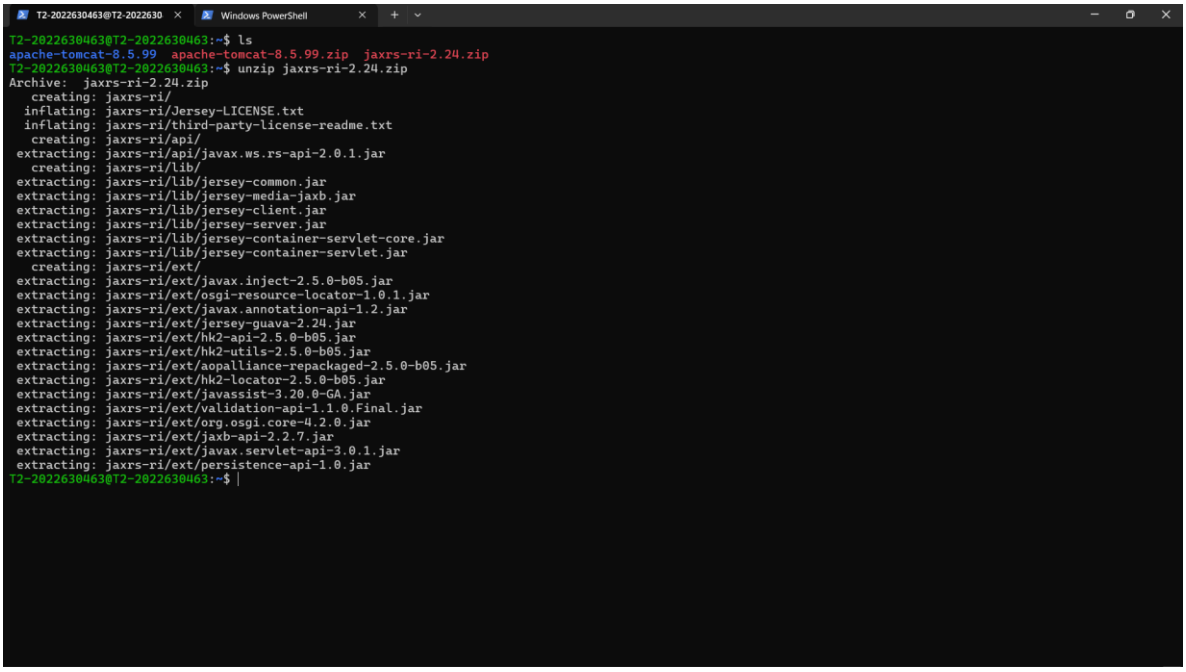


Comprobamos que el archive fue recibido correctamente.

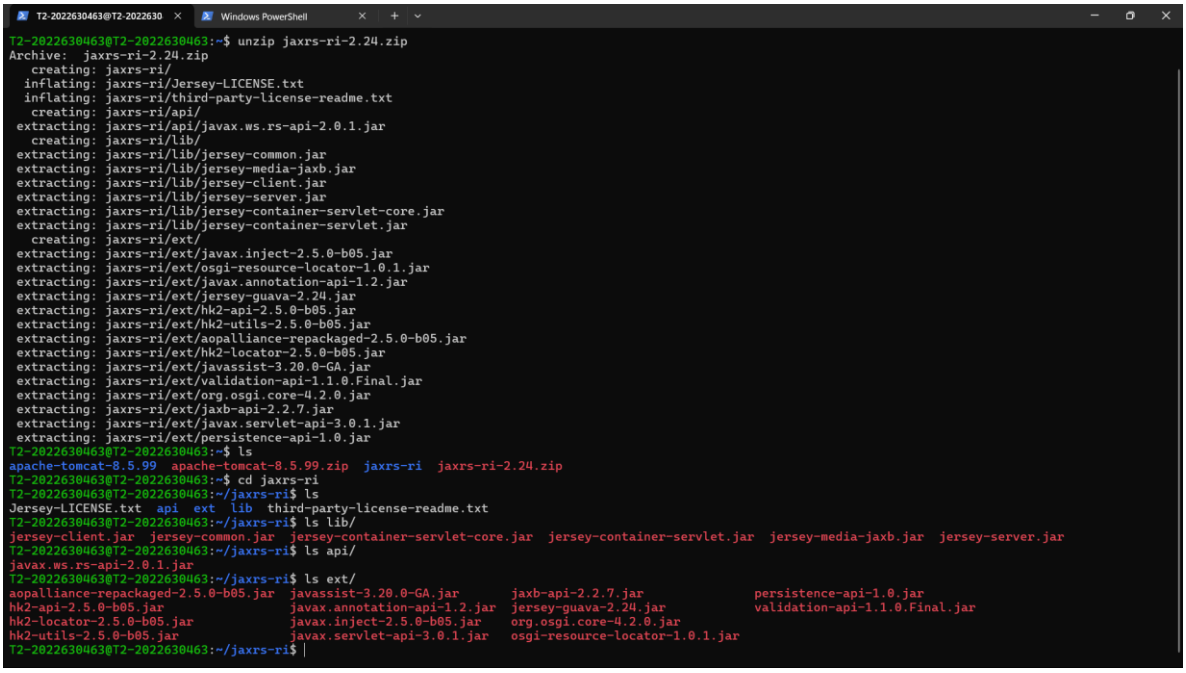


7. Copiar a la máquina virtual el archivo descargado anteriormente, desempacarlo y **copiar todos los archivos** con extensión “.jar” los cuales se encuentran en los directorios desempacados, al directorio "lib" de Tomcat.

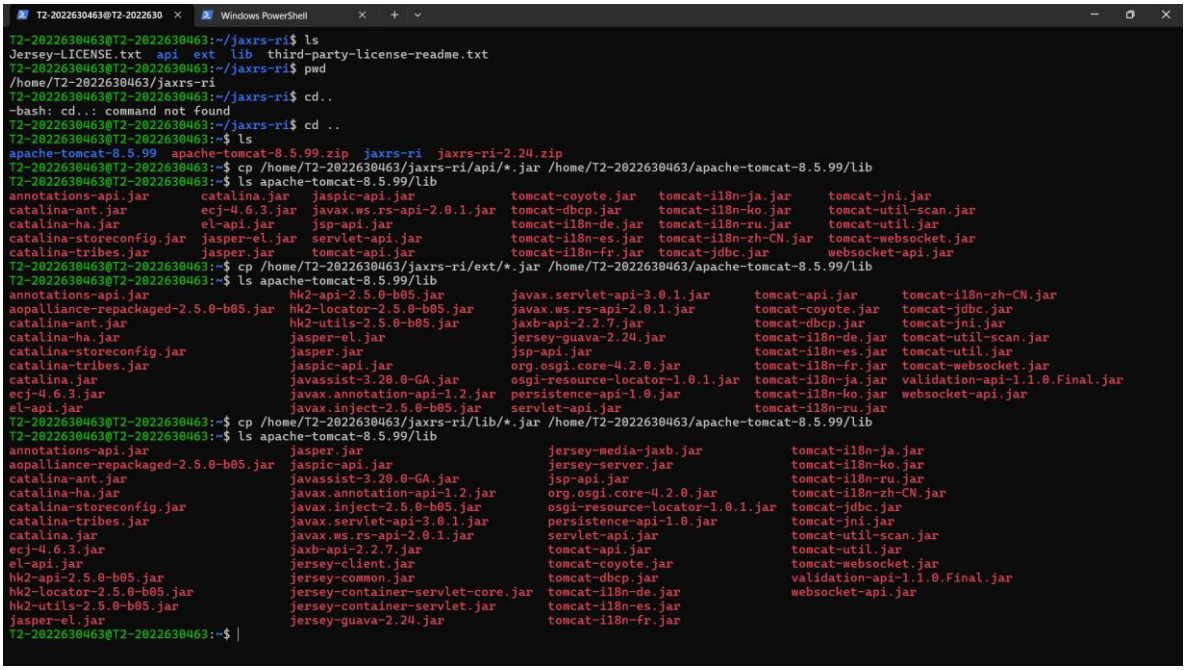
Desempaquetado del archivo .zip



Accedemos al directorio donde se extrajo el contenido del archivo zip. Y comprobamos que dentro de cada directorio existan archivos .jar para ser copiados al directorio lib de tomcat

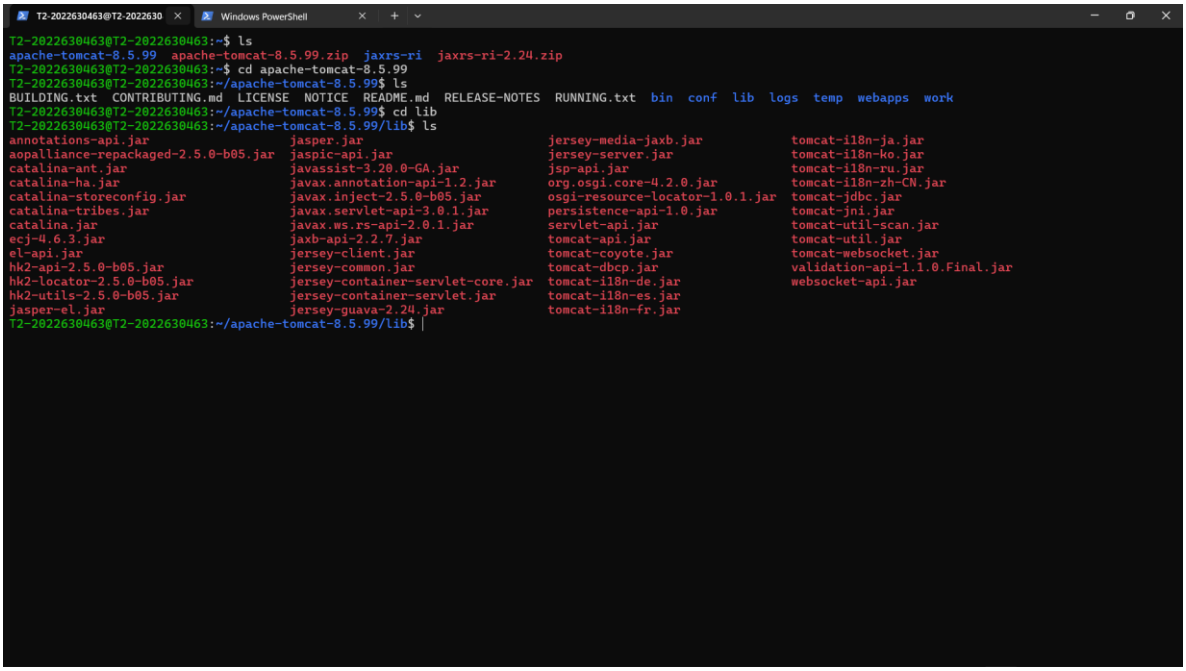


Para copiar los archivos .jar ubicados en cada carpeta utilizamos el comando **cp**
cp /ruta/origen/*.jar /ruta/destino/

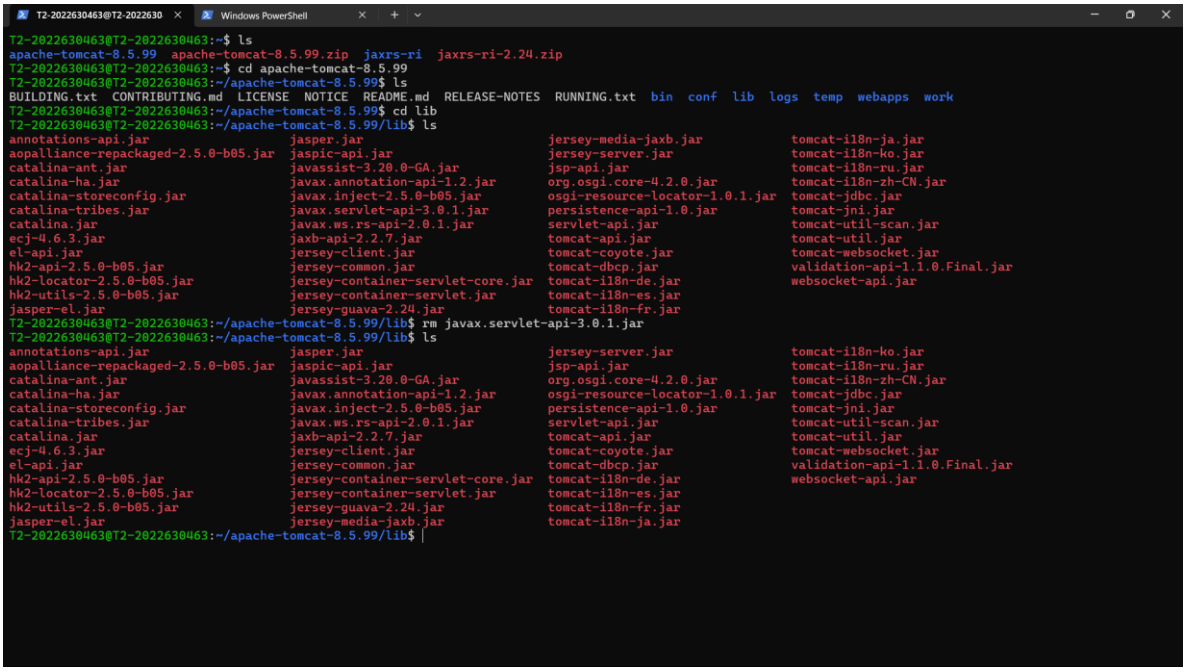


8. Borrar el archivo javax.servlet-api-3.0.1.jar del directorio "lib" de Tomcat (esto debe hacerse ya que existe una incompatibilidad entre Tomcat y Jersey).

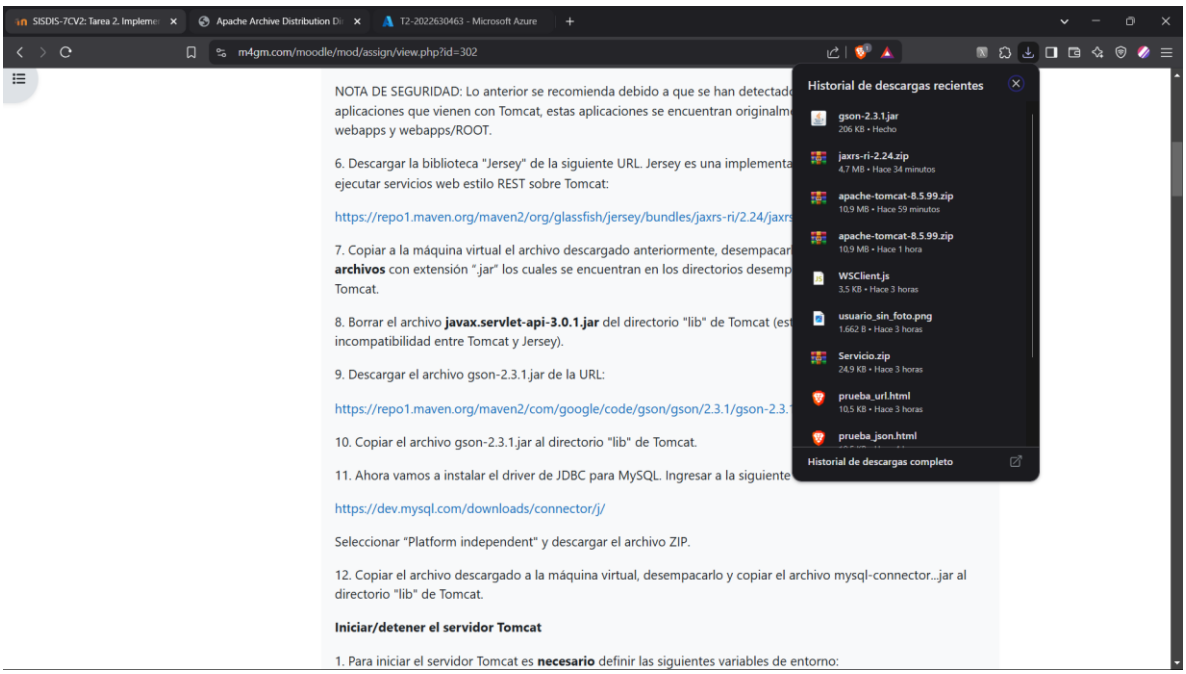
Actualmente se muestra así el listado de archivos almacenados en el directorio lib de tomcat



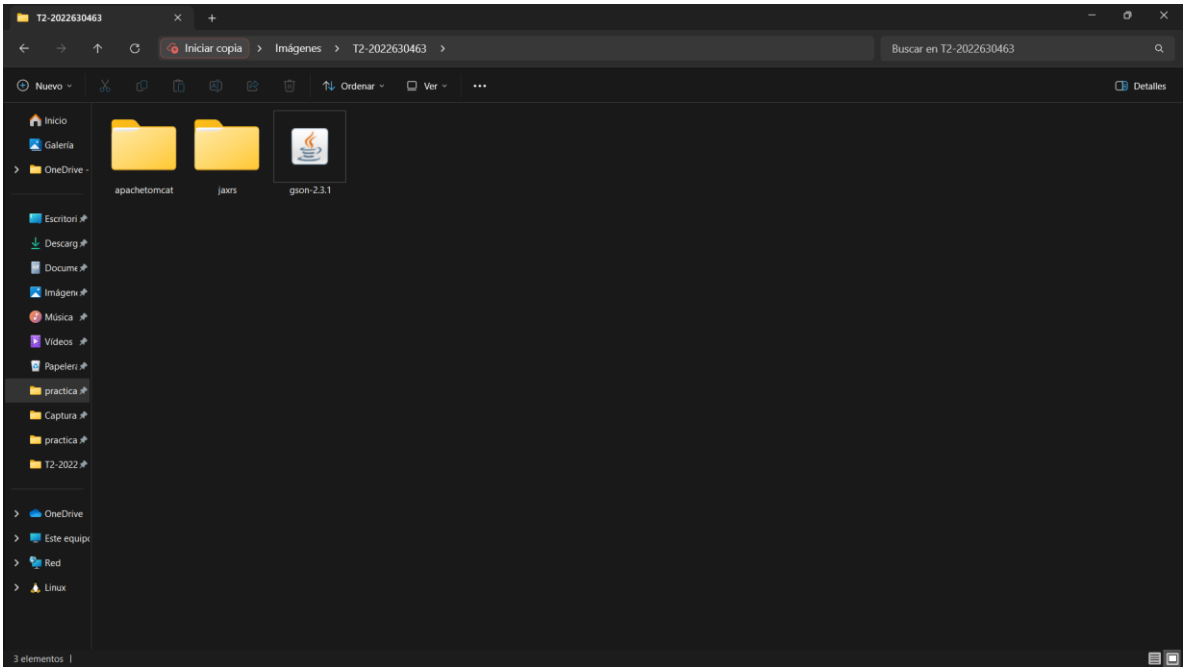
Para eliminar el archivo javax.servlet-api-3.0.1.jar usamos el comando **rm nombre_archivo**



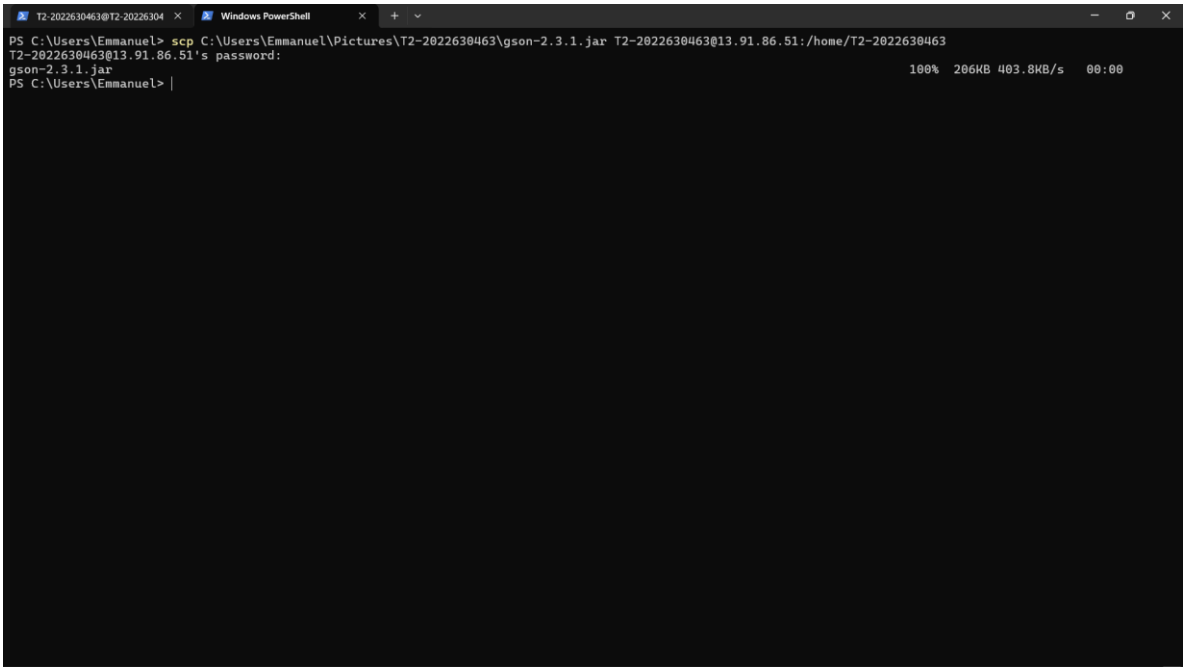
9. Descargar el archivo gson-2.3.1.jar de la URL:
https://repo1.maven.org/maven2/com/google/code/gson/gson/2.3.1/gson-2.3.1.jar



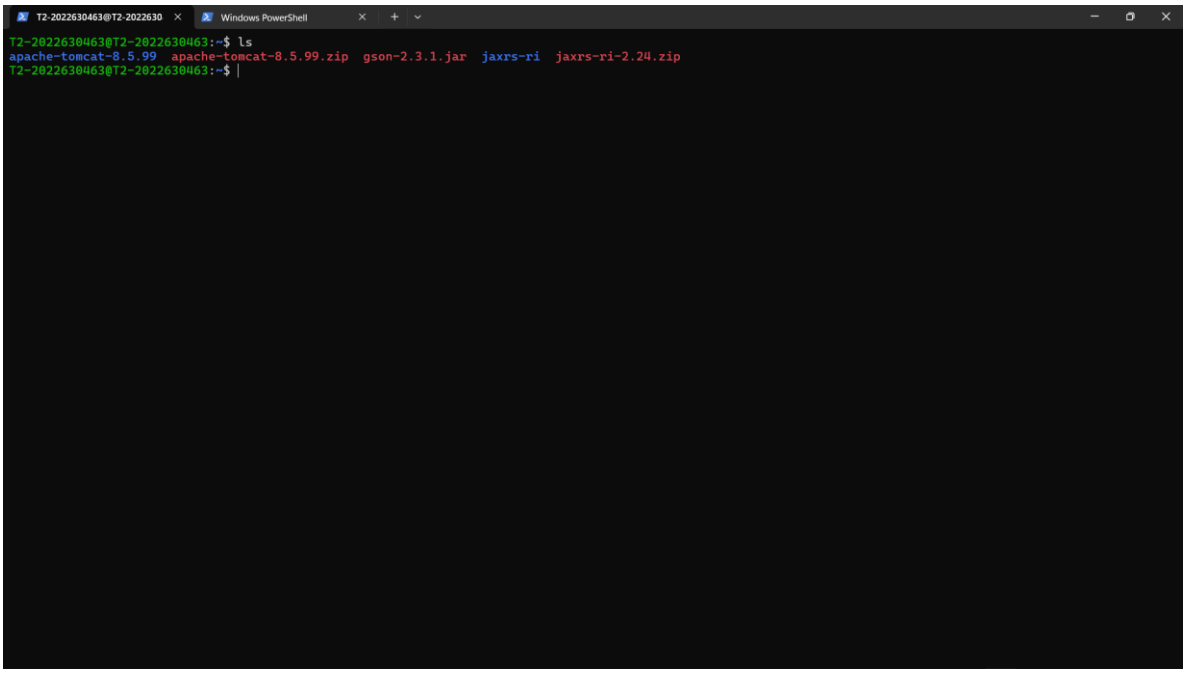
Lo ubicamos en la misma carpeta donde almacenamos las carpetas anteriores.



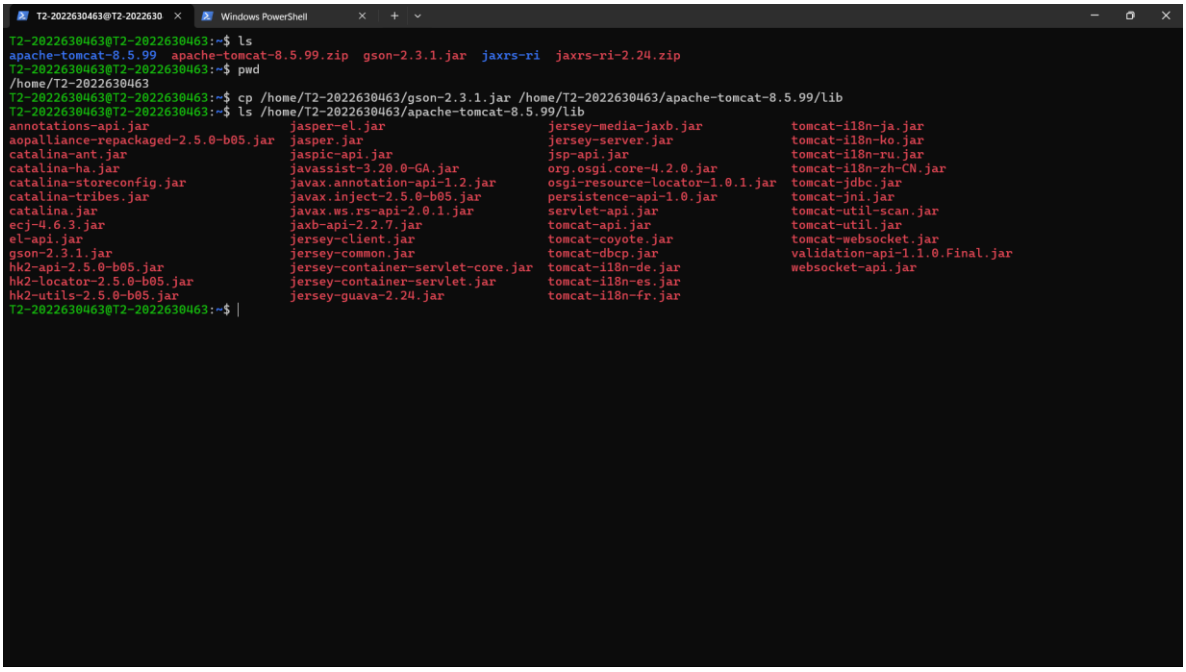
Lo copiamos a la máquina virtual



Comprobamos que se copió correctamente.

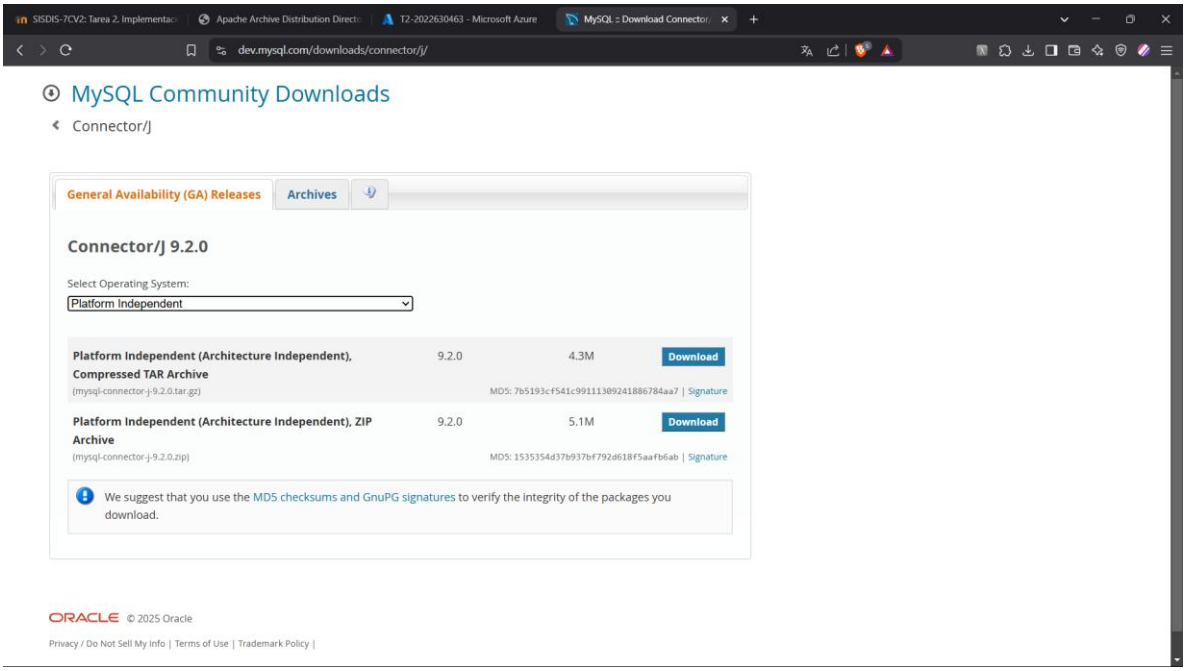


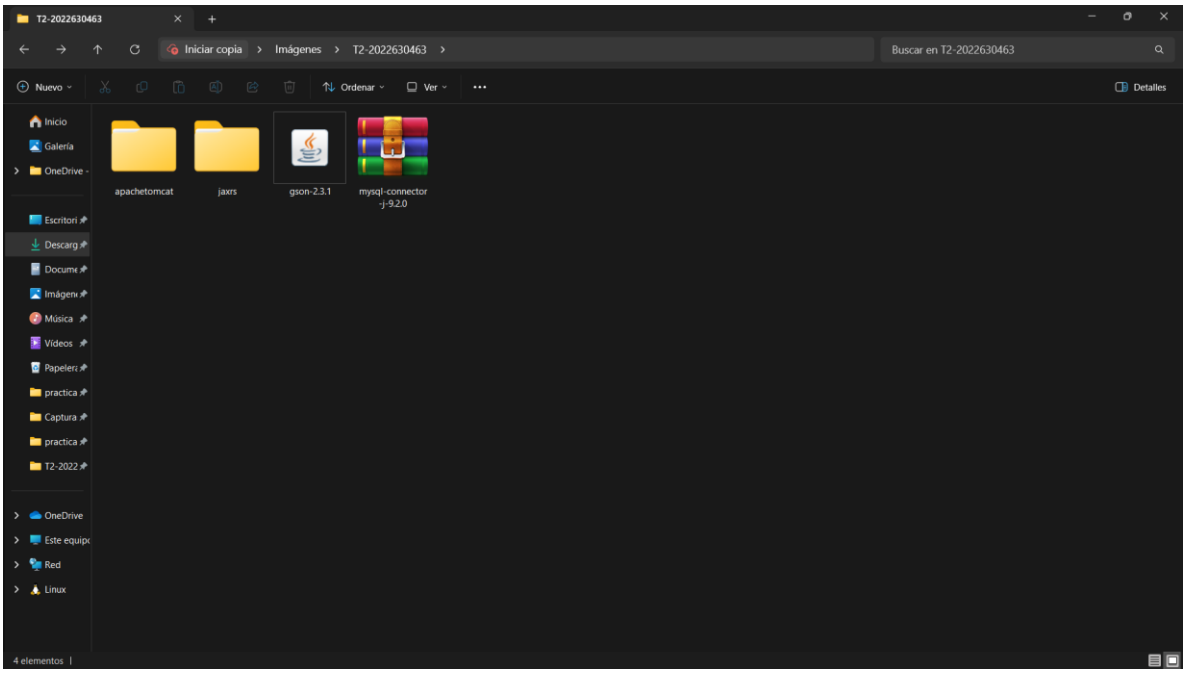
10. Copiar el archivo gson-2.3.1.jar al directorio "lib" de Tomcat.



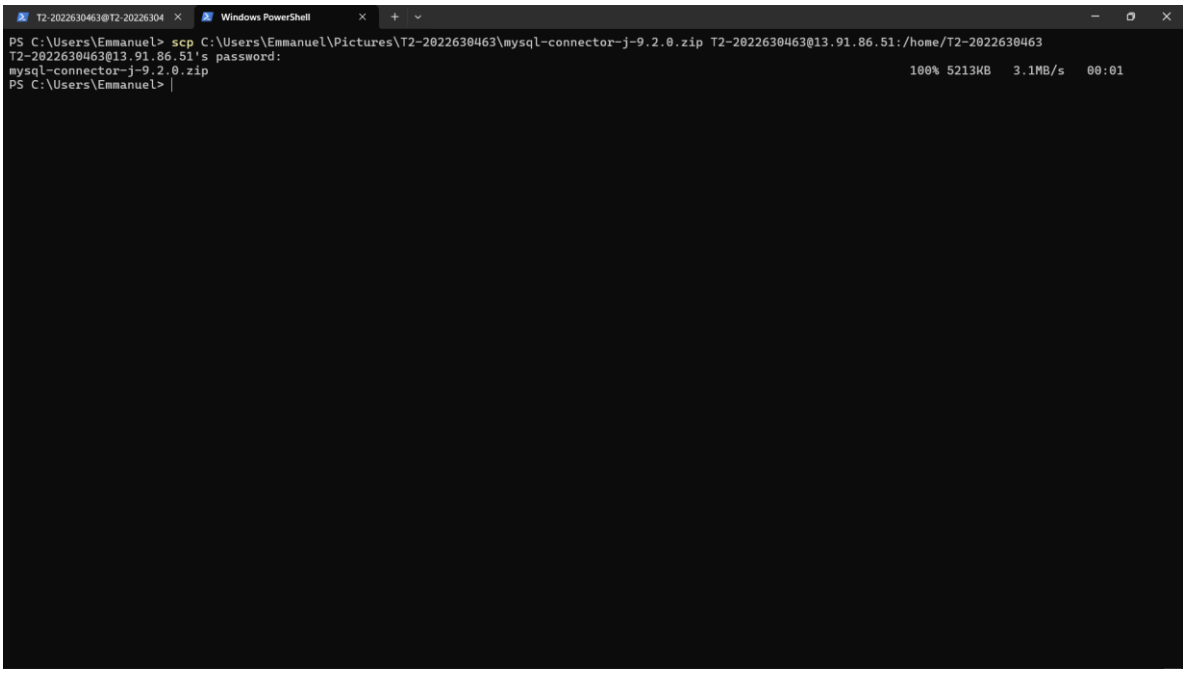
11. Ahora vamos a instalar el driver de JDBC para MySQL. Ingresar a la siguiente URL: https://dev.mysql.com/downloads/connector/j/

Seleccionar “Platform independent" y descargar el archivo ZIP.

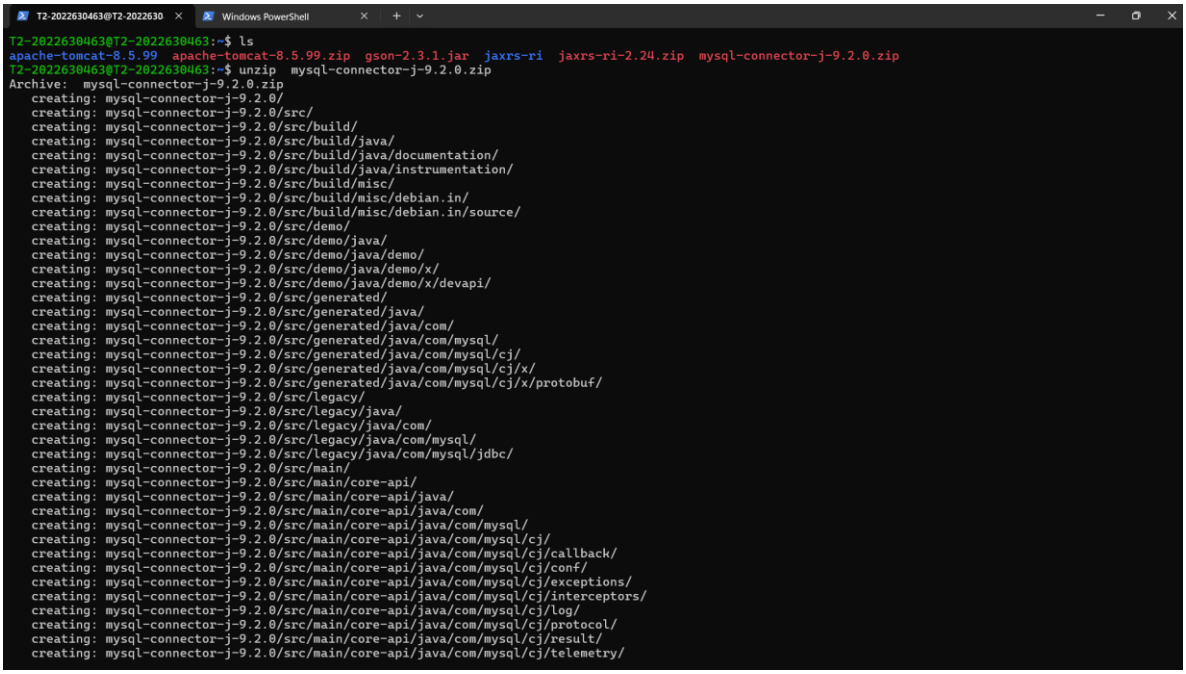


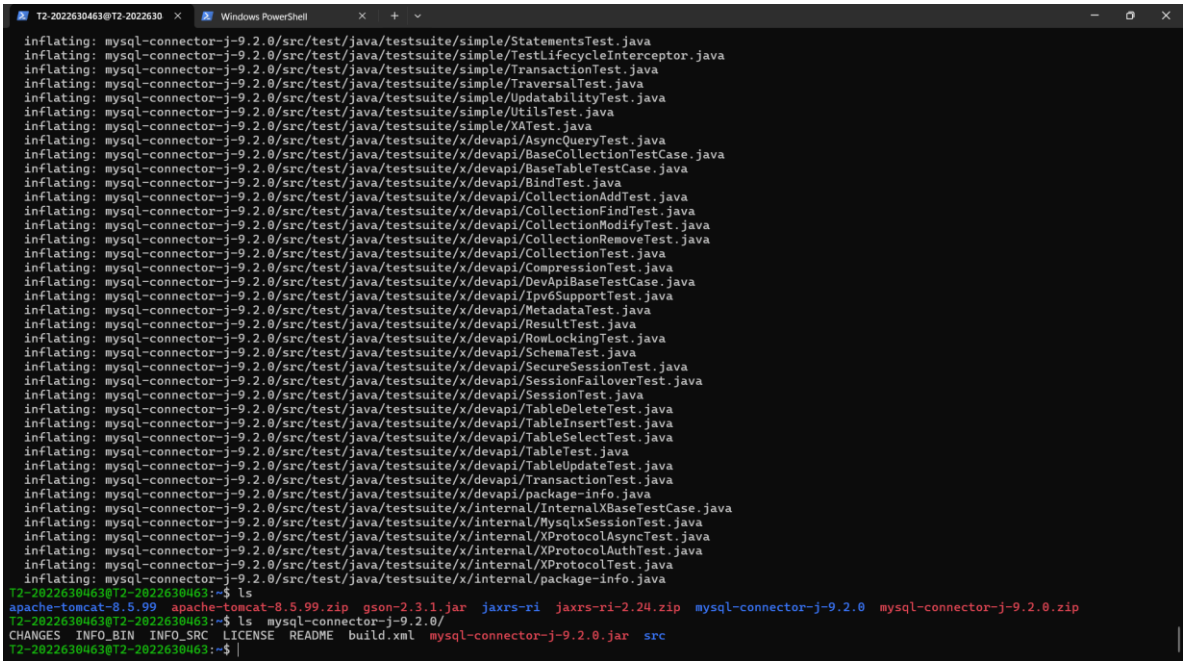


12. Copiar el archivo descargado a la máquina virtual, desempacarlo y copiar el archivo mysql-connector...jar al directorio "lib" de Tomcat.

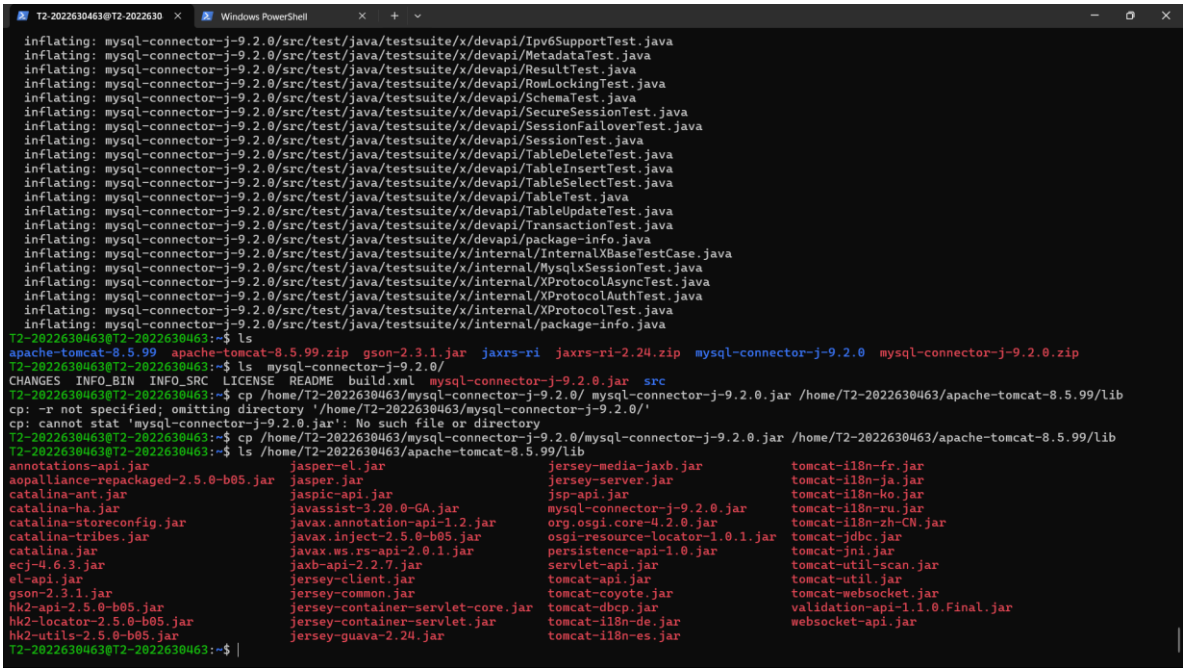


Descomprimimos el zip.





Copiar el archivo .jar al directorio lib de tomcat.



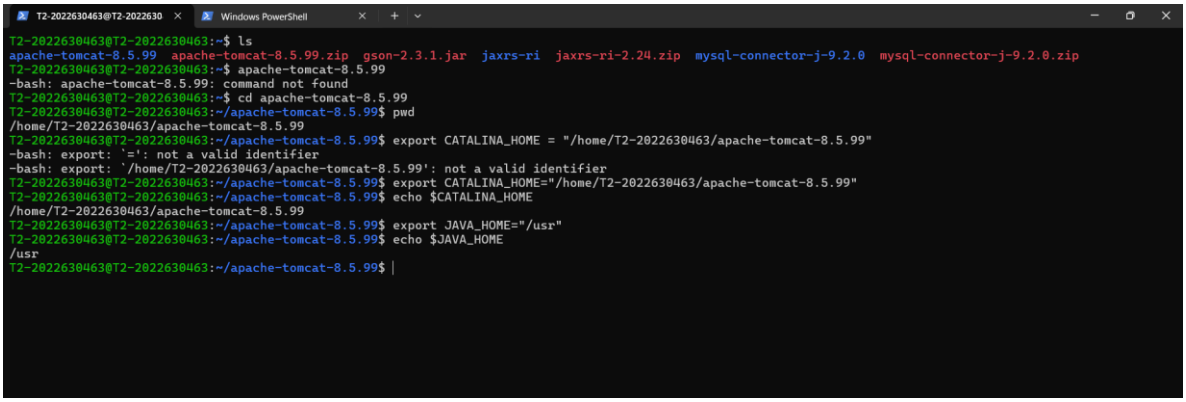
Iniciar/detener el servidor tomcat

1. Para iniciar el servidor Tomcat es **necesario** definir las siguientes variables de entorno:

export CATALINA_HOME= "aquí va la ruta absoluta del directorio de Tomcat 8"

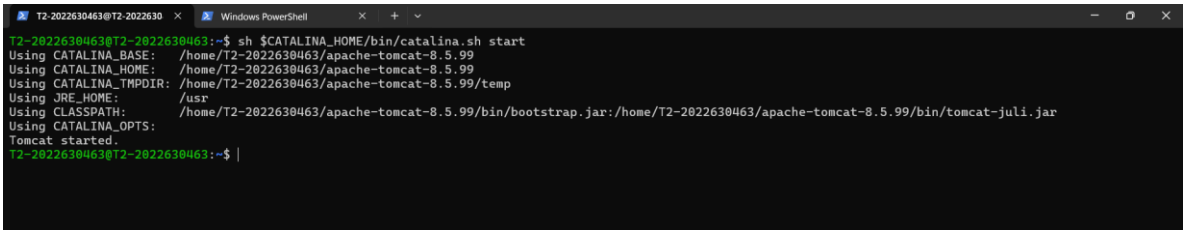
export JAVA_HOME= "aquí va la ruta absoluta del directorio donde está el directorio bin que contiene el programa java"

Nota. Si se instaló openjdk-8-jdk-headless entonces JAVA_HOME=/usr



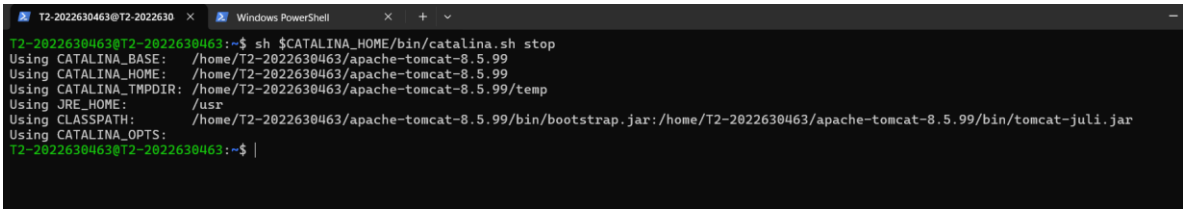
2. Iniciar la ejecución de Tomcat:

sh $CATALINA_HOME/bin/catalina.sh start



3. Para detener Tomcat se deberá ejecutar el siguiente comando:

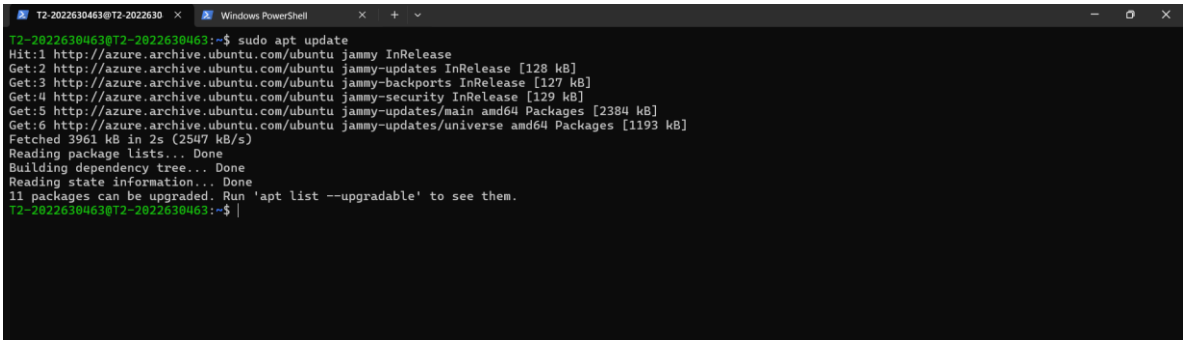
sh $CATALINA_HOME/bin/catalina.sh stop



Instalación de MYSQL

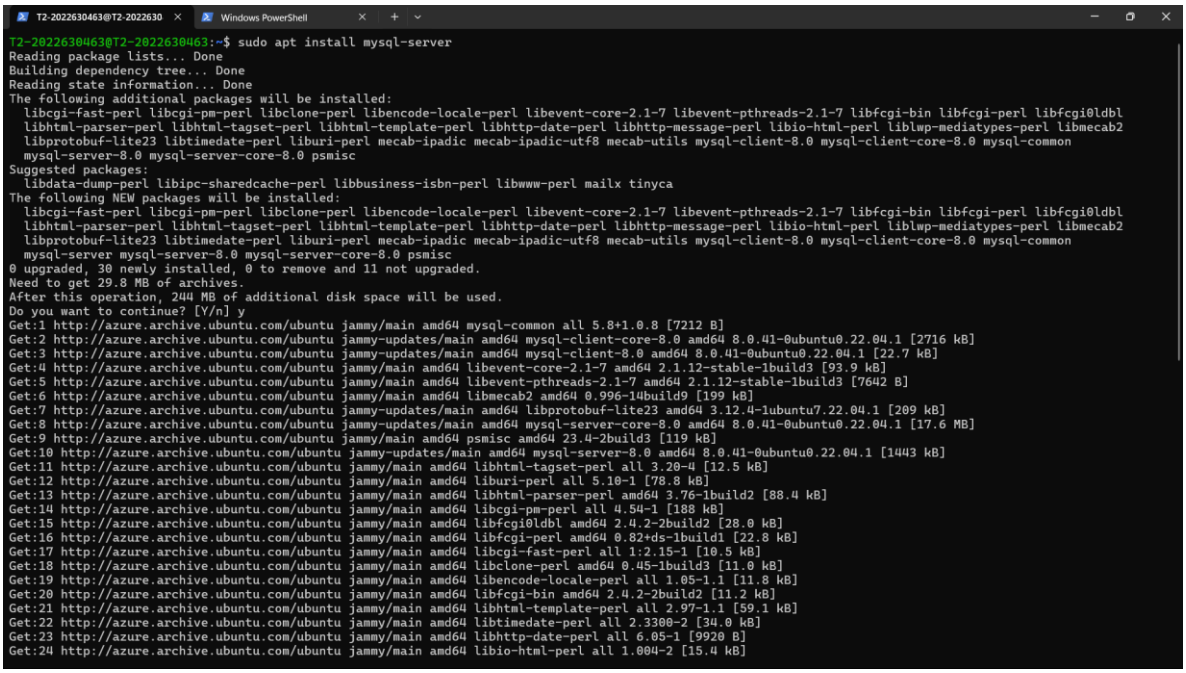
1. Actualizar los paquetes en la máquina virtual ejecutando el siguiente comando:

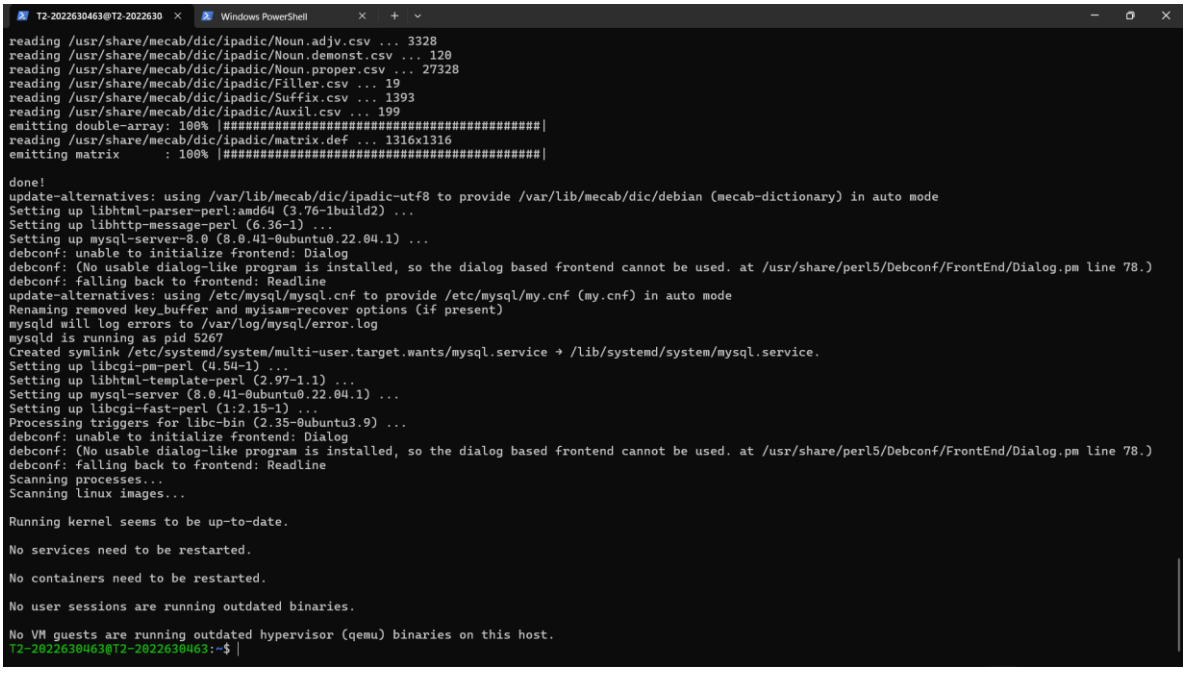
sudo apt update



2. Instalar el paquete default de MySQL:

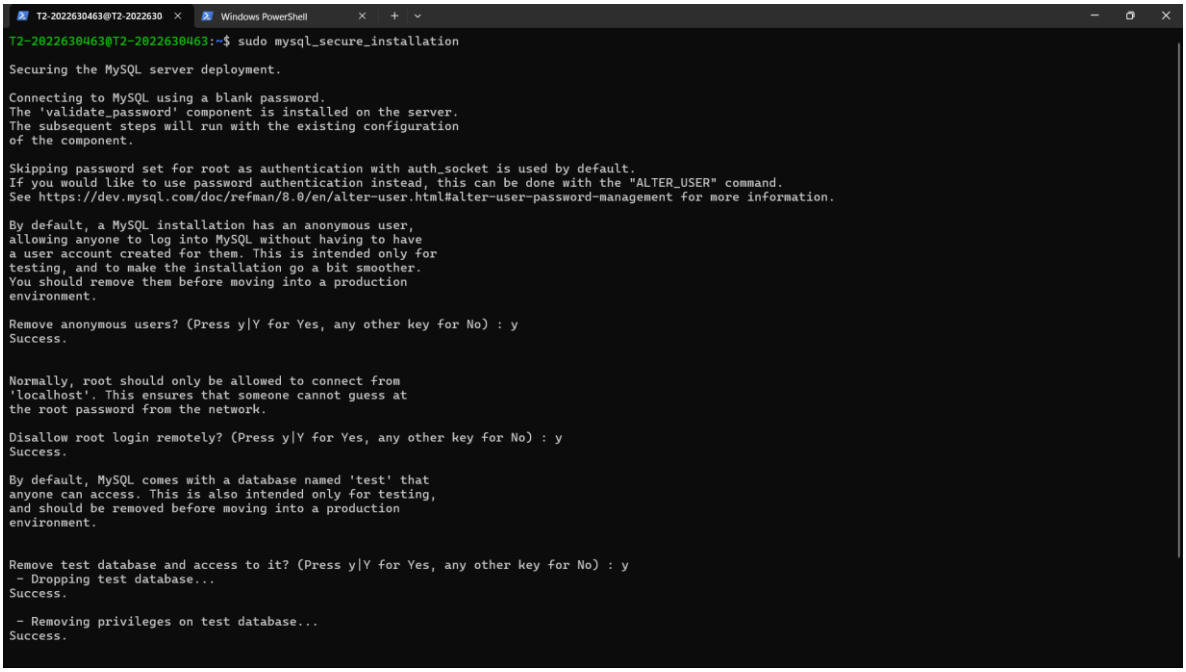
sudo apt install mysql-server

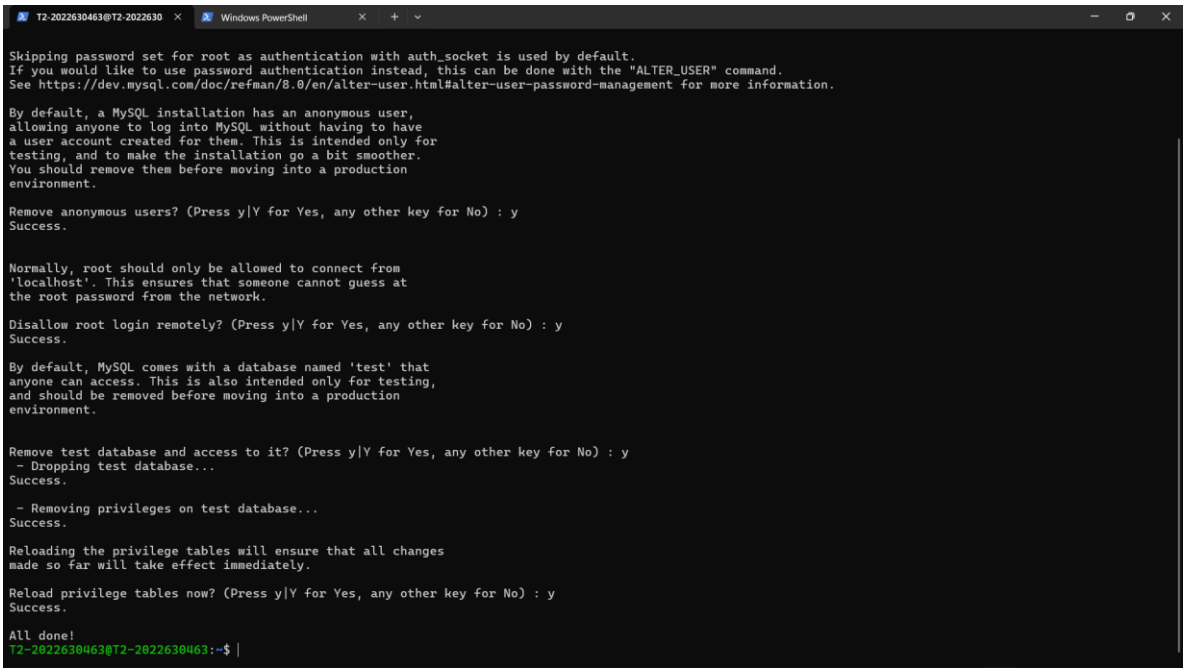

T2-2022630463@T2-2022630
Windows PowerShell
T2-2022630463@T2-2022630463:~$ sudo apt install mysql-server
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
libcgi-fast-perl libcgi-pm-perl libclone-perl libencode-locale-perl libevent-core-2.1-7 libevent-pthreads-2.1-7 libfcgi-bin libfcgi-perl libfcgi0ldbl
libhtml-parser-perl libhtml-tagset-perl libhtml-template-perl libhttp-date-perl libhttp-message-perl libio-html-perl liblwp-mediatypes-perl libmecab2
libprotobuf-lite23 libtimedate-perl liburi-perl mecab-ipadic mecab-ipadic-utf8 mecab-utils mysql-client-8.0 mysql-client-core-8.0 mysql-common
mysql-server-8.0 mysql-server-core-8.0 psmisc
Suggested packages:
libdata-dump-perl libipc-sharedcache-perl libbusiness-isbn-perl libwww-perl mailx tinyca
The following NEW packages will be installed:
libcgi-fast-perl libcgi-pm-perl libclone-perl libencode-locale-perl libevent-core-2.1-7 libevent-pthreads-2.1-7 libfcgi-bin libfcgi-perl libfcgi0ldbl
libhtml-parser-perl libhtml-tagset-perl libhtml-template-perl libhttp-date-perl libhttp-message-perl libio-html-perl liblwp-mediatypes-perl libmecab2
libprotobuf-lite23 libtimedate-perl liburi-perl mecab-ipadic mecab-ipadic-utf8 mecab-utils mysql-client-8.0 mysql-client-core-8.0 mysql-common
mysql-server mysql-server-8.0 mysql-server-core-8.0 psmisc
0 upgraded, 30 newly installed, 0 to remove and 11 not upgraded.
Need to get 29.8 MB of archives.
After this operation, 244 MB of additional disk space will be used.
Do you want to continue? [Y/n] y
Get:1 http://azure.archive.ubuntu.com/ubuntu jammy/main amd64 mysql-common all 5.8+1.0.8 [7212 B]
Get:2 http://azure.archive.ubuntu.com/ubuntu jammy-updates/main amd64 mysql-client-core-8.0 amd64 8.0.41-0ubuntu0.22.04.1 [2716 kB]
Get:3 http://azure.archive.ubuntu.com/ubuntu jammy-updates/main amd64 mysql-client-8.0 amd64 8.0.41-0ubuntu0.22.04.1 [22.7 kB]
Get:4 http://azure.archive.ubuntu.com/ubuntu jammy/main amd64 libevent-core-2.1-7 amd64 2.1.12-stable-1build3 [93.9 kB]
Get:5 http://azure.archive.ubuntu.com/ubuntu jammy/main amd64 libevent-pthreads-2.1-7 amd64 2.1.12-stable-1build3 [7642 B]
Get:6 http://azure.archive.ubuntu.com/ubuntu jammy/main amd64 libmecab2 amd64 0.996-14build9 [199 kB]
Get:7 http://azure.archive.ubuntu.com/ubuntu jammy-updates/main amd64 libprotobuf-lite23 amd64 3.12.4-1ubuntu7.22.04.1 [209 kB]
Get:8 http://azure.archive.ubuntu.com/ubuntu jammy-updates/main amd64 mysql-server-core-8.0 amd64 8.0.41-0ubuntu0.22.04.1 [17.6 MB]
Get:9 http://azure.archive.ubuntu.com/ubuntu jammy/main amd64 psmisc amd64 23.4-2build3 [119 kB]
Get:10 http://azure.archive.ubuntu.com/ubuntu jammy-updates/main amd64 mysql-server-8.0 amd64 8.0.41-0ubuntu0.22.04.1 [1443 kB]
Get:11 http://azure.archive.ubuntu.com/ubuntu jammy/main amd64 libhtml-tagset-perl all 3.20-4 [12.5 kB]
Get:12 http://azure.archive.ubuntu.com/ubuntu jammy/main amd64 liburi-perl all 5.10-1 [78.8 kB]
Get:13 http://azure.archive.ubuntu.com/ubuntu jammy/main amd64 libhtml-parser-perl amd64 3.76-1build2 [88.4 kB]
Get:14 http://azure.archive.ubuntu.com/ubuntu jammy/main amd64 libcgi-pm-perl all 4.54-1 [188 kB]
Get:15 http://azure.archive.ubuntu.com/ubuntu jammy/main amd64 libfcgi0ldbl amd64 2.4.2-2build2 [28.0 kB]
Get:16 http://azure.archive.ubuntu.com/ubuntu jammy/main amd64 libfcgi-perl amd64 0.82+ds-1build1 [22.8 kB]
Get:17 http://azure.archive.ubuntu.com/ubuntu jammy/main amd64 libcgi-fast-perl all 1:2.15-1 [10.5 kB]
Get:18 http://azure.archive.ubuntu.com/ubuntu jammy/main amd64 libclone-perl amd64 0.45-1build3 [11.0 kB]
Get:19 http://azure.archive.ubuntu.com/ubuntu jammy/main amd64 libencode-locale-perl all 1.05-1.1 [11.8 kB]
Get:20 http://azure.archive.ubuntu.com/ubuntu jammy/main amd64 libfcgi-bin amd64 2.4.2-2build2 [11.2 kB]
Get:21 http://azure.archive.ubuntu.com/ubuntu jammy/main amd64 libhtml-template-perl all 2.97-1.1 [59.1 kB]
Get:22 http://azure.archive.ubuntu.com/ubuntu jammy/main amd64 libtimedate-perl all 2.3300-2 [34.0 kB]
Get:23 http://azure.archive.ubuntu.com/ubuntu jammy/main amd64 libhttp-date-perl all 6.05-1 [9920 B]
Get:24 http://azure.archive.ubuntu.com/ubuntu jammy/main amd64 libio-html-perl all 1.004-2 [15.4 kB]


T2-2022630463@T2-2022630
Windows PowerShell
reading /usr/share/mecab/dic/ipadic/Noun.adjv.csv ... 3328
reading /usr/share/mecab/dic/ipadic/Noun.demonst.csv ... 120
reading /usr/share/mecab/dic/ipadic/Noun.proper.csv ... 27328
reading /usr/share/mecab/dic/ipadic/Filler.csv ... 19
reading /usr/share/mecab/dic/ipadic/Suffix.csv ... 1393
reading /usr/share/mecab/dic/ipadic/Auxil.csv ... 199
emitting double-array: 100% |###|
reading /usr/share/mecab/dic/ipadic/matrix.def ... 1316x1316
emitting matrix   : 100% |###|
done!
update-alternatives: using /var/lib/mecab/dic/ipadic-utf8 to provide /var/lib/mecab/dic/debian (mecab-dictionary) in auto mode
Setting up libhtml-parser-perl:amd64 (3.76-1build2) ...
Setting up libhttp-message-perl (6.36-1) ...
Setting up mysql-server-8.0 (8.0.41-0ubuntu0.22.04.1) ...
debconf: unable to initialize frontend: Dialog
debconf: (No usable dialog-like program is installed, so the dialog based frontend cannot be used. at /usr/share/perl5/Debconf/FrontEnd/Dialog.pm line 78.)
debconf: falling back to frontend: Readline
update-alternatives: using /etc/mysql/mysql.cnf to provide /etc/mysql/my.cnf (my.cnf) in auto mode
Renaming removed key_buffer and myisam-recover options (if present)
mysqld will log errors to /var/log/mysql/error.log
mysqld is running as pid 5267
Created symlink /etc/systemd/system/multi-user.target.wants/mysql.service → /lib/systemd/system/mysql.service.
Setting up libcgi-pm-perl (4.54-1) ...
Setting up libhtml-template-perl (2.97-1.1) ...
Setting up mysql-server (8.0.41-0ubuntu0.22.04.1) ...
Setting up libcgi-fast-perl (1:2.15-1) ...
Processing triggers for libc-bin (2.35-0ubuntu3.9) ...
debconf: unable to initialize frontend: Dialog
debconf: (No usable dialog-like program is installed, so the dialog based frontend cannot be used. at /usr/share/perl5/Debconf/FrontEnd/Dialog.pm line 78.)
debconf: falling back to frontend: Readline
Scanning processes...
Scanning linux images...
Running kernel seems to be up-to-date.
No services need to be restarted.
No containers need to be restarted.
No user sessions are running outdated binaries.
No VM guests are running outdated hypervisor (qemu) binaries on this host.
T2-2022630463@T2-2022630463:~$

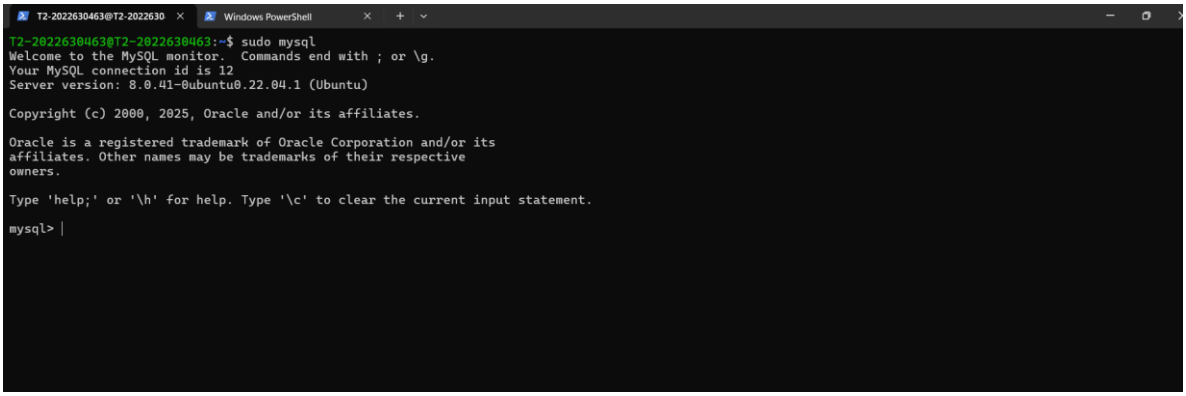
3. Ejecutar el script de seguridad:

sudo mysql_secure_installation





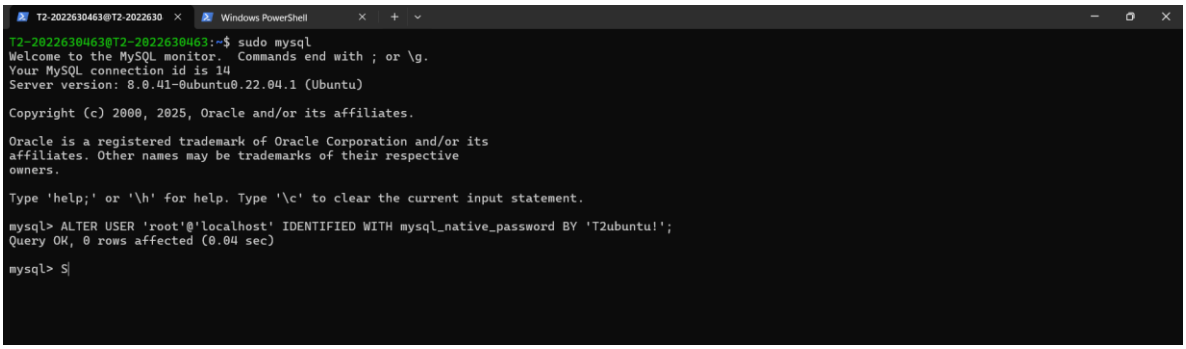
4. Ejecutar el monitor de MySQL:



5. Ejecutar el siguiente comando SQL para modificar la contraseña de root:

ALTER USER 'root'@'localhost' IDENTIFIED WITH mysql_native_password BY 'aquí-va-la-contraseña-de-root-de-mysql'

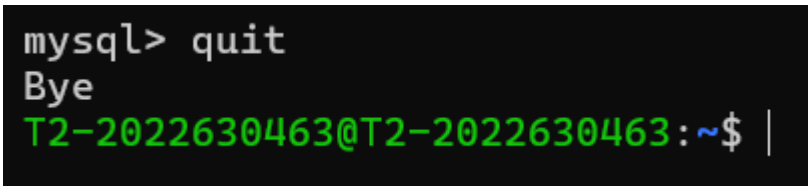
La contraseña debe de combinar mayúsculas, minúsculas y caracteres especiales con un mínimo de 8 caracteres.



6. Actualizar los privilegios:

7. Ejecutar el siguiente comando para salir del monitor de MySQL:

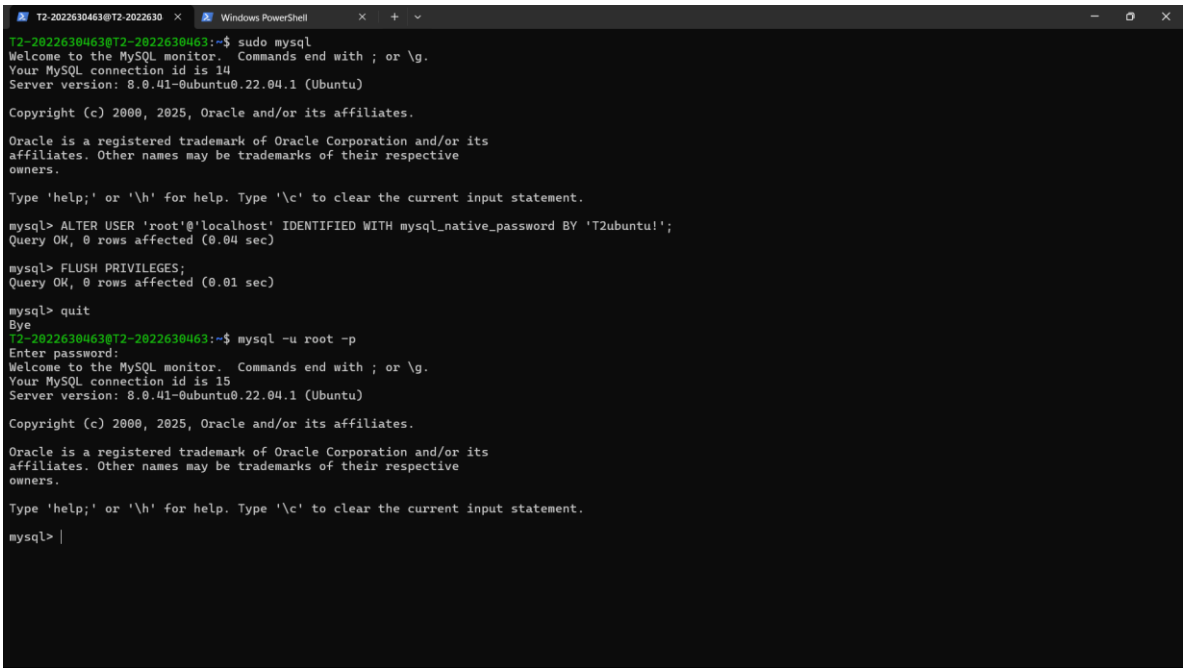
quit



Crear un usuario en MYSQL

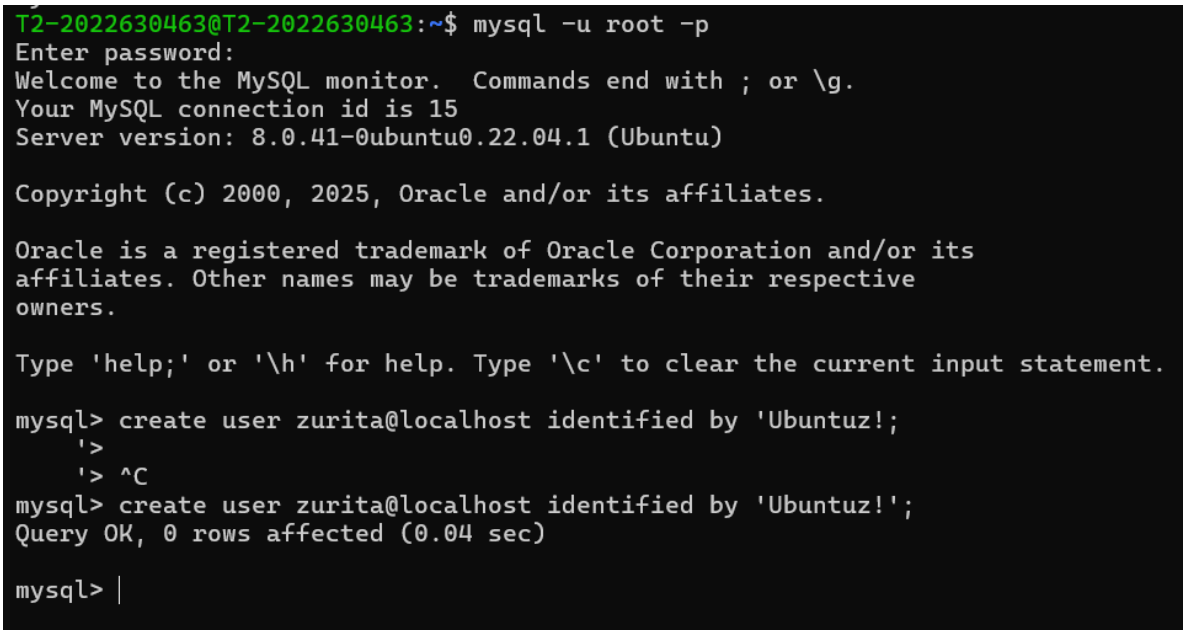
1. Ejecutar el monitor de MySQL:

mysql -u root -p



2. Crear un usuario con el comando:

create user usuario@localhost identified by 'aquí-va-la-contraseña-del-usuario-hugo';



3. Otorgar todos los permisos al usuario nuevo sobre la base de datos "servicio_web":

grant all on servicio_web.* to usuario@localhost;

```
mysql> grant all on servicio_web.* to zurita@localhost;
Query OK, 0 rows affected (0.02 sec)
```

4. Ejecutar el siguiente comando para salir del monitor de MySQL:

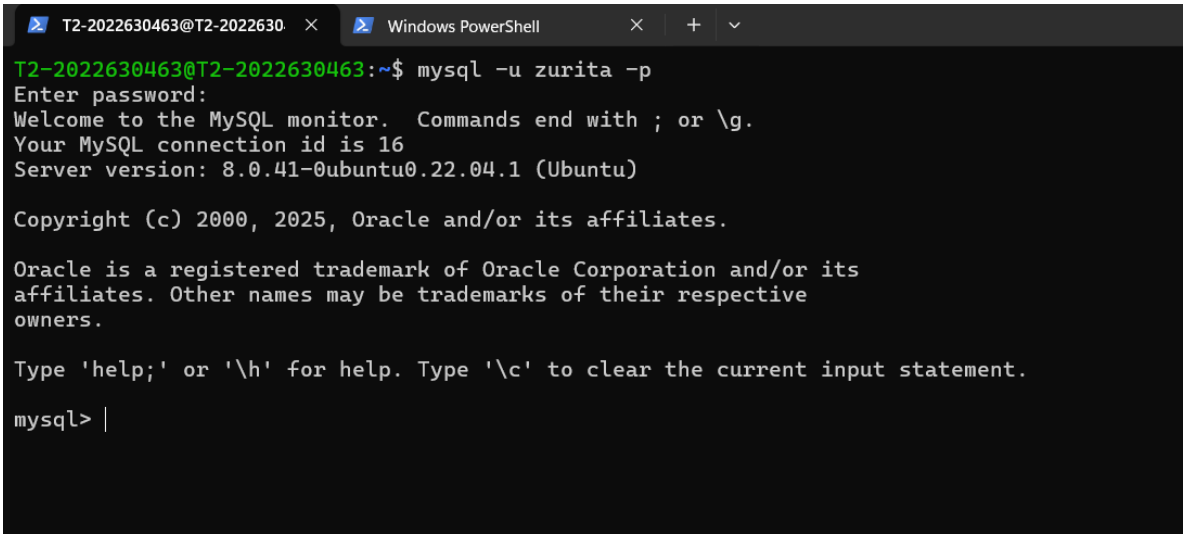
quit

```
T2-2022630463@T2-2022630463:~$ mysql -u root -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 15
Server version: 8.0.41-0ubuntu0.22.04.1 (Ubuntu)

Copyright (c) 2000, 2025, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> create user zurita@localhost identified by 'Ubuntuz!;
    '>
    '> ^C
mysql> create user zurita@localhost identified by 'Ubuntuz!';
Query OK, 0 rows affected (0.04 sec)

mysql> grant all on servicio_web.* to zurita@localhost;
Query OK, 0 rows affected (0.02 sec)

mysql> quit
Bye
T2-2022630463@T2-2022630463:~$
```

Crear la base de datos

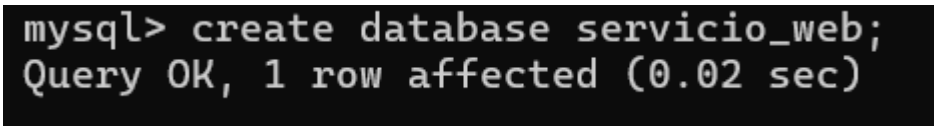
1. Ejecutar el monitor de MySQL (notar que ahora se utiliza el usuario "zurita"):

mysql -u zurita -p

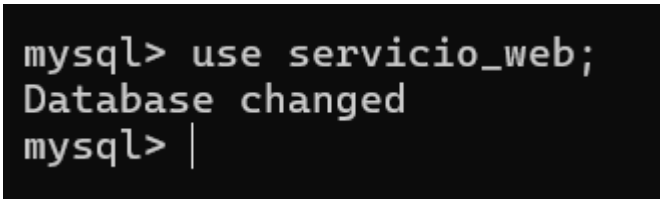


2. Crear la base de datos "servicio_web":

create database servicio_web;



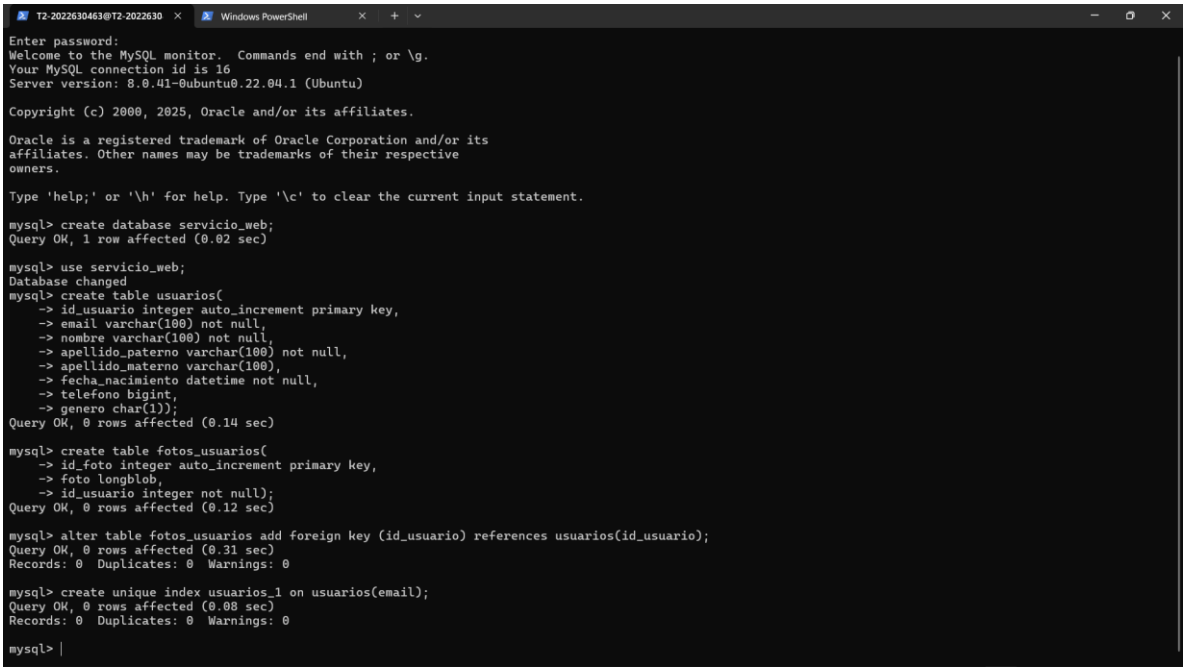
3. Conectar a la base de datos creada anteriormente:



4. Crear las tablas "usuarios" y "fotos_usuarios", así mismo, se crea una regla de integridad referencial y un índice único:

```
create table usuarios
(
  id_usuario integer auto_increment primary key,
  email varchar(100) not null,
  nombre varchar(100) not null,
  apellido_paterno varchar(100) not null,
```

```
  apellido_materno varchar(100),
  fecha_nacimiento datetime not null,
  telefono bigint,
  genero char(1)
);

create table fotos_usuarios
(
  id_foto integer auto_increment primary key,
  foto longblob,
  id_usuario integer not null
);

alter table fotos_usuarios add foreign key (id_usuario) references
usuarios(id_usuario);
create unique index usuarios_1 on usuarios(email);
```


T2-2022630463@T2-2022630
Windows PowerShell
Enter password:
Welcome to the MySQL monitor. Commands end with ; or \g.
Your MySQL connection id is 16
Server version: 8.0.41-0ubuntu0.22.04.1 (Ubuntu)
Copyright (c) 2000, 2025, Oracle and/or its affiliates.
Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.
Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.
mysql> create database servicio_web;
Query OK, 1 row affected (0.02 sec)
mysql> use servicio_web;
Database changed
mysql> create table usuarios(
    -> id_usuario integer auto_increment primary key,
    -> email varchar(100) not null,
    -> nombre varchar(100) not null,
    -> apellido_paterno varchar(100) not null,
    -> apellido_materno varchar(100),
    -> fecha_nacimiento datetime not null,
    -> telefono bigint,
    -> genero char(1));
Query OK, 0 rows affected (0.14 sec)
mysql> create table fotos_usuarios(
    -> id_foto integer auto_increment primary key,
    -> foto longblob,
    -> id_usuario integer not null);
Query OK, 0 rows affected (0.12 sec)
mysql> alter table fotos_usuarios add foreign key (id_usuario) references usuarios(id_usuario);
Query OK, 0 rows affected (0.31 sec)
Records: 0 Duplicates: 0 Warnings: 0
mysql> create unique index usuarios_1 on usuarios(email);
Query OK, 0 rows affected (0.08 sec)
Records: 0 Duplicates: 0 Warnings: 0
mysql>
```
```

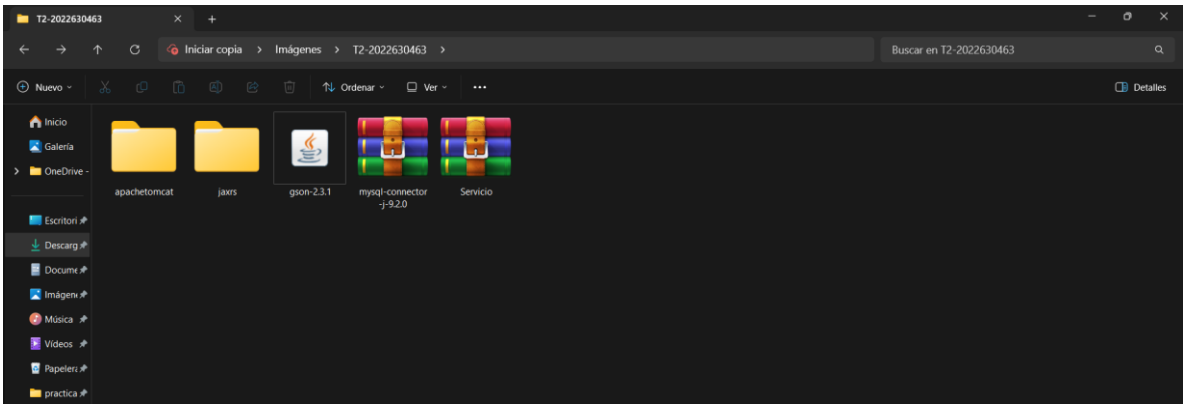
5. Salir del monitor de MySQL:

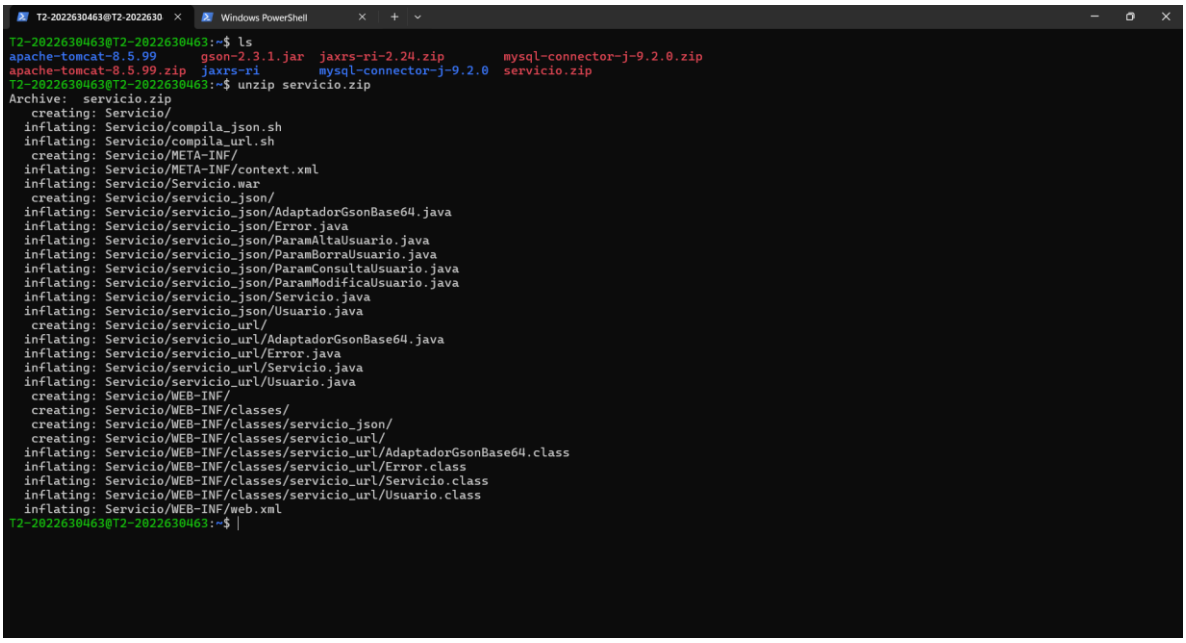
Quit

```
mysql> quit
Bye
T2-2022630463@T2-2022630463:~$
```

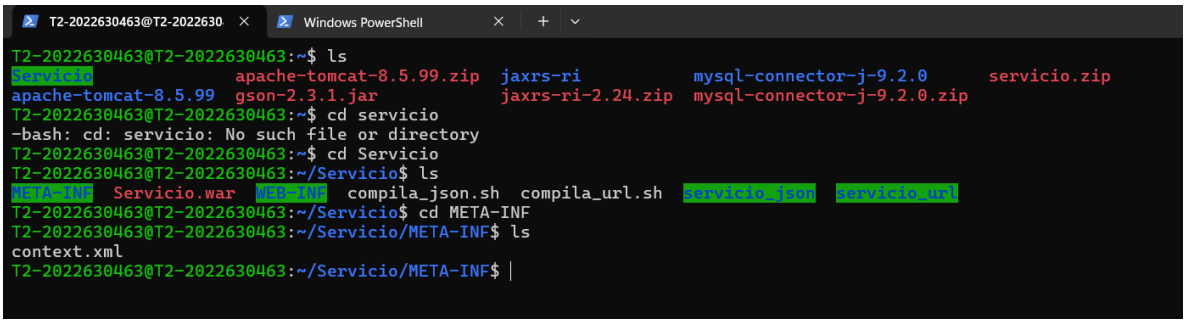
Editar el archivo "context.xml" que está en el directorio "META-INF" y definir el username de la base de datos y el password correspondiente.

Copiamos el zip a la máquina virtual.

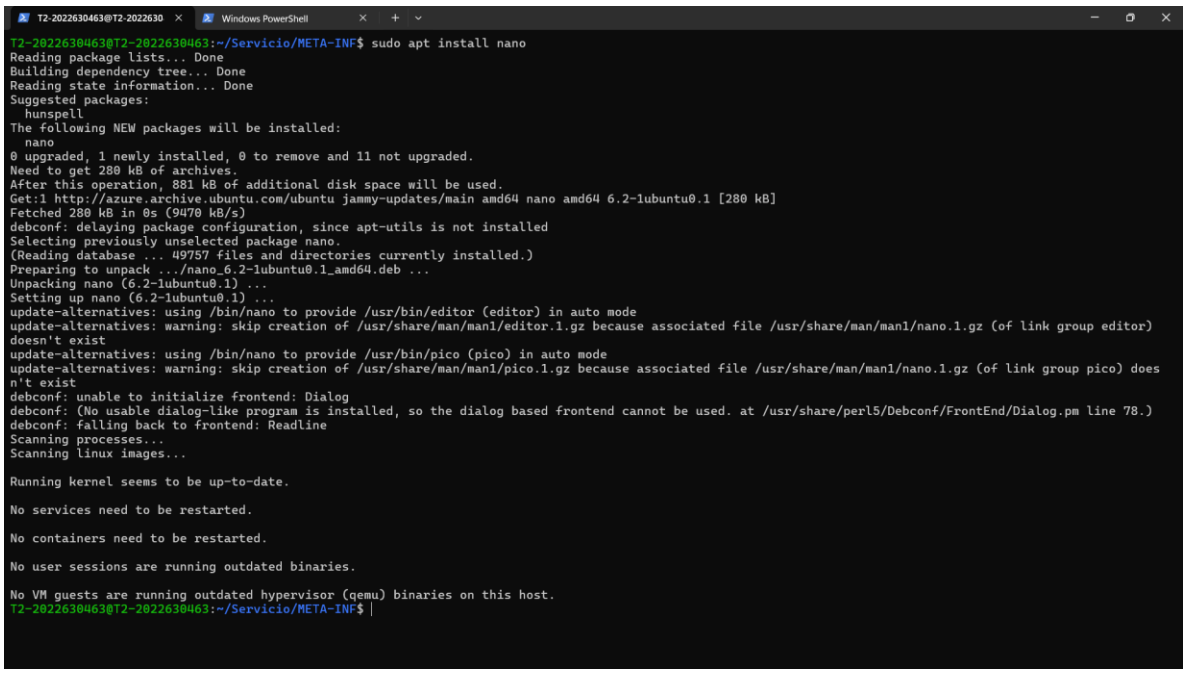




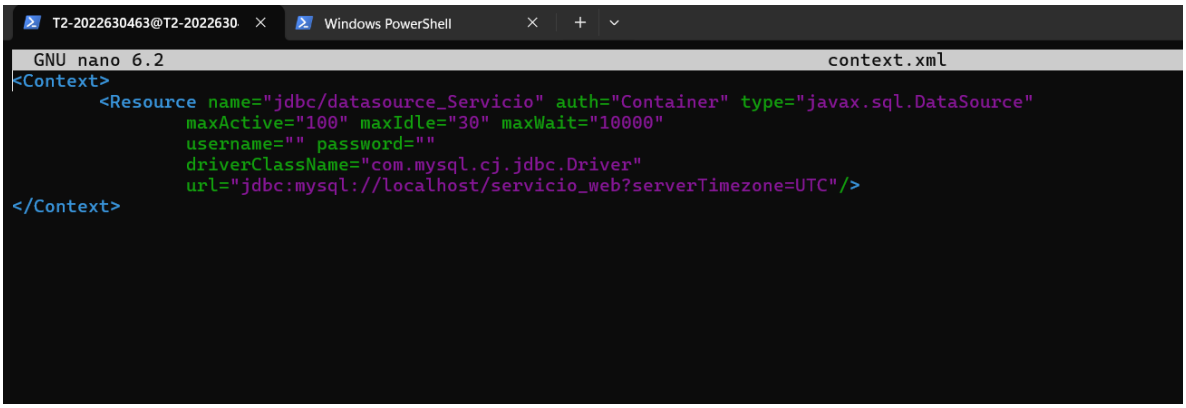
Accedemos al directorio META-INF y comprobamos que exista el archivo context.xml



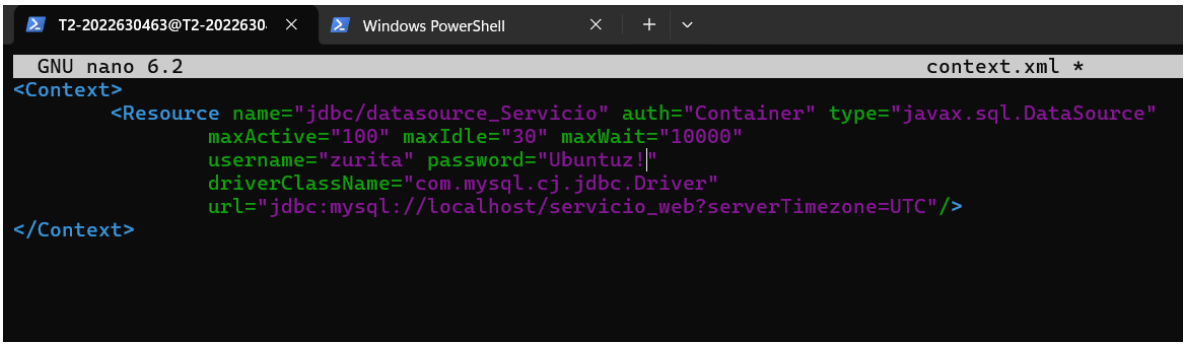
Para modificar el archivo necesitamos instalar la paquetería de nano



Con el comando **sudo nano context.xml** accedemos al contenido del archivo y aquí ya se puede modificar.



En username y password ponemos el usuario y contraseña configurados anteriormente.

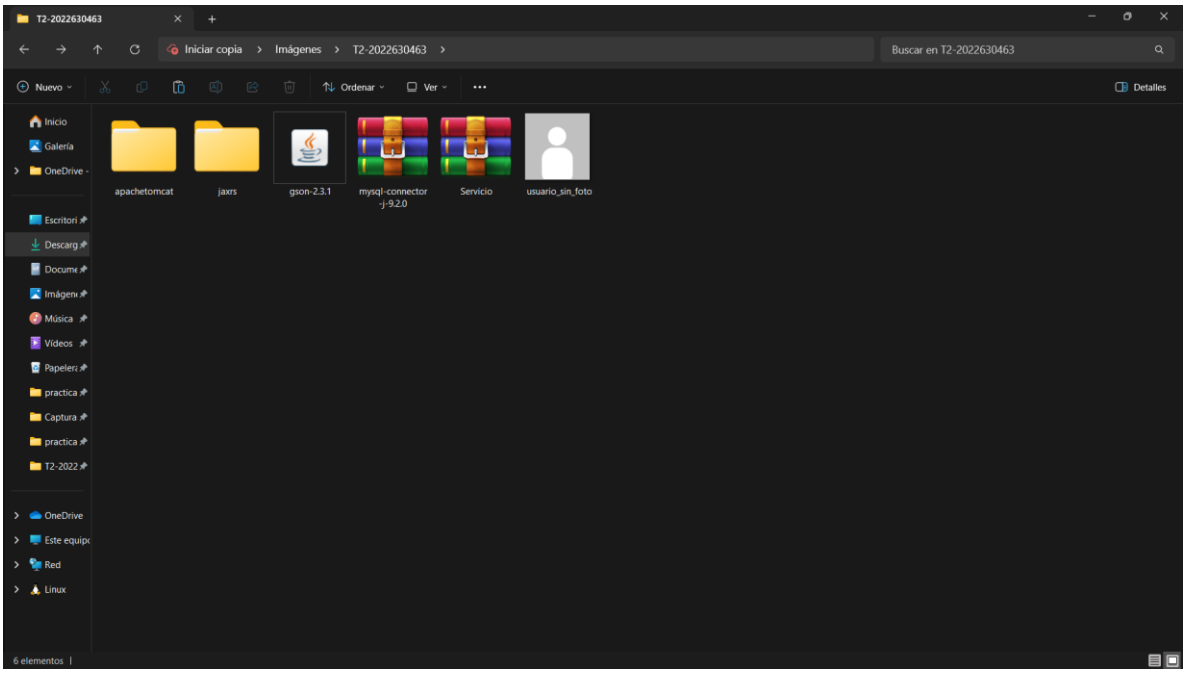


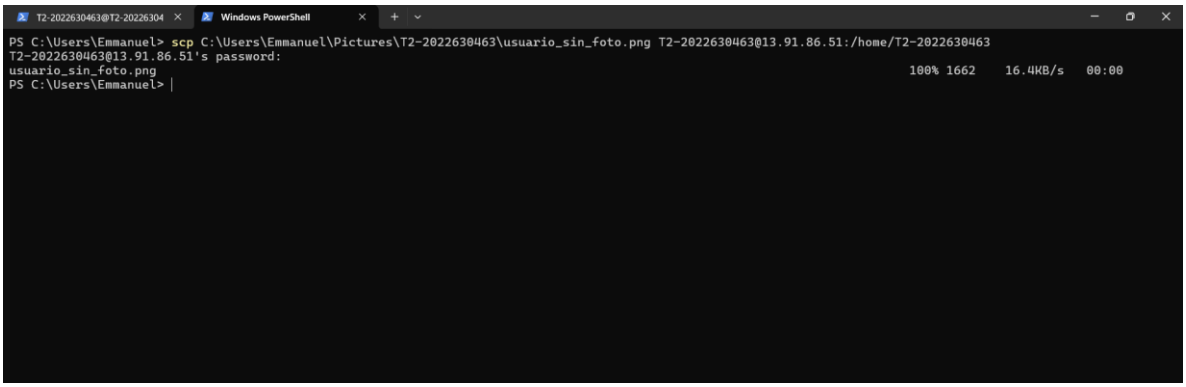
Y pulsando ctrl+ o guardamos el contenido modificado.

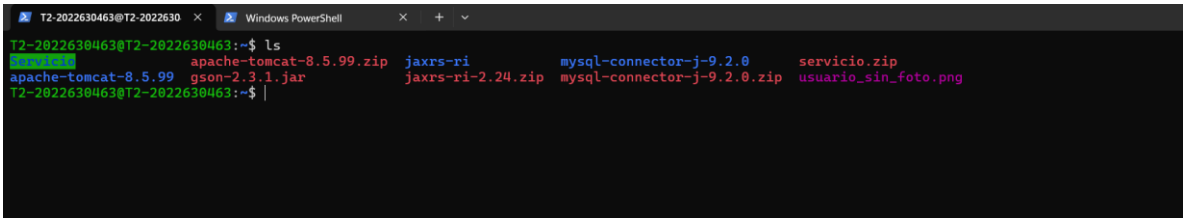
Y salimos del archivo con ctrl+x

Publicar el cliente en tomcat

1. Copiar el archivo usuario_sin_foto.png al subdirectorio webapps/ROOT de Tomcat.







Copiamos la imagen a la carpeta webapps/root de tomcat.

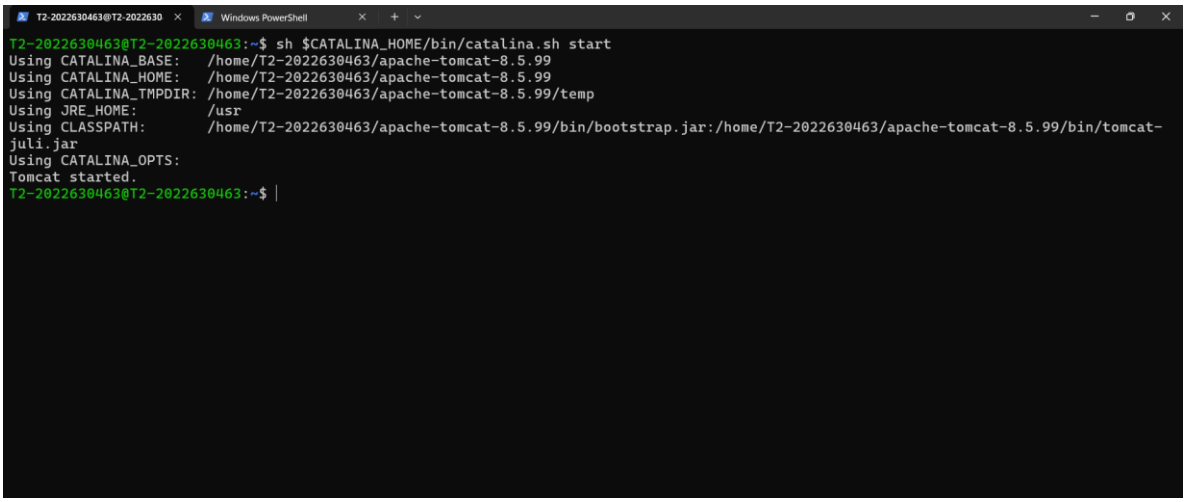
Nota. Todos los archivos que se encuentran en el directorio webapps/ROOT de Tomcat son accesibles públicamente.

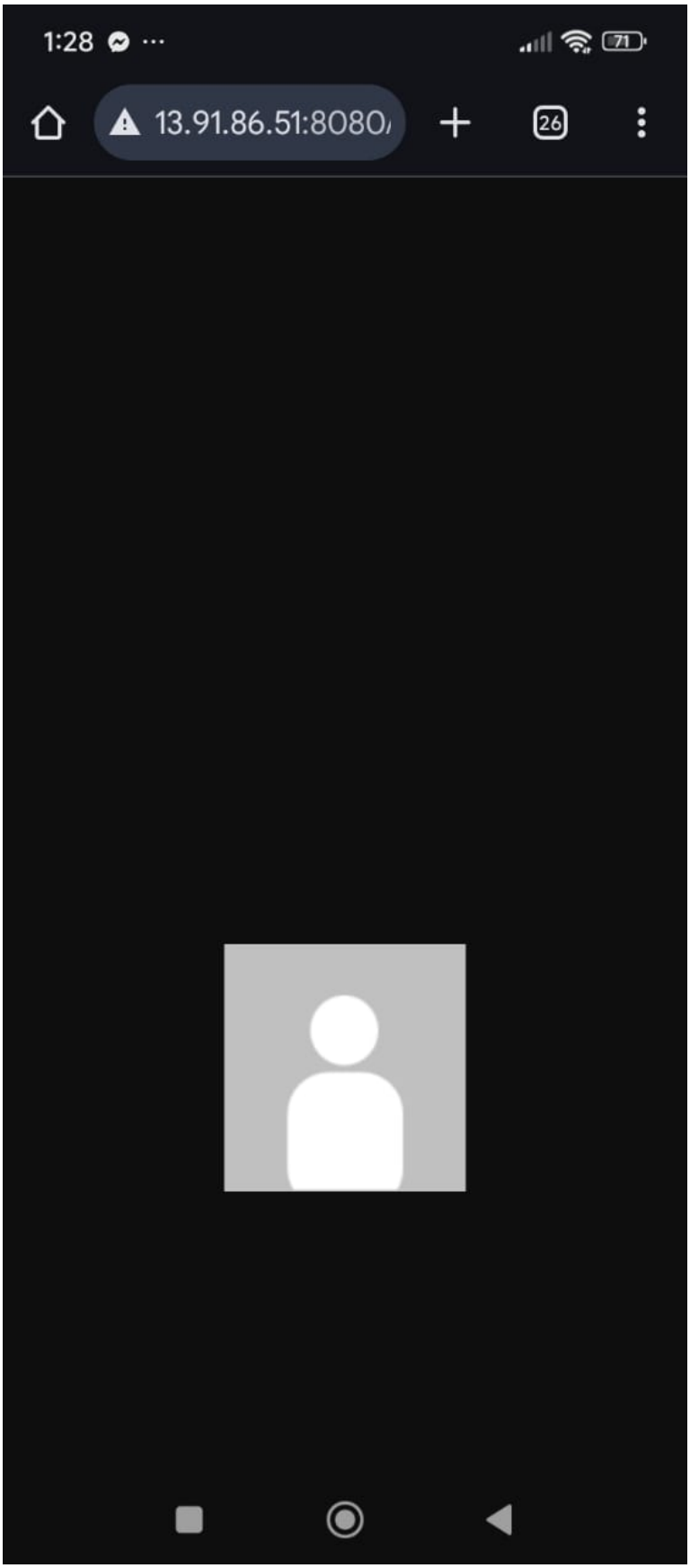
Para poder ejecutarlo correctamente agregando a la parte del comando de cp, la parte de comando sudo, debido a que no tiene permisos

Para probar que Tomcat esté en línea y el puerto 8080 esté abierto, ingresar la siguiente URL en un navegador:

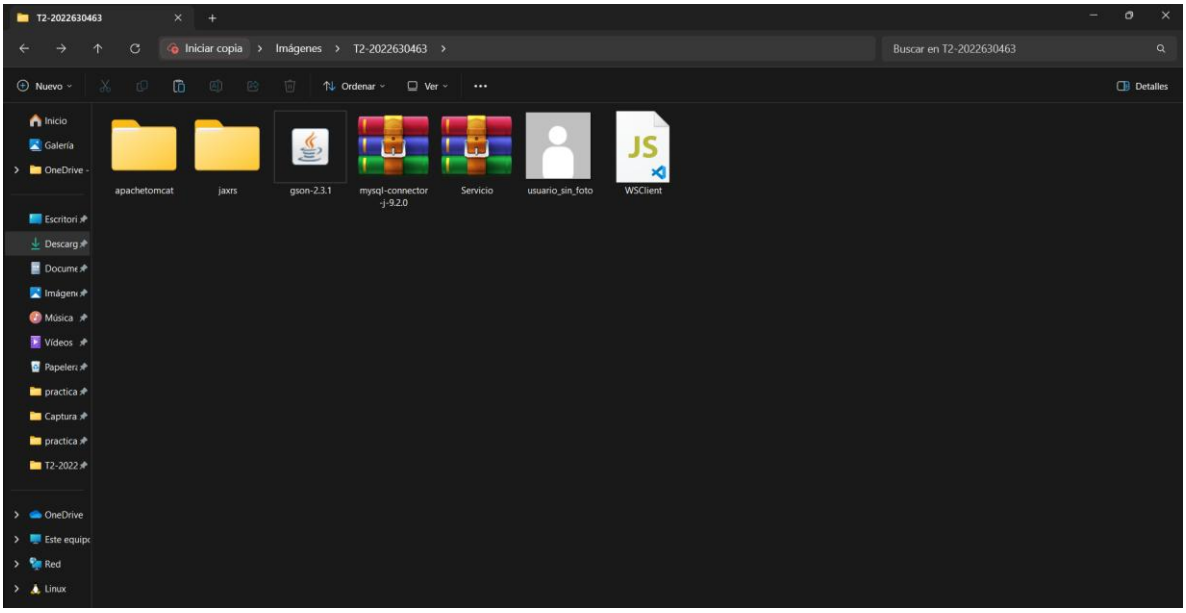
http://ip-de-la-máquina-virtual:8080/usuario_sin_foto.png

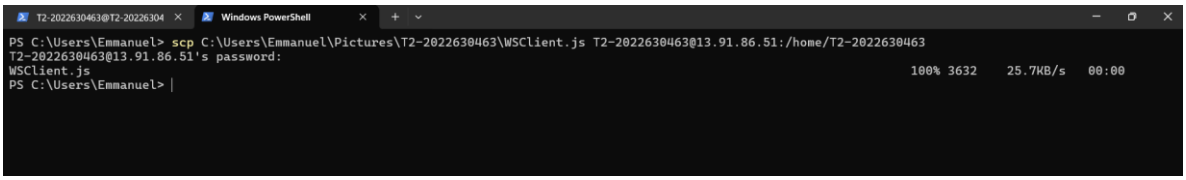
Primero debemos iniciar el servidor de tomcat.

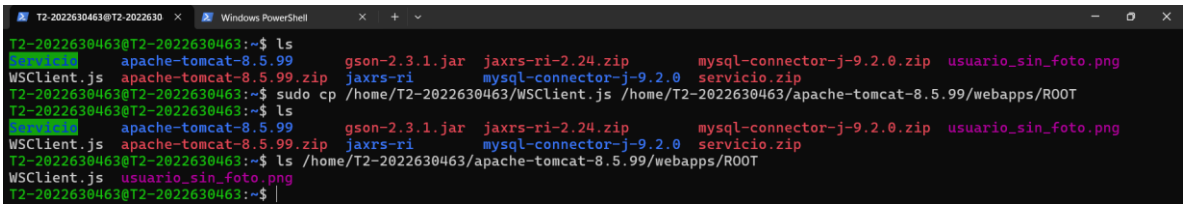



1:28
13.91.86.51:8080/
26

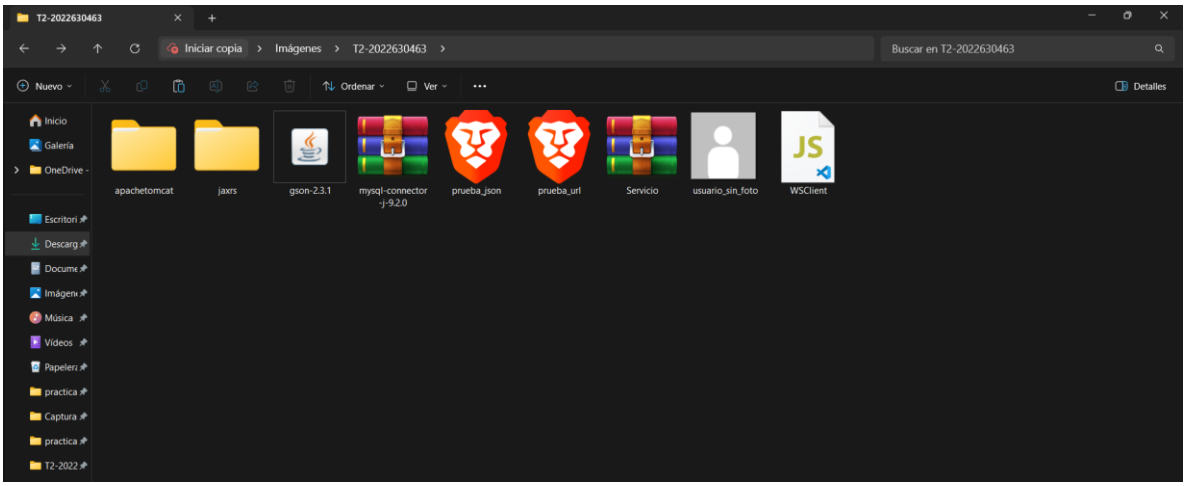
2. Copiar el archivo WSClient.js al directorio webapps/ROOT de Tomcat.

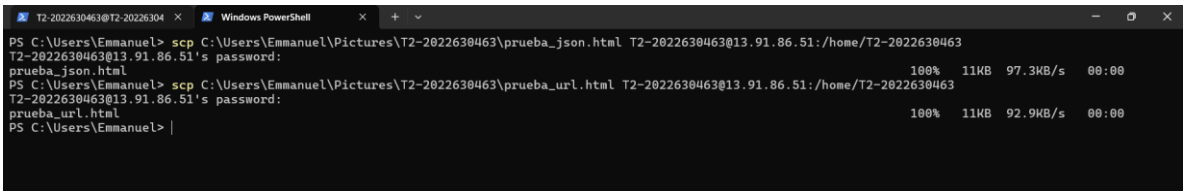


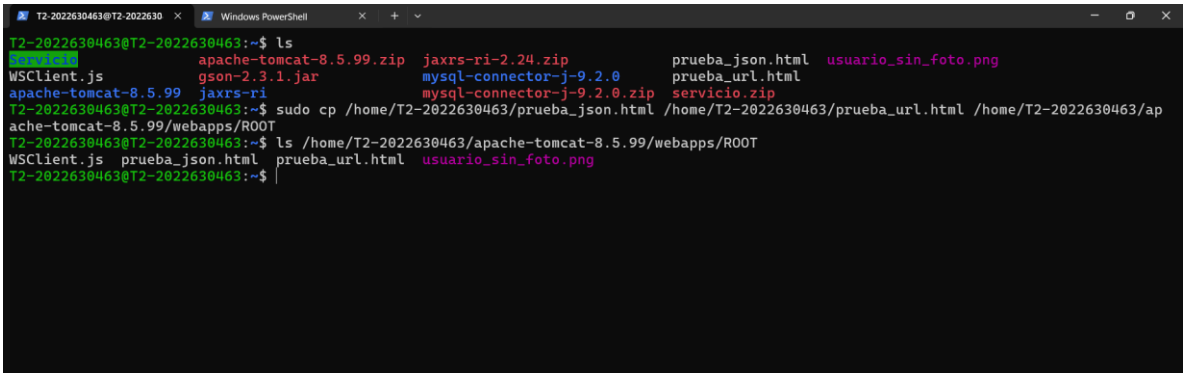




3. Copiar los archivos prueba_json.html y prueba_url.html al directorio webapps/ROOT de Tomcat.

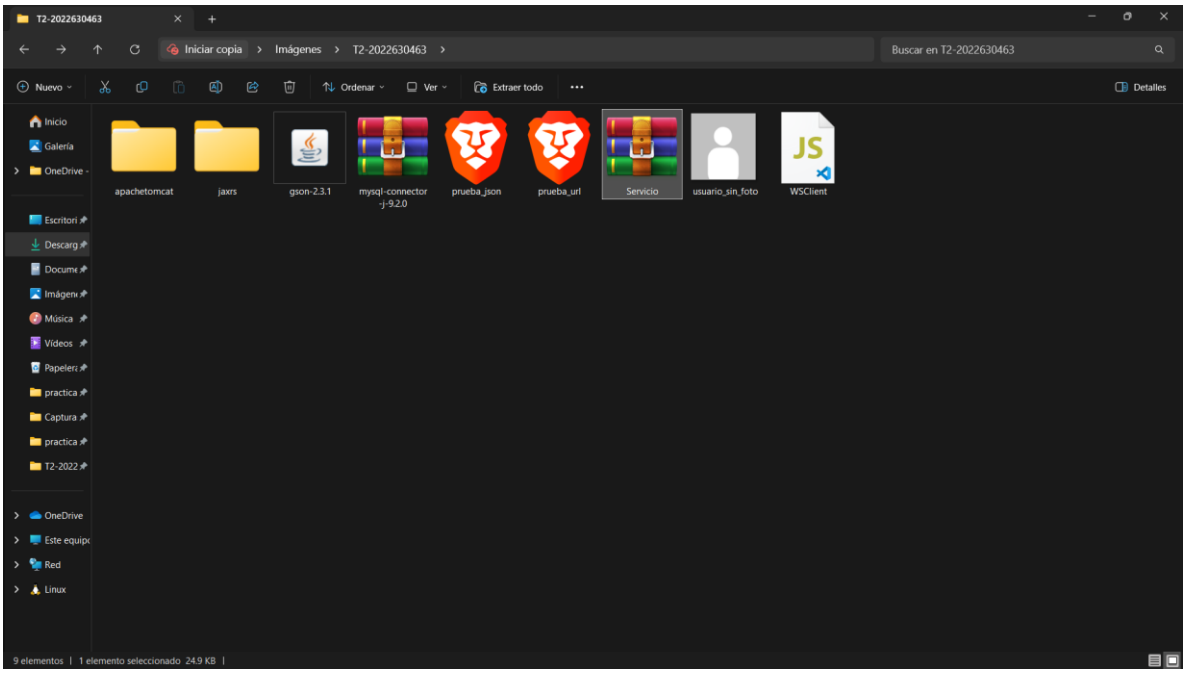


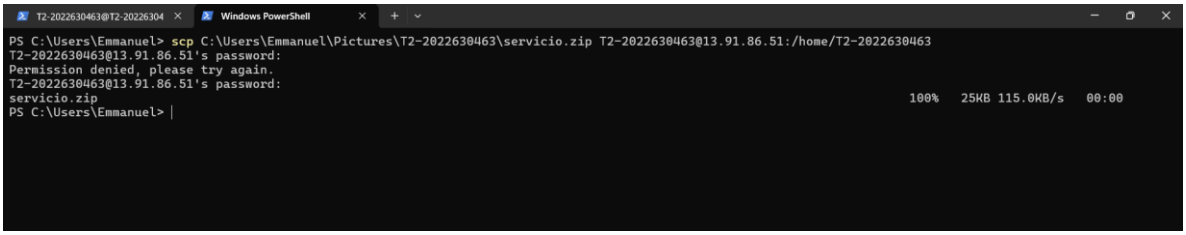




Compilar, empacar y desplegar el servicio web (versión URL)

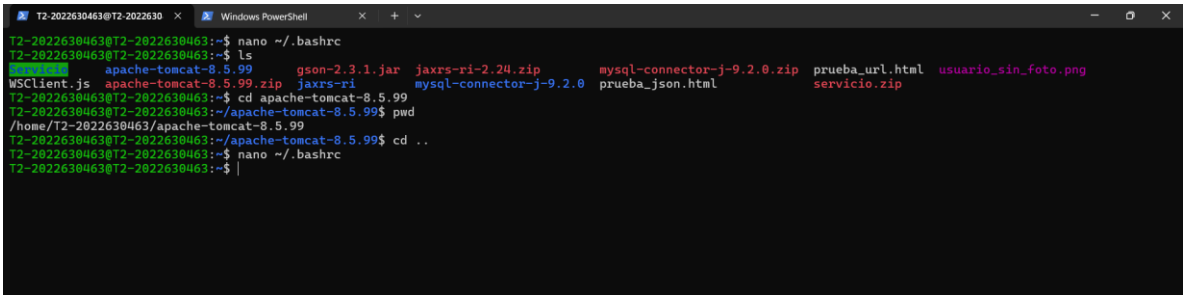
1. Descargar de la plataforma y desempacar el archivo Servicio.zip

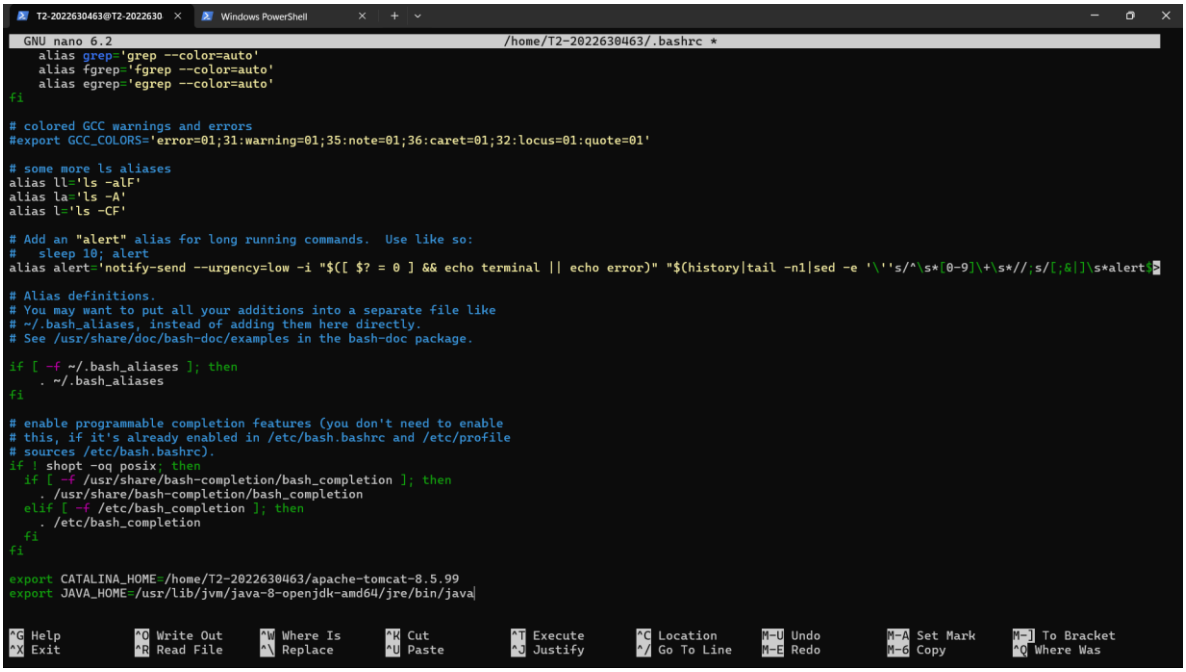


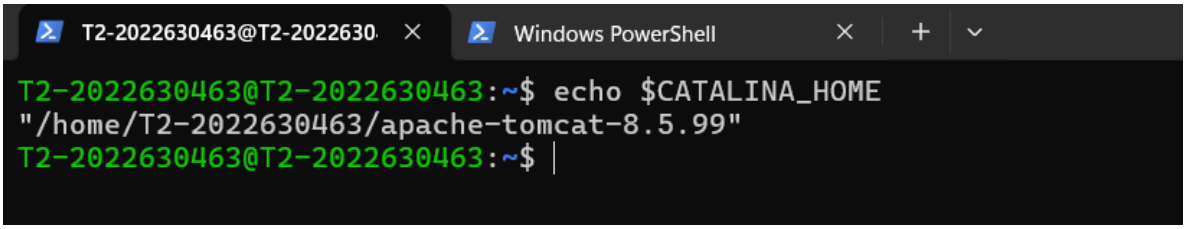


2. Definir la variable de ambiente CATALINA_HOME:

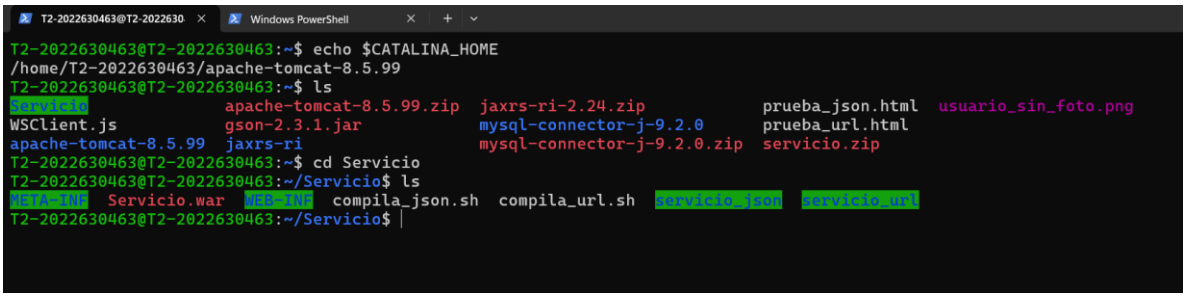
Accedemos a nano ~/.bashrc para agregar al final del archivo las nuevas variables de entorno.





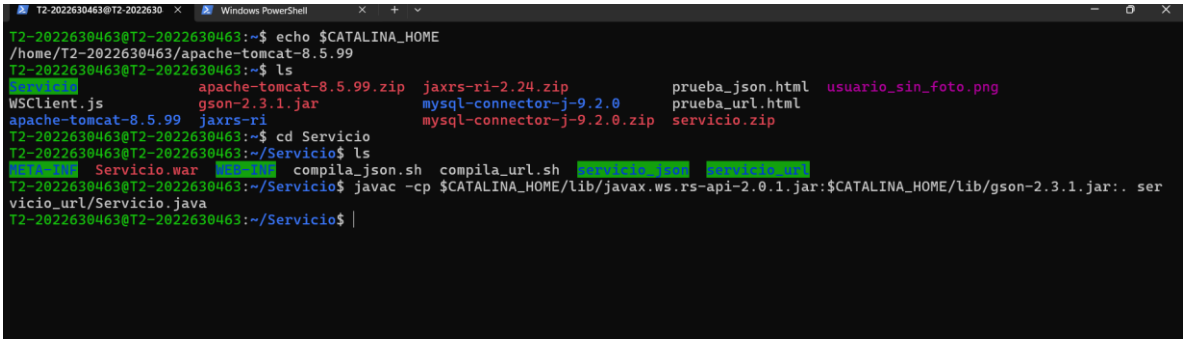


3. Cambiar al directorio donde se encuentran los directorios "servicio_url" y "servicio_json".



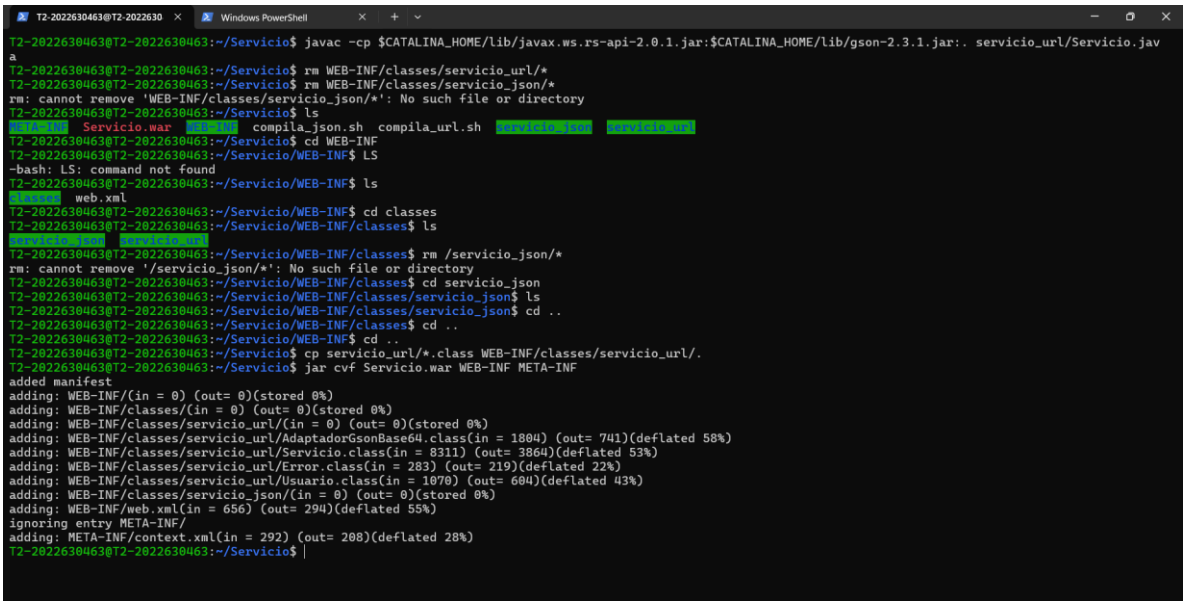
4. Compilar el servicio web:

javac -cp $CATALINA_HOME/lib/javax.ws.rs-api-2.0.1.jar:$CATALINA_HOME/lib/gson-2.3.1.jar:. servicio_url/Servicio.java



5. Crear el archivo Servicio.war:

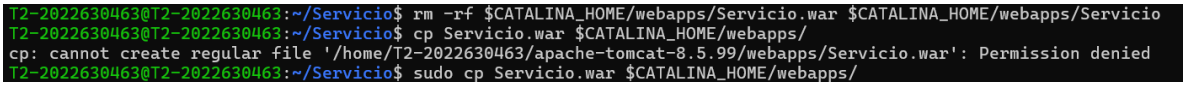
rm WEB-INF/classes/servicio_url/*
rm WEB-INF/classes/servicio_json/*
cp servicio_url/*.class WEB-INF/classes/servicio_url/.
jar cvf Servicio.war WEB-INF META-INF



6. Para remover (undeploy) y desplegar (deploy) el servicio web, se deberá eliminar el archivo Servicio.war y el directorio Servicio (en este orden), y luego copiar el archivo Servicio.war al directorio webapps de Tomcat:

rm -rf $CATALINA_HOME/webapps/Servicio.war $CATALINA_HOME/webapps/Servicio

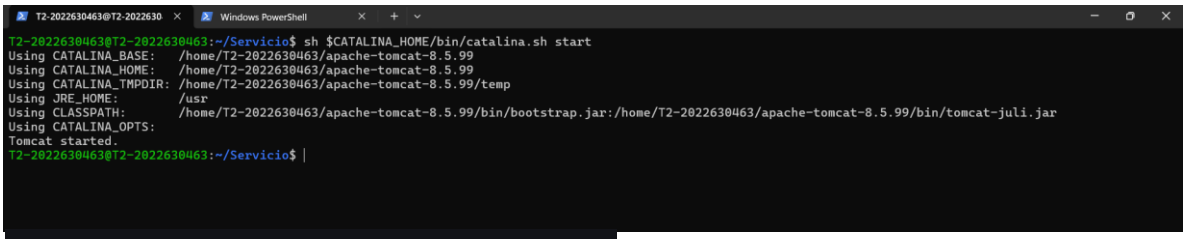
cp Servicio.war $CATALINA_HOME/webapps/.

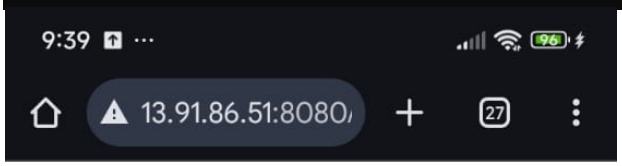


El comando cp se debe de ejecutar con sudo ya que de lo contrario te bloquea los permisos.

7. Utilizando un teléfono inteligente o una tableta, ingresar la siguiente URL en un navegador:

http://ip-de-la-máquina-virtual:8080/prueba_url.html





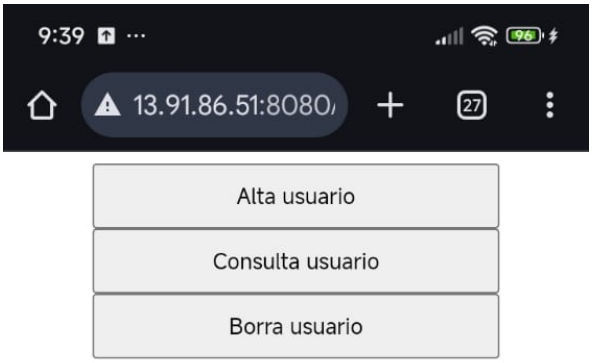
Alta usuario

Consulta usuario

Borra usuario

Probar el servicio web utilizando el cliente HTML-Javascript

1. Utilizando un teléfono inteligente o una tableta, ingresar la siguiente URL en un navegador:

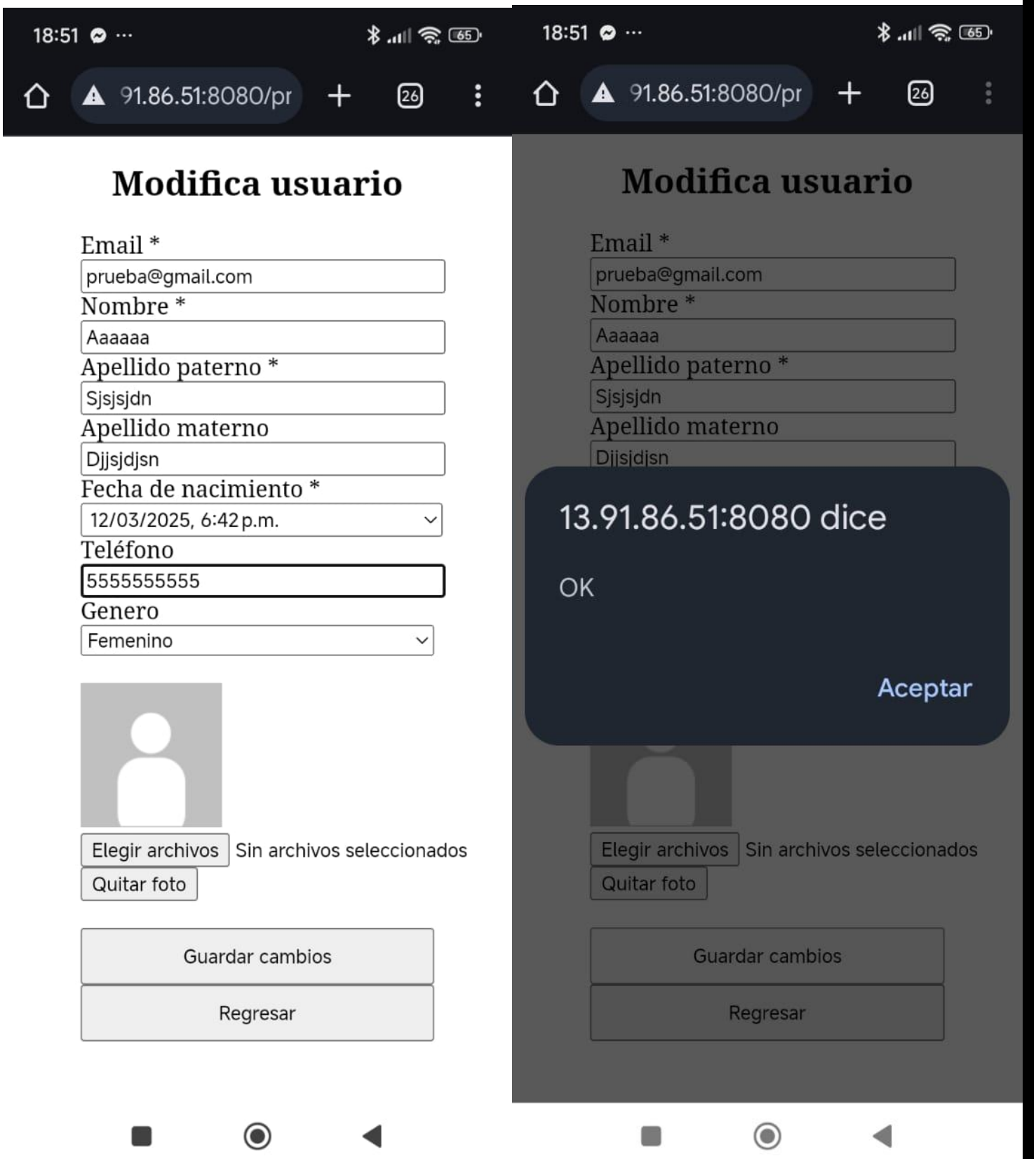


2. Dar clic en el botón “Alta usuario” para dar de alta un nuevo usuario. Capturar los campos y dar clic en el botón “Alta”,

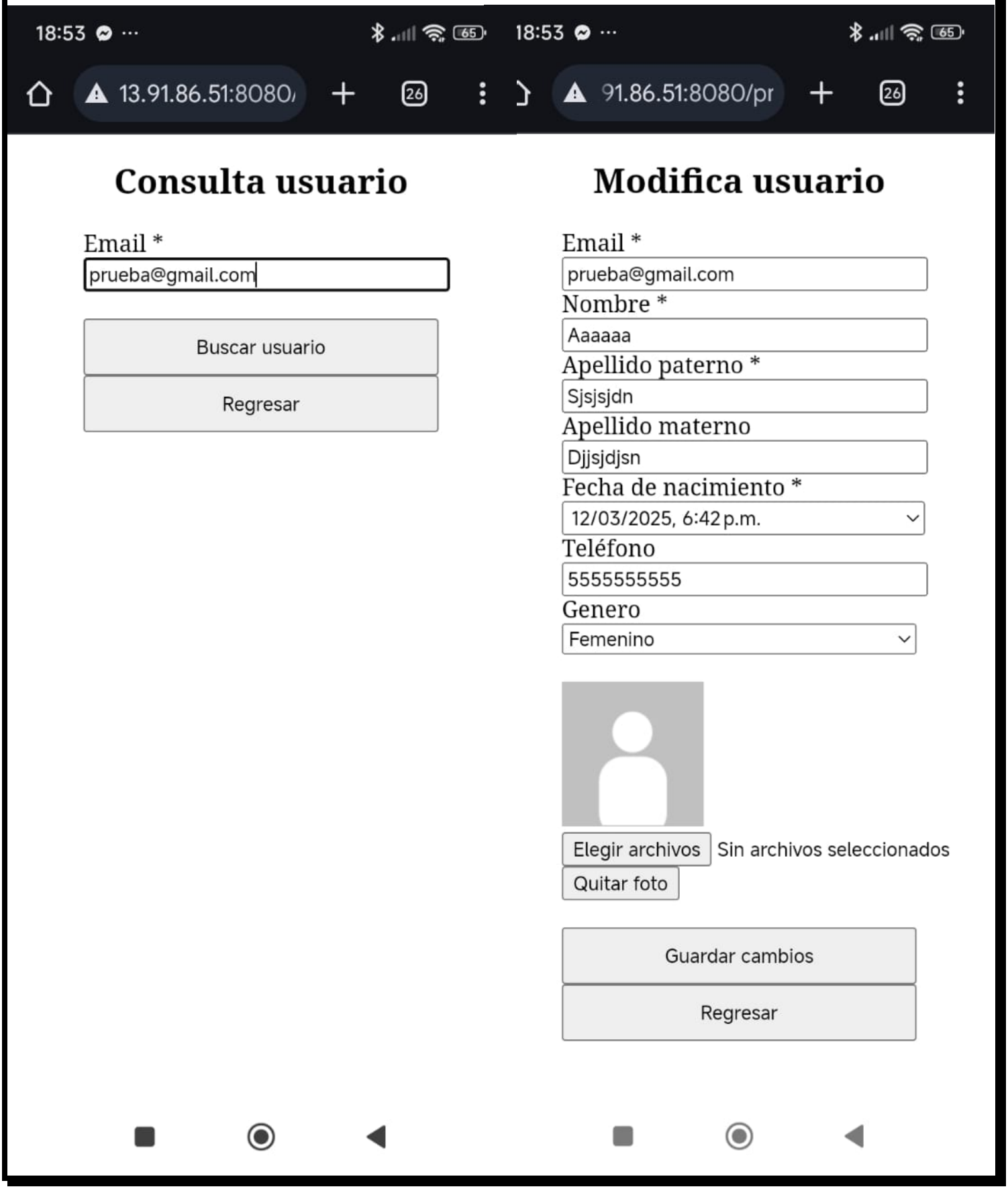
3. Intentar dar de alta otro usuario con el mismo email (se deberá mostrar una ventana de error indicando que el email ya existe).

4. Dar clic en el botón “Consulta usuario” para consultar el usuario dado de alta en el paso 2. Capturar el email y dar clic en el botón “Consulta”.

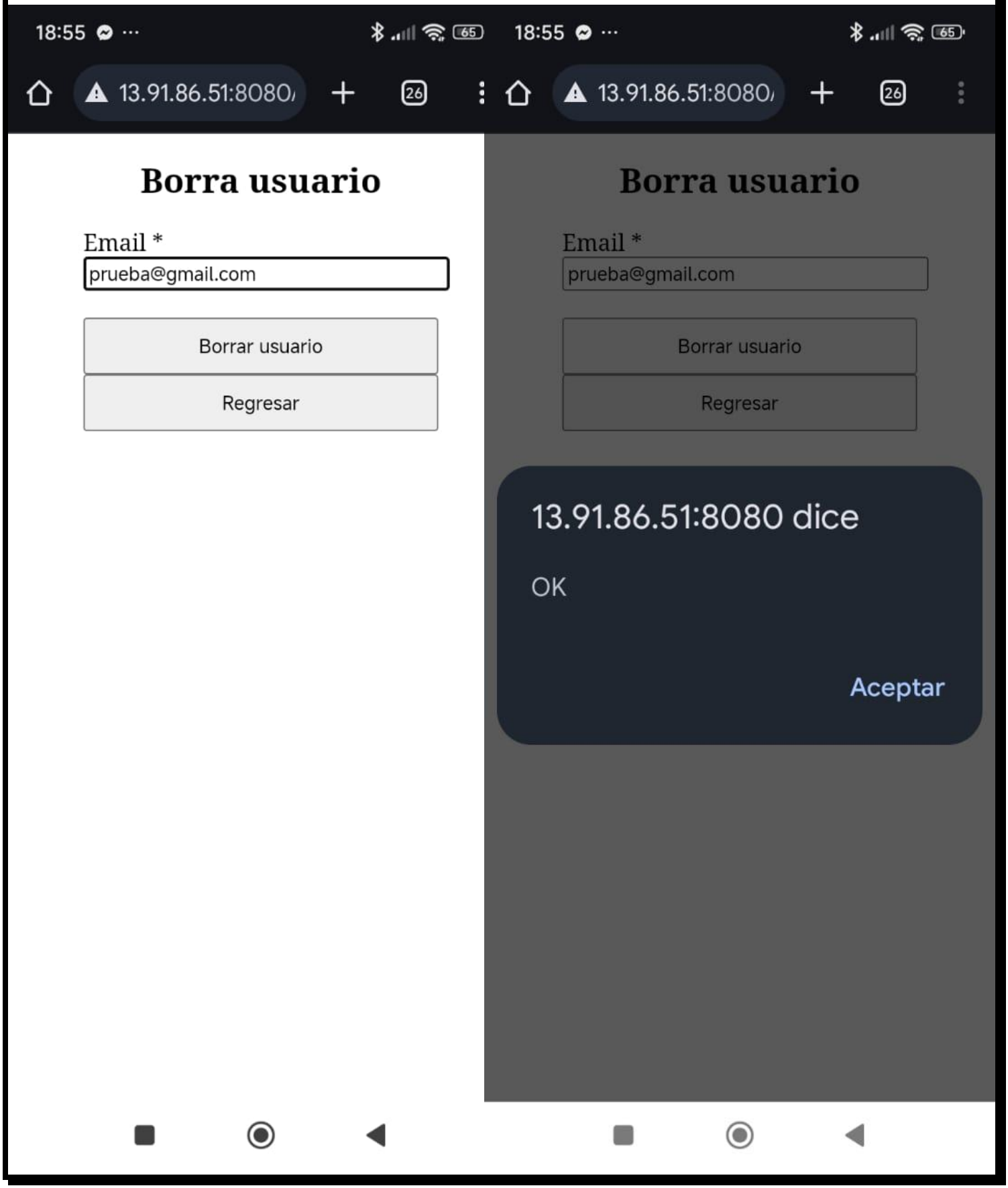
5. Modificar algún dato del usuario y dar clic en el botón “Modifica”.



6. Recargar la página actual y consultar el usuario modificado, para verificar que la modificación se realizó.



7. Dar clic en el botón “Borra usuario” para borrar el usuario. Capturar el email del usuario borrado y dar clic en el botón “Consulta”.



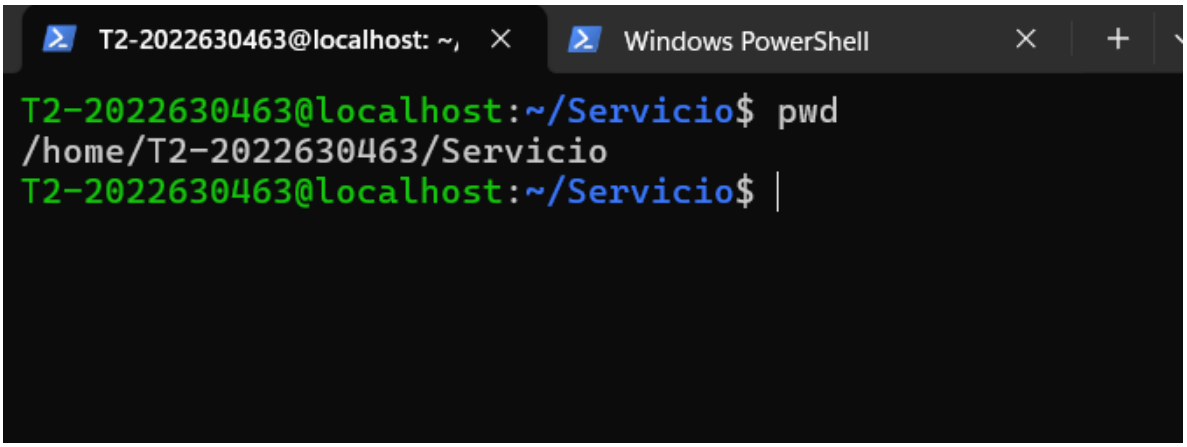
Compilar, empacar y desplegar el servicio web(versión JSON)

1. Remover (undeploy) el servicio web anterior.

rm -rf $CATALINA_HOME/webapps/Servicio.war $CATALINA_HOME/webapps/Servicio
cp Servicio.war $CATALINA_HOME/webapps/.

```
T2-2022630463@localhost:~/Servicio$ sudo rm -rf $CATALINA_HOME/webapps/Servicio.war $CATALINA_HOME/webapps/Servicio
T2-2022630463@localhost:~/Servicio$ sudo cp Servicio.war $CATALINA_HOME/webapps/
T2-2022630463@localhost:~/Servicio$
```

2. Cambiar al directorio donde se encuentran los directorios "servicio_url" y "servicio_json".

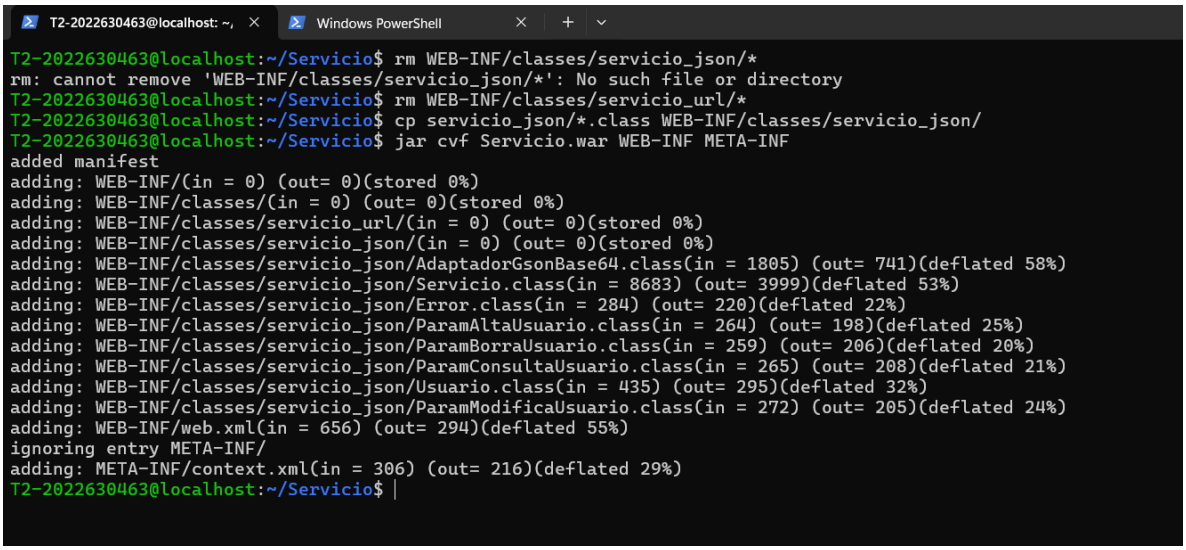


3.- Compilar la clase Servicio.java:

javac -cp $CATALINA_HOME/lib/javax.ws.rs-api-2.0.1.jar:$CATALINA_HOME/lib/gson-2.3.1.jar:. servicio_json/Servicio.java

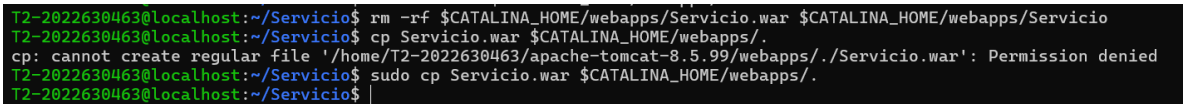
4. Ejecutar los siguientes comandos para crear el servicio web para Tomcat (notar que los servicios web para Tomcat son archivos JAR con la extensión .war):

```
rm WEB-INF/classes/servicio_json/*
rm WEB-INF/classes/servicio_url/*
cp servicio_json/*.class WEB-INF/classes/servicio_json/.
jar cvf Servicio.war WEB-INF META-INF
```



5. Para remover (undeploy) y desplegar (deploy) el servicio web, se deberá eliminar el archivo Servicio.war y el directorio Servicio (en este orden), y luego copiar el archivo Servicio.war al directorio webapps de Tomcat:

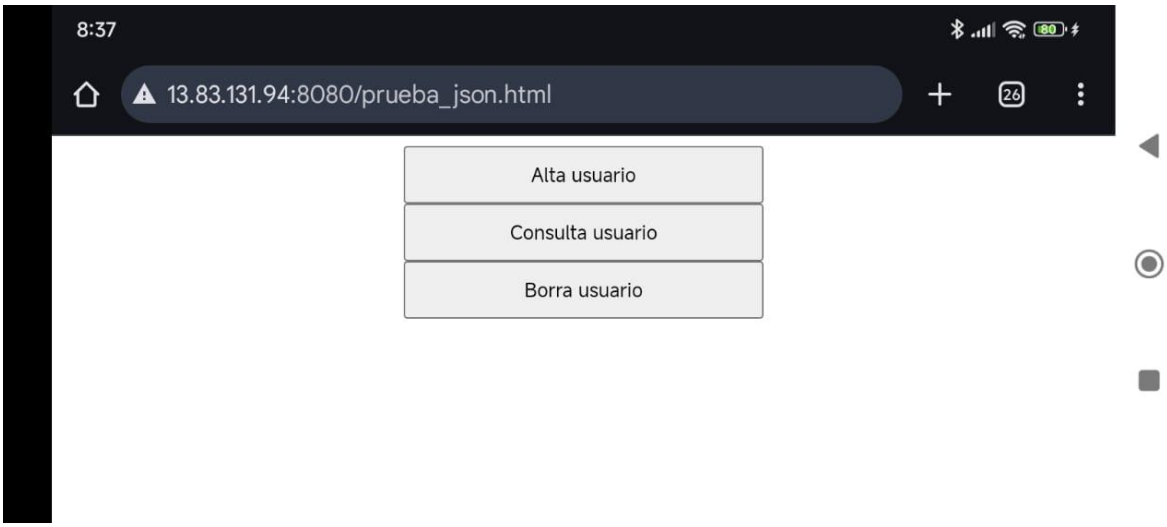
rm -rf $CATALINA_HOME/webapps/Servicio.war
$CATALINA_HOME/webapps/Servicio
cp Servicio.war $CATALINA_HOME/webapps/.



Probar el servicio web utilizando el cliente JSON

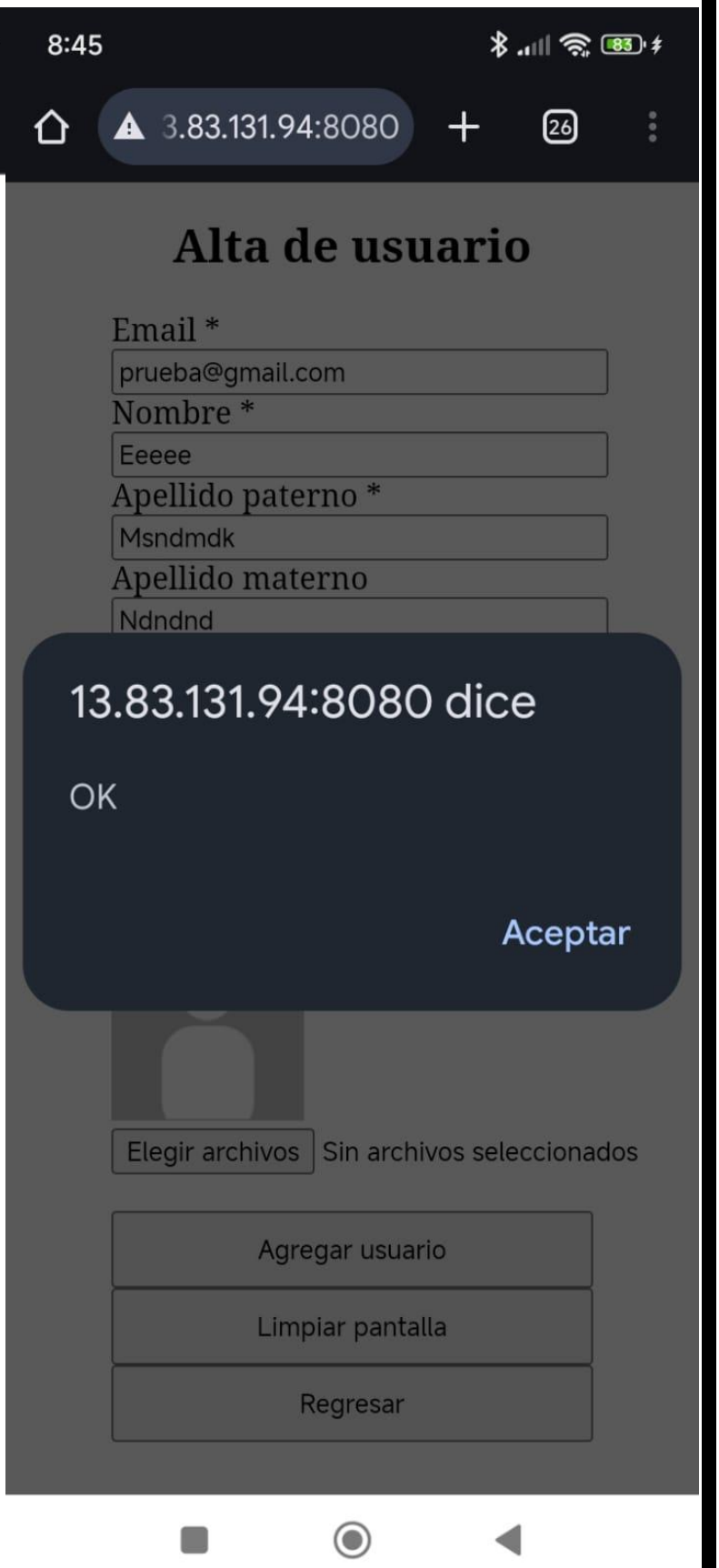
1. Utilizando un teléfono inteligente o una tableta, ingresar la siguiente URL en un navegador:

http://ip-de-la-máquina-virtual:8080/prueba_json.html

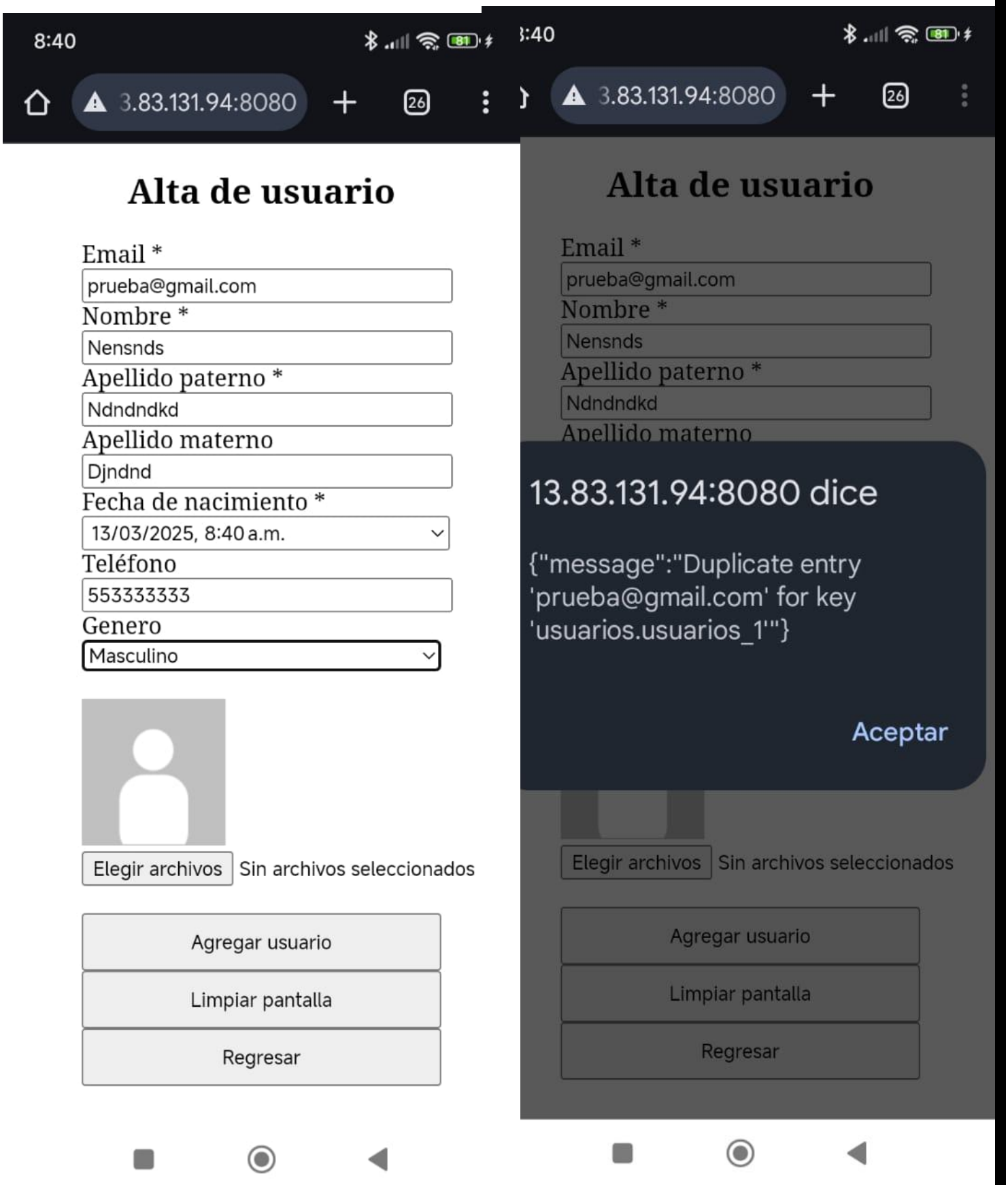


2. Dar clic en el botón “Alta usuario” para dar de alta un nuevo usuario. Capturar los campos y dar clic en el botón “Alta”

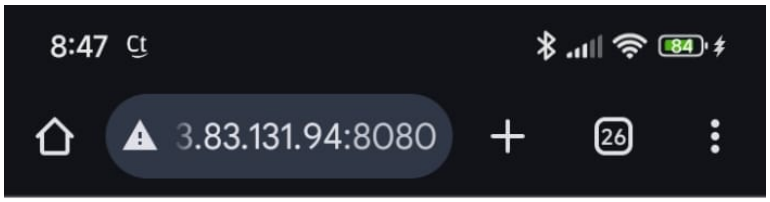

8:45
3.83.131.94:8080
26
Alta de usuario
Email *
prueba@gmail.com
Nombre *
Eeeee
Apellido paterno *
Msndmdk
Apellido materno
Ndndnd
Fecha de nacimiento *
13/03/2025, 8:45 a.m.
Teléfono
5588888858
Genero
Masculino
Elegir archivos
Sin archivos seleccionados
Agregar usuario
Limpiar pantalla
Regresar


8:45
3.83.131.94:8080
26
Alta de usuario
Email *
prueba@gmail.com
Nombre *
Eeeee
Apellido paterno *
Msndmdk
Apellido materno
Ndndnd
13.83.131.94:8080 dice
OK
Aceptar
Elegir archivos
Sin archivos seleccionados
Agregar usuario
Limpiar pantalla
Regresar

3. Intentar dar de alta otro usuario con el mismo email (se deberá mostrar una ventana de error indicando que el email ya existe).



4. Dar clic en el botón “Consulta usuario” para consultar el usuario dado de alta en el paso 2. Capturar el email y dar clic en el botón “Consulta”.



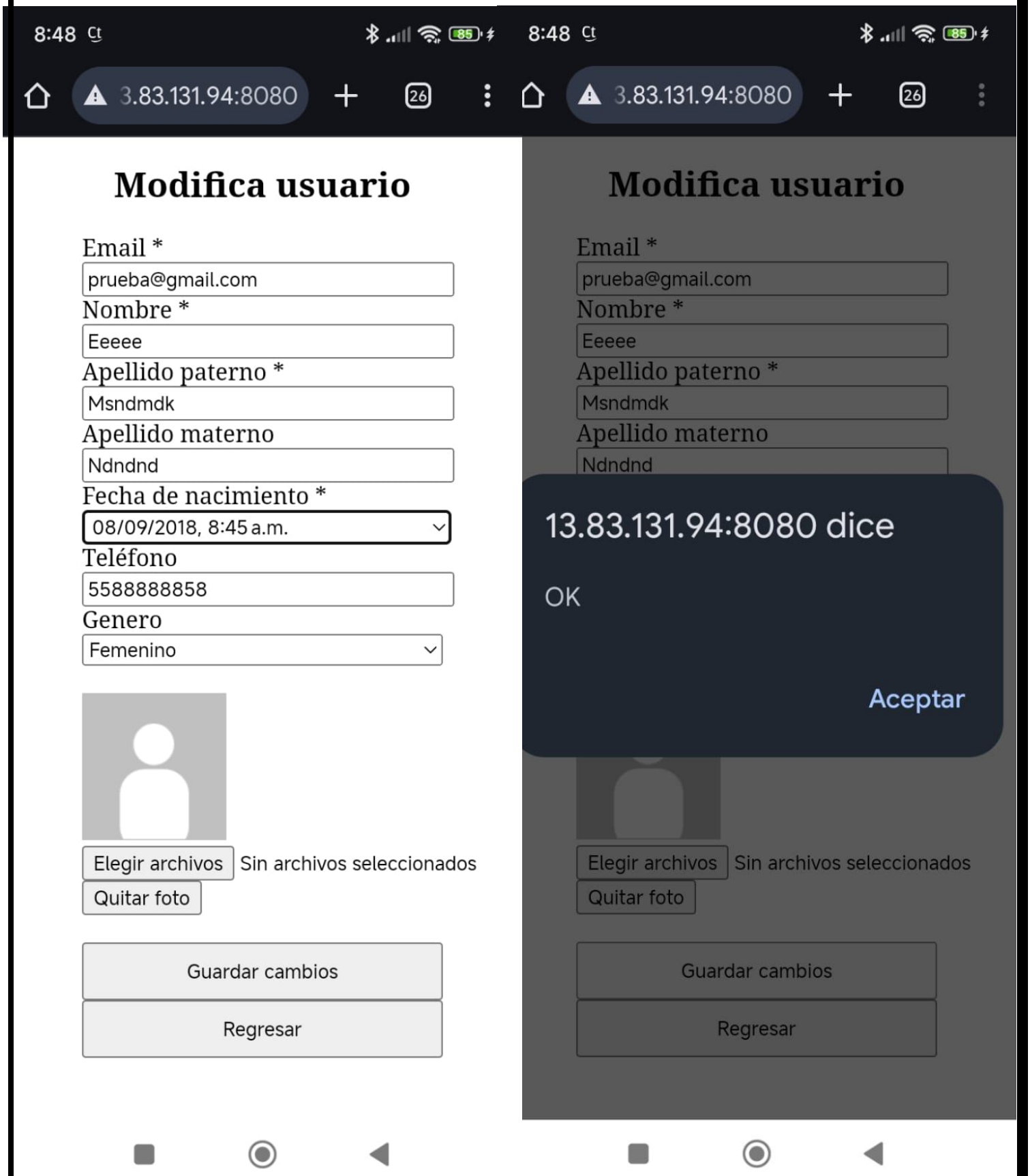
Modifica usuario

Email *
prueba@gmail.com
Nombre *
Eeeee
Apellido paterno *
Msndmdk
Apellido materno
Ndndnd
Fecha de nacimiento *
13/03/2025, 8:45 a.m.
Teléfono
5588888858
Genero
Masculino

Elegir archivos Sin archivos seleccionados
Quitar foto

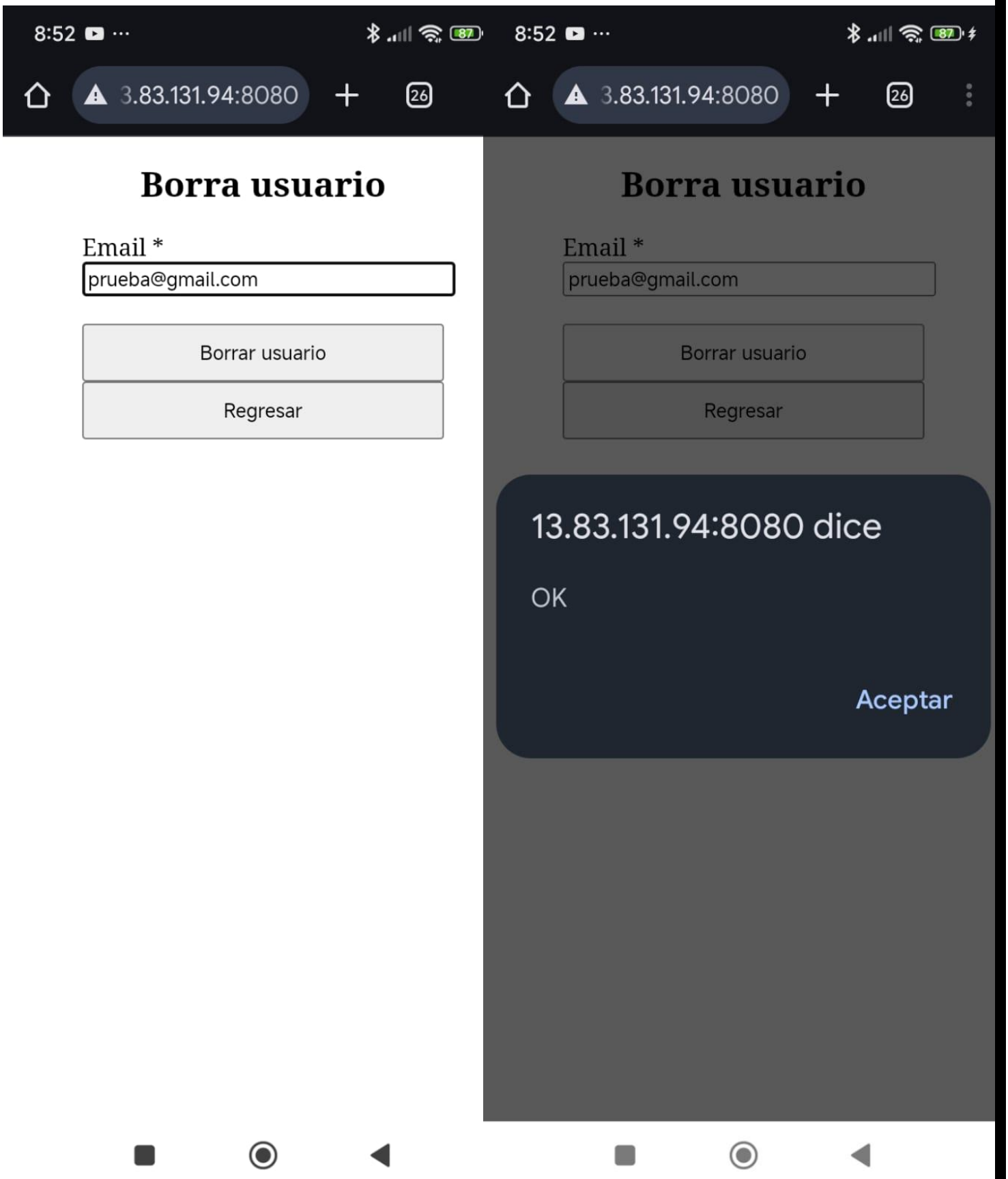
Guardar cambios
Regresar

5. Modificar algún dato del usuario y dar clic en el botón “Modifica”.



6. Recargar la página actual y consultar el usuario modificado, para verificar que la modificación se realizó.

7. Dar clic en el botón “Borra usuario” para borrar el usuario. Capturar el email del usuario borrado y dar clic en el botón “Consulta”.

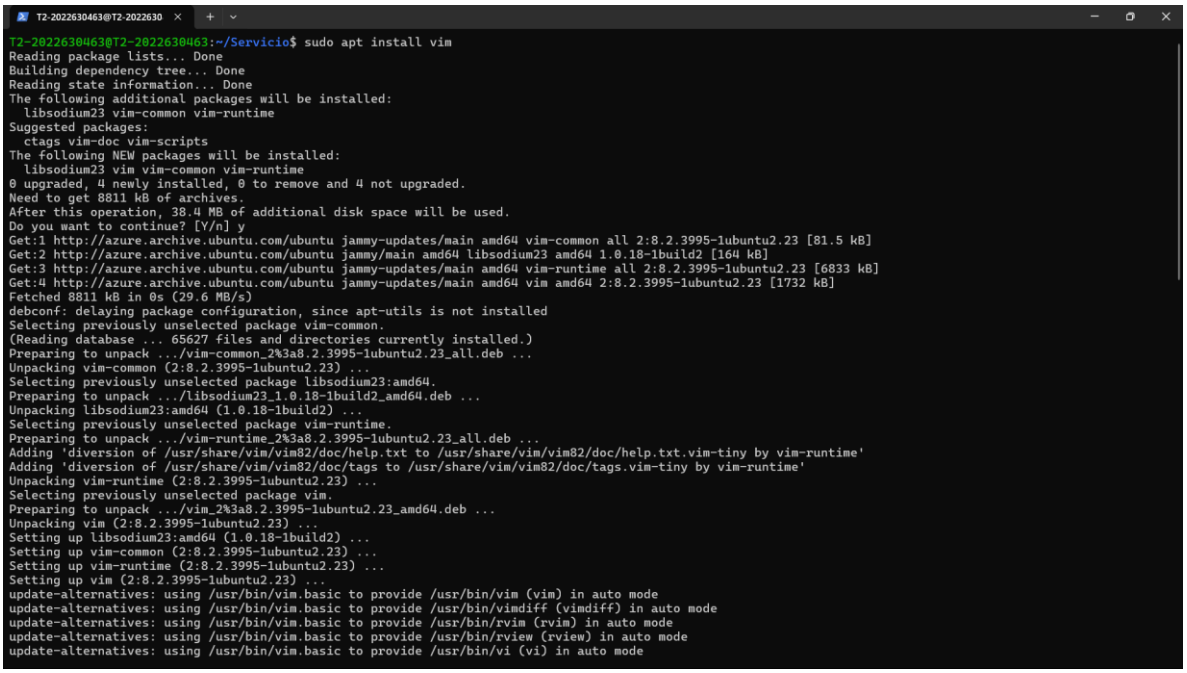


Iniciar Tomcat cuando encienda la máquina virtual

1. Crear el archivo /etc/rc.local ejecutando el siguiente comando:

sudo vi /etc/rc.local

Instalación del comando vi



Al ejecutar el código sale esto:

2. Agregar al archivo lo siguiente:

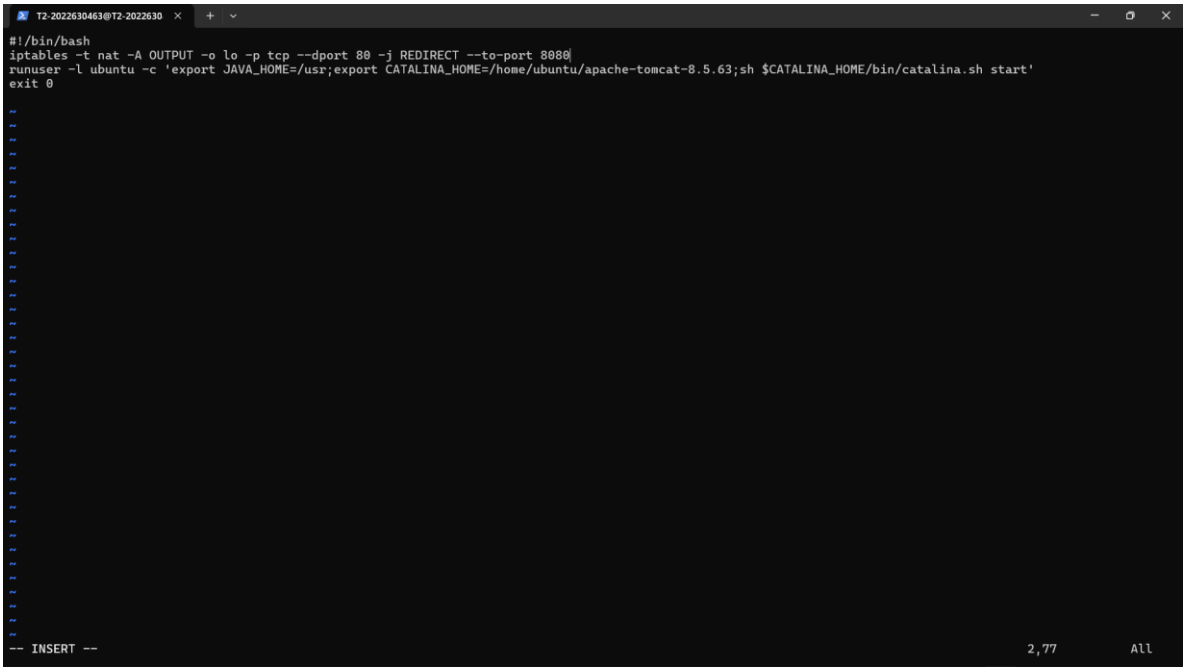
Pulsamos la tecla i para ingresar al modo inserción y poder agregar el siguiente código:

#!/bin/bash

iptables -t nat -A OUTPUT -o lo -p tcp --dport 80 -j REDIRECT --to-port 8080

runuser -l ubuntu -c 'export JAVA_HOME=/usr;export CATALINA_HOME=/home/ubuntu/apache-tomcat-8.5.63;sh $CATALINA_HOME/bin/catalina.sh start'

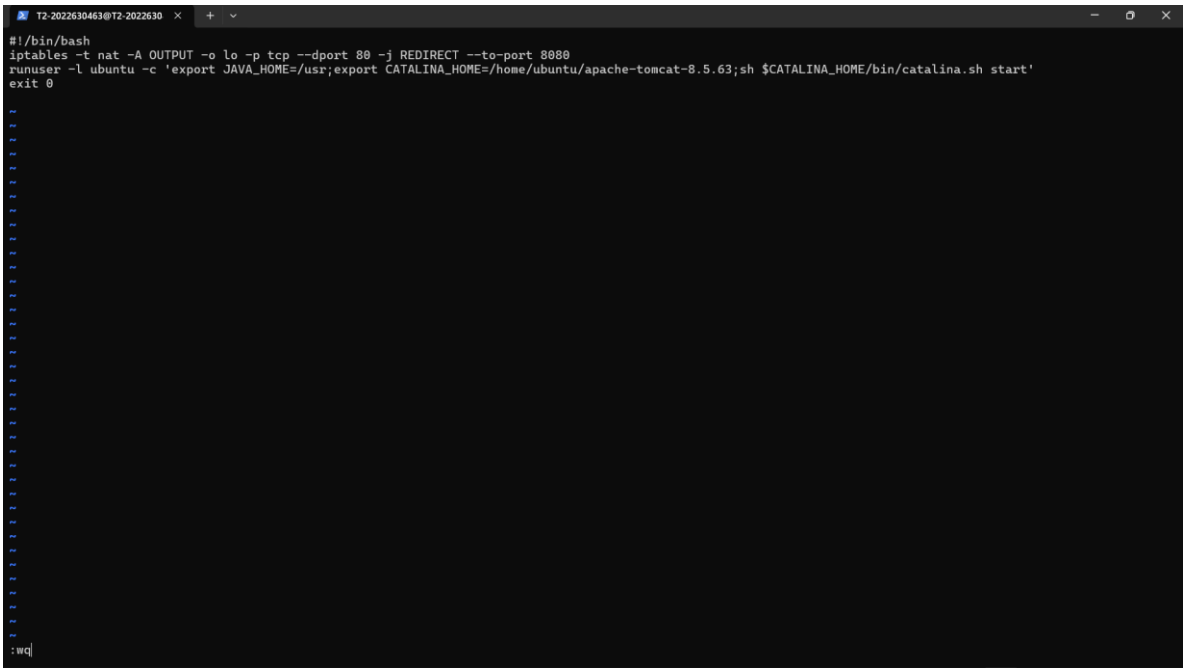
exit 0



3. Guardar el archivo.

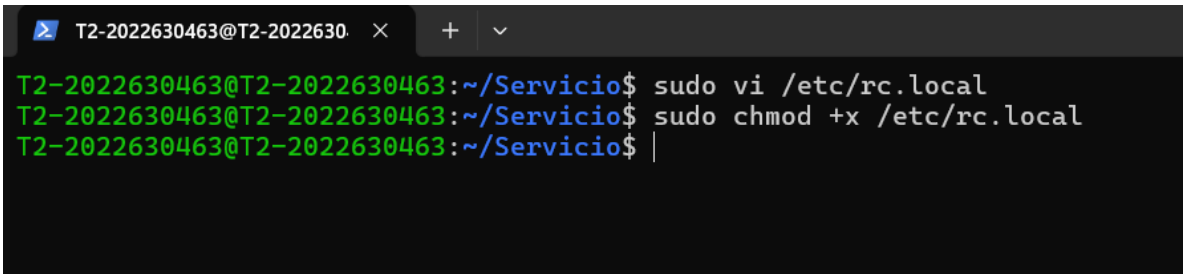
Pulsamos ESC para salir del modo inserción.

Posteriormente escribí :wq y presioné enter para guardar los cambios y cerrar el editor



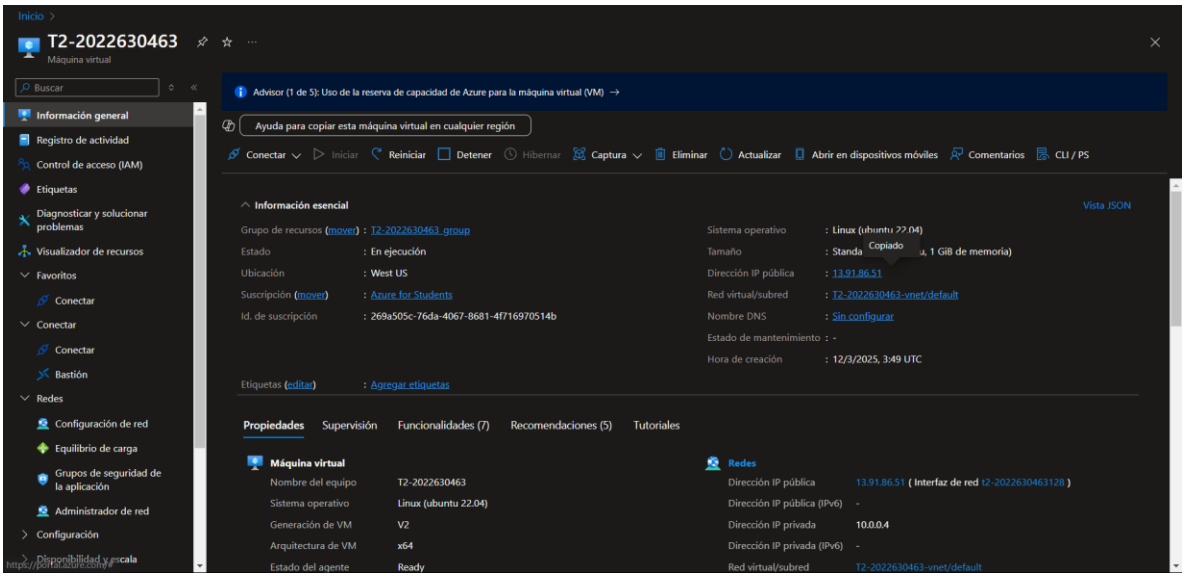
4. Ejecutar el siguiente comando para hacer ejecutable el archivo /etc/rc.local:

sudo chmod +x /etc/rc.local



Crear una imagen de la máquina virtual

1.- Crear la máquina virtual

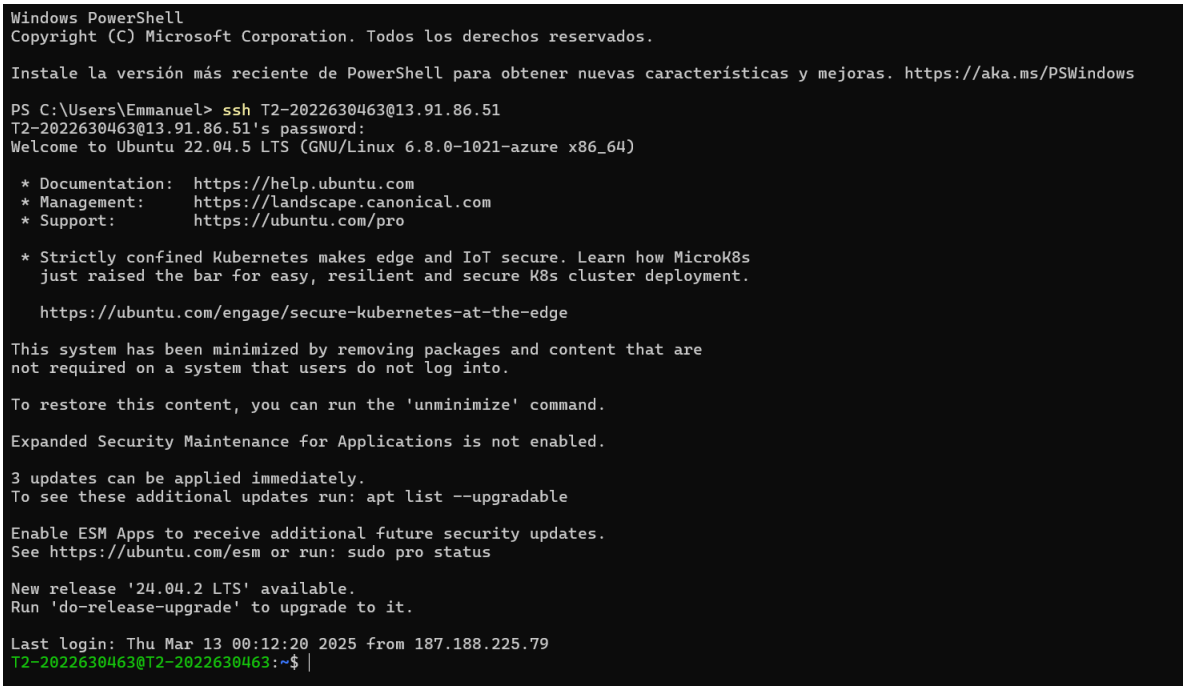


2.- Utilizando el power Shell de Windows nos conectaremos a la máquina virtual a replicar por ssh.

El comando es:

ssh nombre_maquina@ip_publica

Posteriormente les pedirá su contraseña para acceder a la máquina virtual



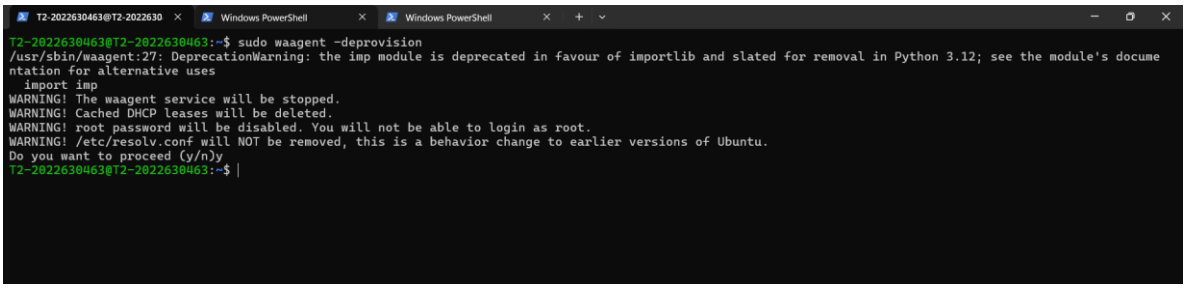
4.- Para generalizar la máquina virtual y eliminar la última cuenta de usuario creada incluyendo el directorio del usuario, ejecutar el comando:

sudo waagent -deprovision+user

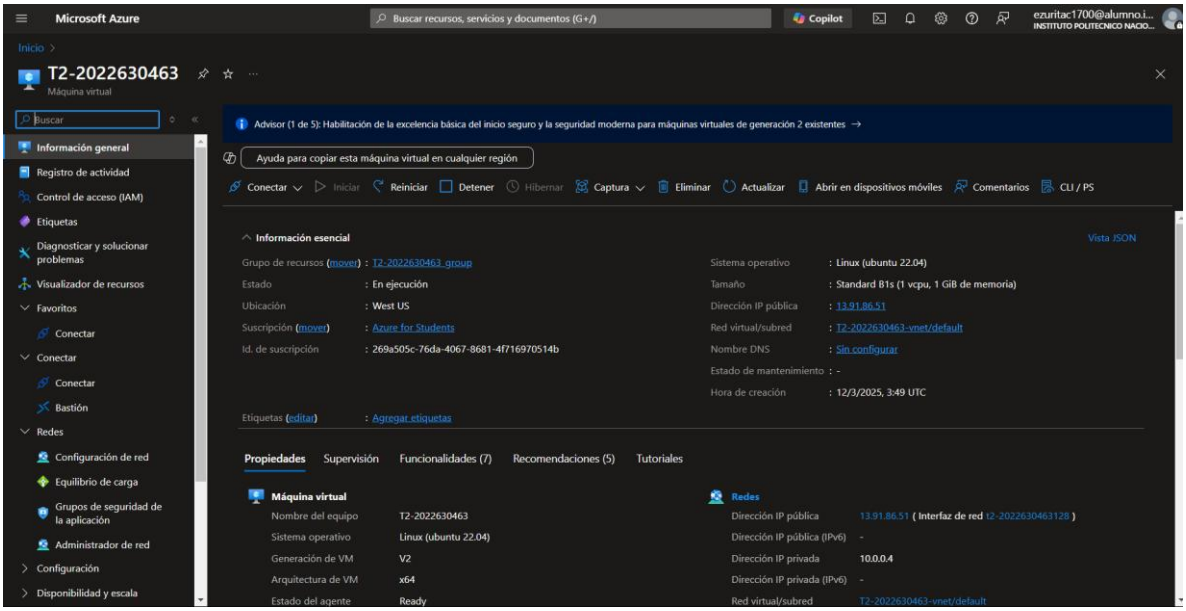
Si se quiere conservar en la imagen la última cuenta de usuario creada, ejecutar el comando:

sudo waagent -deprovision

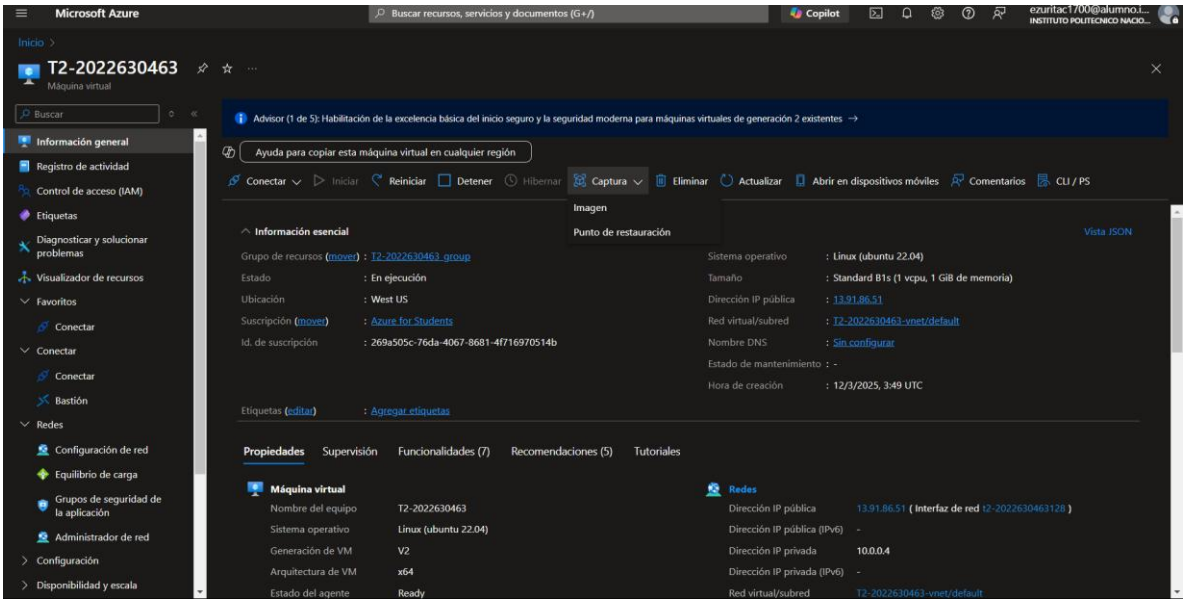
En este caso utilicé el comando **sudo waagent -deprovision**

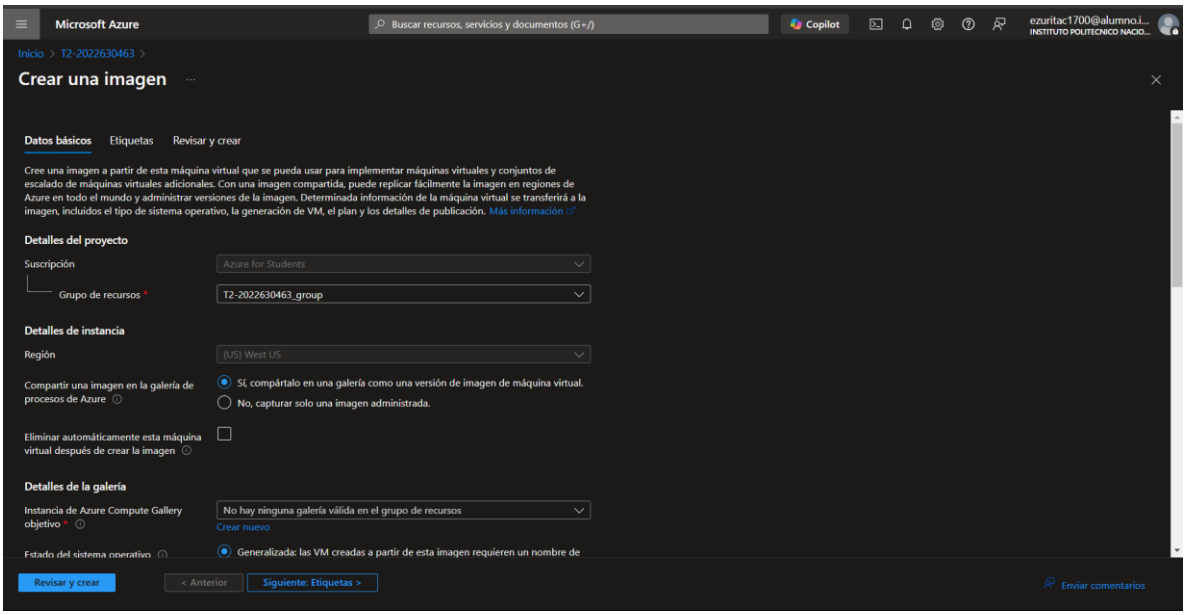


5. En el portal de Azure seleccionar la máquina virtual que se quiera capturar como imagen.

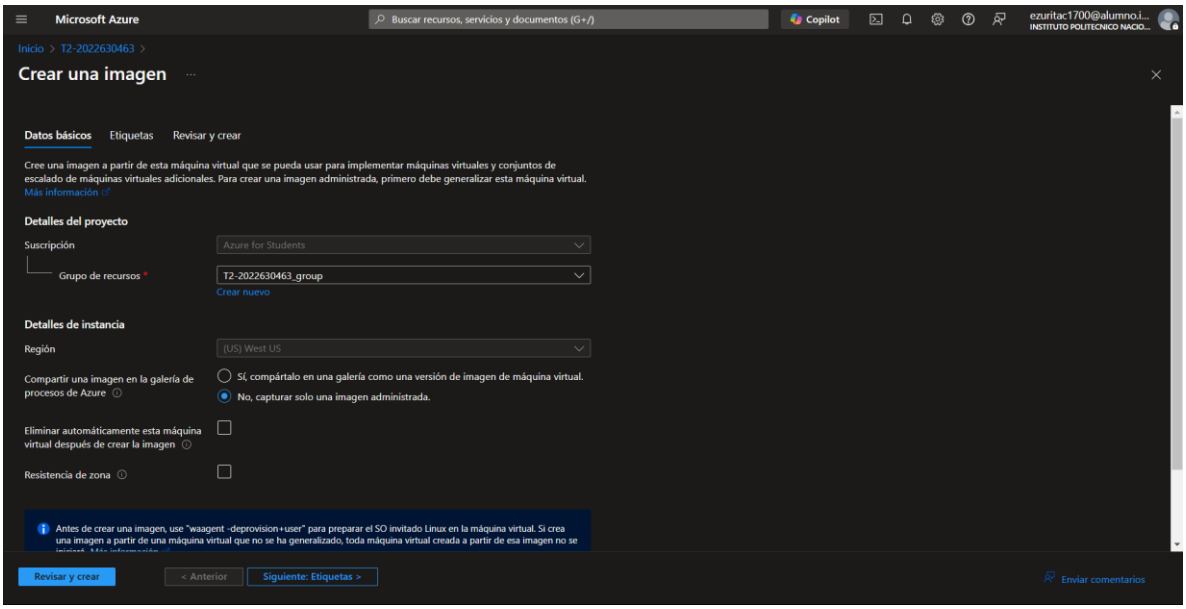


6.- Pulsar en el botón que dice captura y elegir la opción de imagen

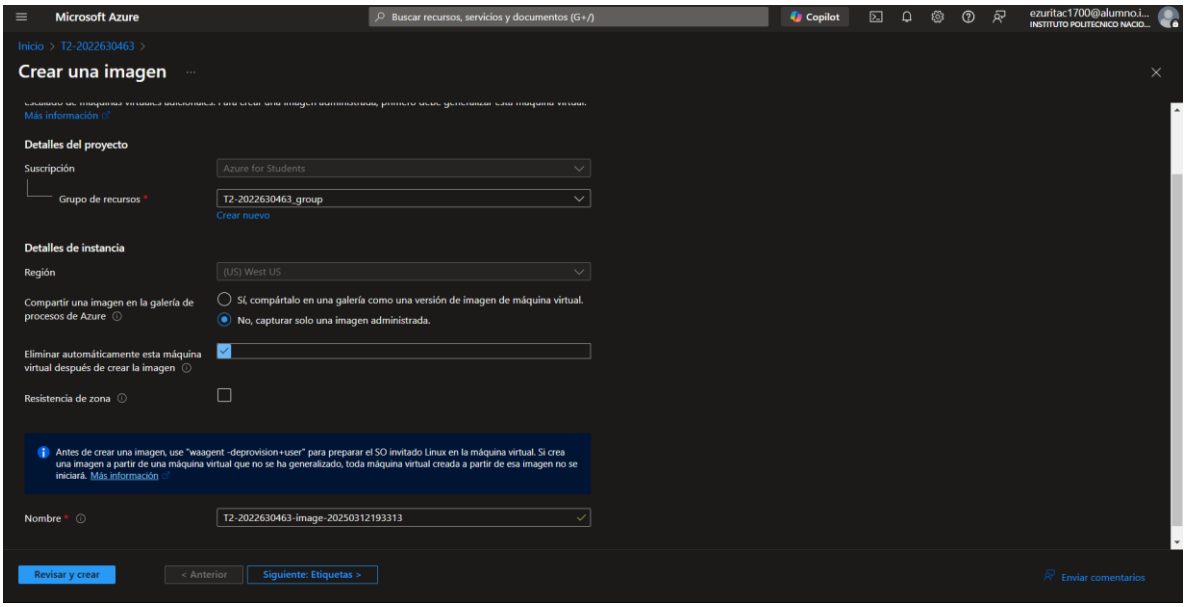




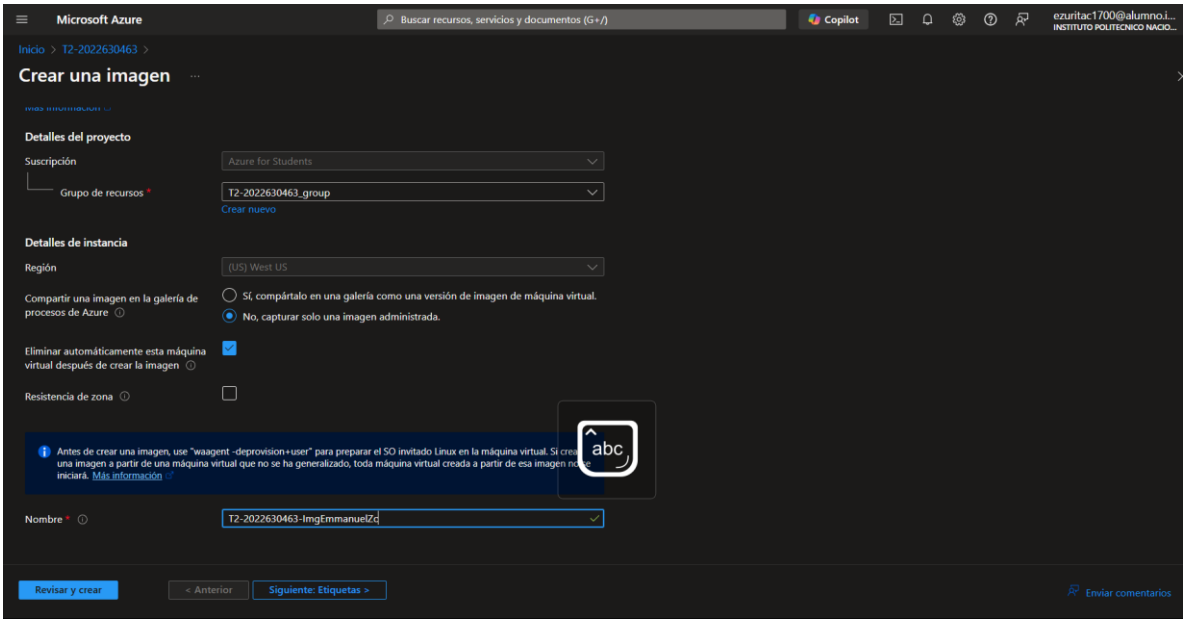
En la opción "Compartir imagen con Shared Image Gallery" seleccionar "No, capturar solo una imagen administrada".

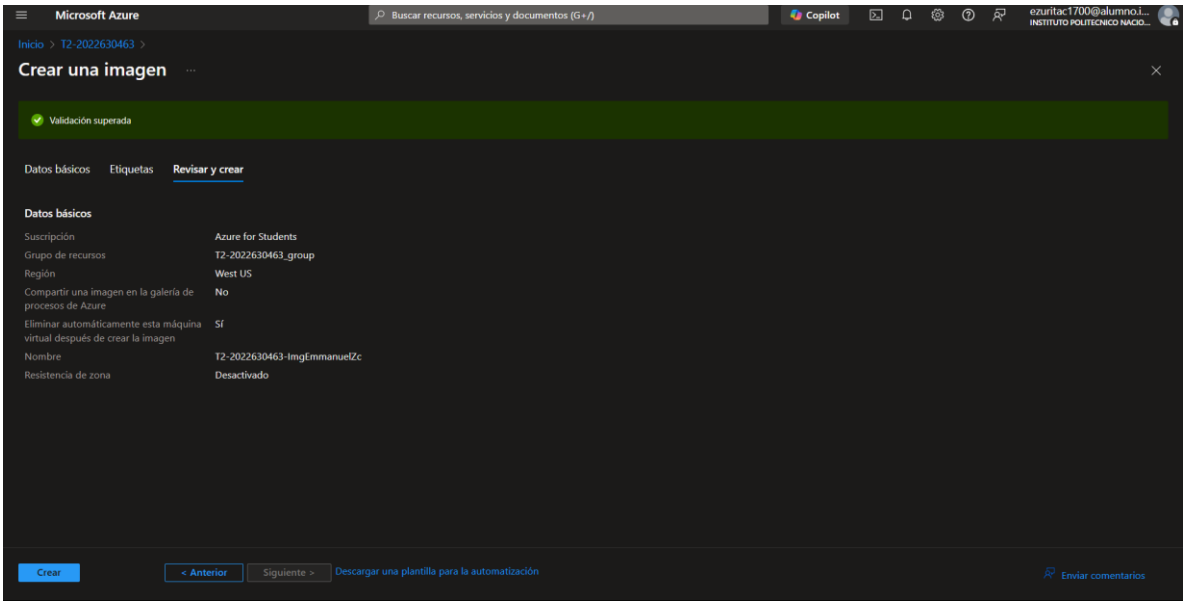


7. Marcar la casilla "Eliminar automáticamente esta máquina virtual después de crear la imagen", ya que una máquina virtual generalizada no se puede iniciar o modificar.

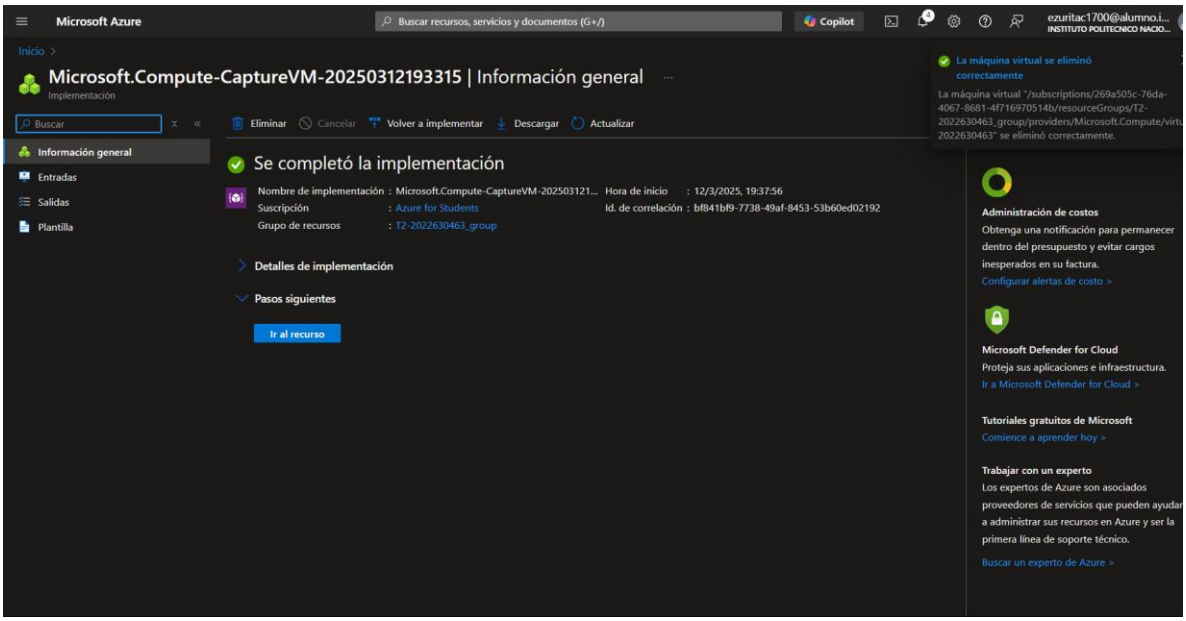


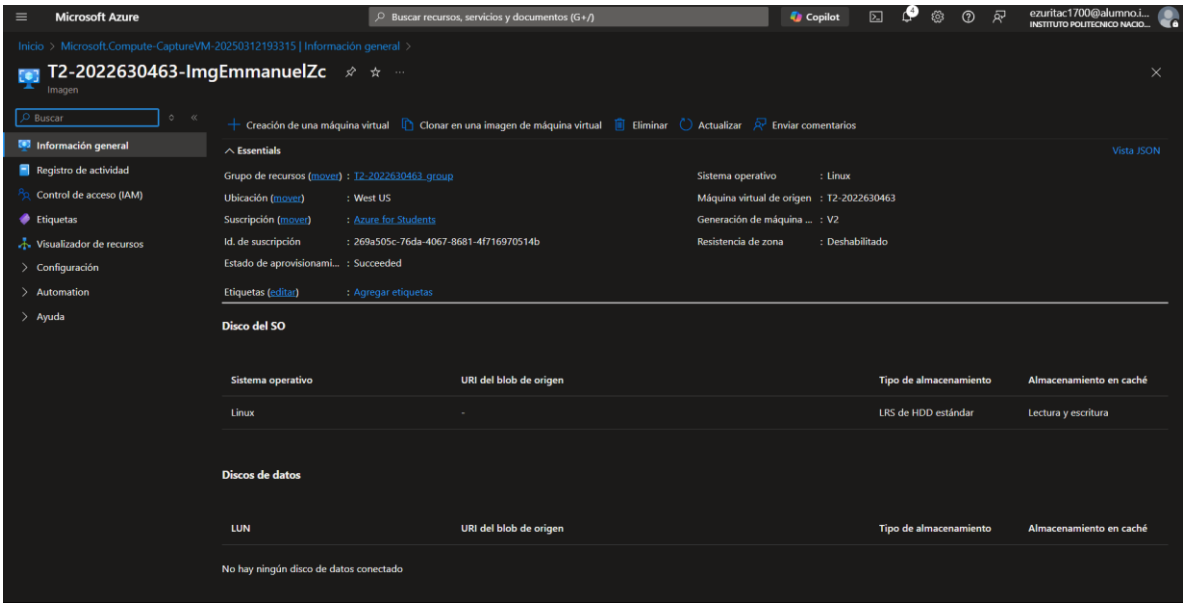
8. Ingresar el nombre de la imagen a crear y posteriormente pulsar Revisar y crear.





Pulsar crear





Uso de chatgpt

Lo utilicé para consultar algunos de los comandos utilizados en Linux

https://chatgpt.com/share/67d237f8-8948-800e-9a8f-a50aa8b131e5

Conclusión

Considero que esta tarea fue más fácil que la anterior. Sin embargo, no es menor su importancia, ya que lo que se busca con esta tarea es configurar bien todos estos elementos para que trabajen juntos de forma eficiente en la nube. Así, la aplicación no solo tendrá una estructura clara y funcional, sino que también será capaz de escalar y adaptarse a las necesidades que puedan surgir más adelante. La clave aquí está en integrar bien las tres capas para que la aplicación sea rápida, segura y fácil de usar.