****

****

**Instituto Politecnico Nacional**

**ESCOM “ESCUELA SUPERIOR DE CÓMPUTO”**

*DESARROLLO DE SISTEMAS DISTRIBUIDOS*

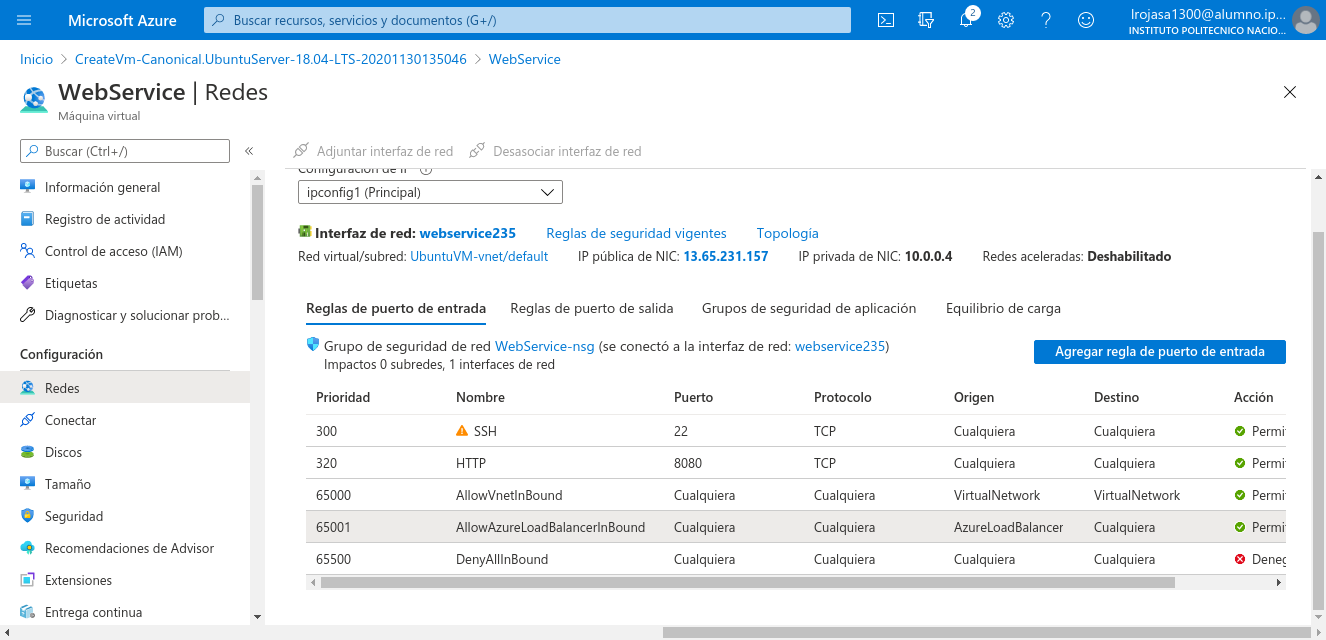
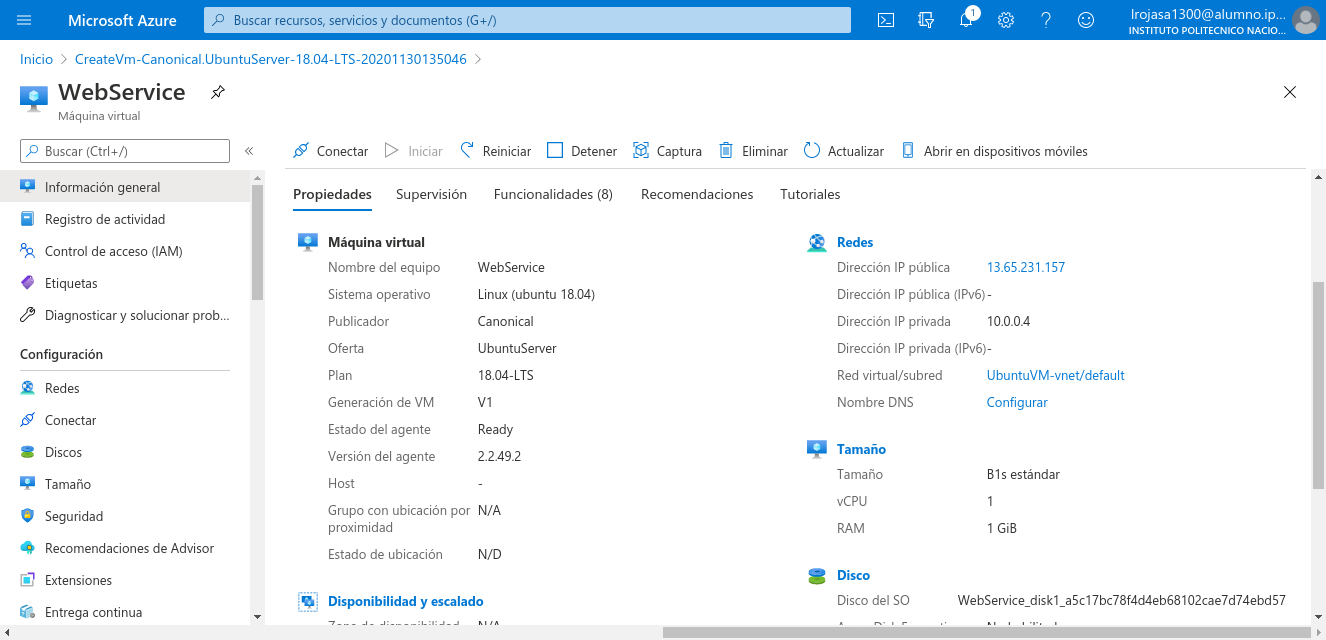
*TAREA 7. IMPLEMENTACIÓN DE UN SERVICIO WEB ESTILO REST*

PROFE: CARLOS PINEDA GUERRERO

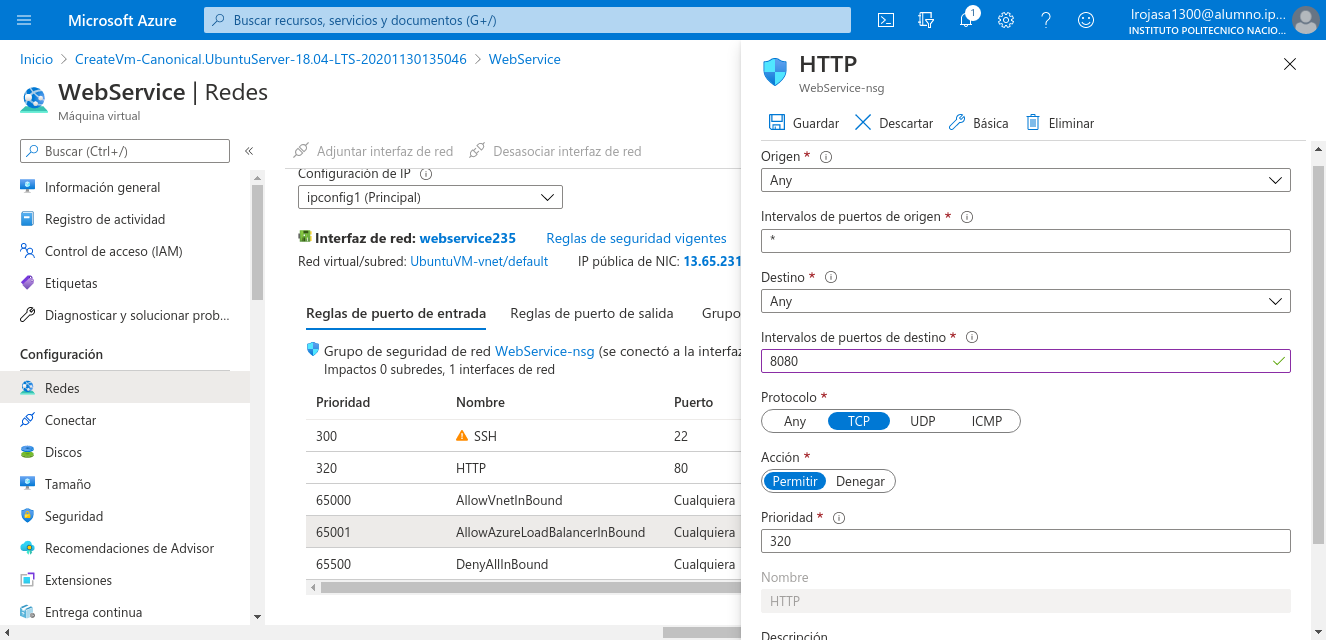
ALUMNO: Rojas Alvarado Luis Enrique

GRUPO: 4CM5

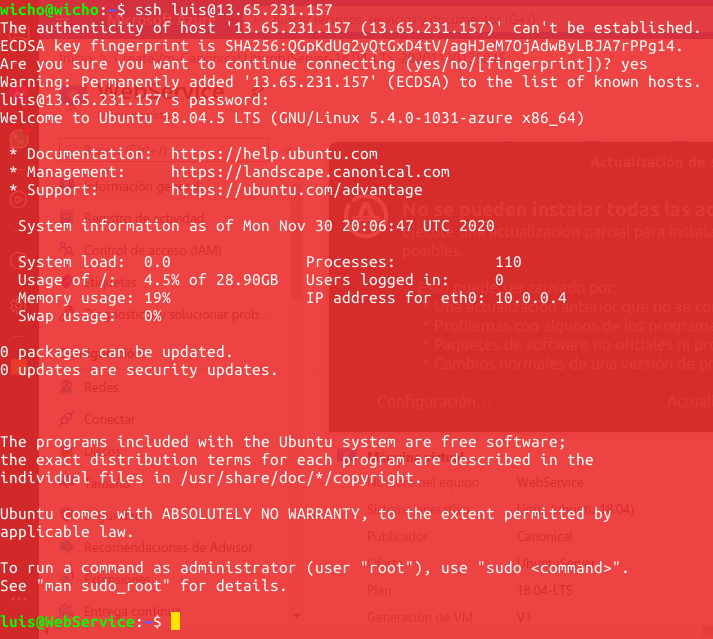
Una vez creada nuestra máquina virtual de Ubuntu 18.04 con 1GB de RAM, activaremos el puerto 8080 de TCP, esto se hace yendo al apartado “Agregar regla de puerto de entrada” en el menú de “Redes” dentro de la información de nuestro recurso.



Al dar click en el botón, podremos activar el puerto 8080 de TCP, simplemente seleccionando la casilla etiquetada con el nombre “TCP” y escribimos el puerto 8080 en el campo “Intervalo de puertos de destino y permitimos.



Hecho esto, procedemos a conectarnos a nuestra máquina virtual. En esta ocasión usaremos el puerto 22 de SSH, simplemente escribiremos nuestro usuario con el que creamos nuestro recurso en Azure y la dirección IP pública, que se encuentra en la información del mismo. Al escribir ssh usuario@[i](mailto:usuario@ip)p\_Pública nos preguntará que si queremos conectarnos intercambiando llaves SSH a lo que responderemos “yes” e ingresaremos la contraseña y nos dejará conectarnos con nuestra máquina virtual.

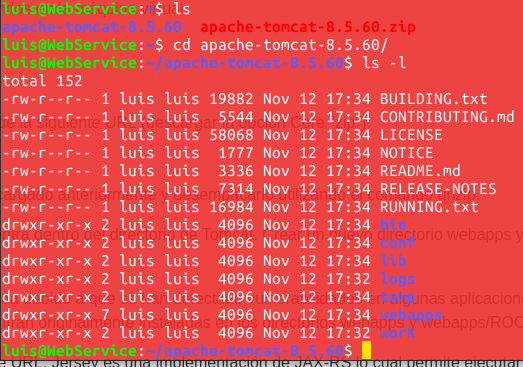
Ahora debemos instalar JDK8 ejecutando los siguientes comandos en la máquina virtual: *sudo apt update* y *sudo apt install openjdk-8-jdk-headless.*

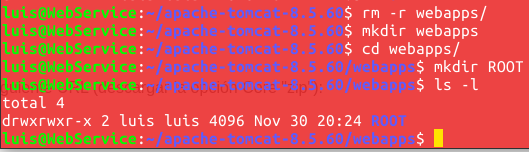
Comprobamos la instalación con java -version

Con wget <https://downloads.apache.org/tomcat/tomcat-8/v8.5.60/bin/apache-tomcat-8.5.60.zip> hacia la url donde se encuentra el recurso .zip de tomcat y desempacarlo utilizando el comando unzip.

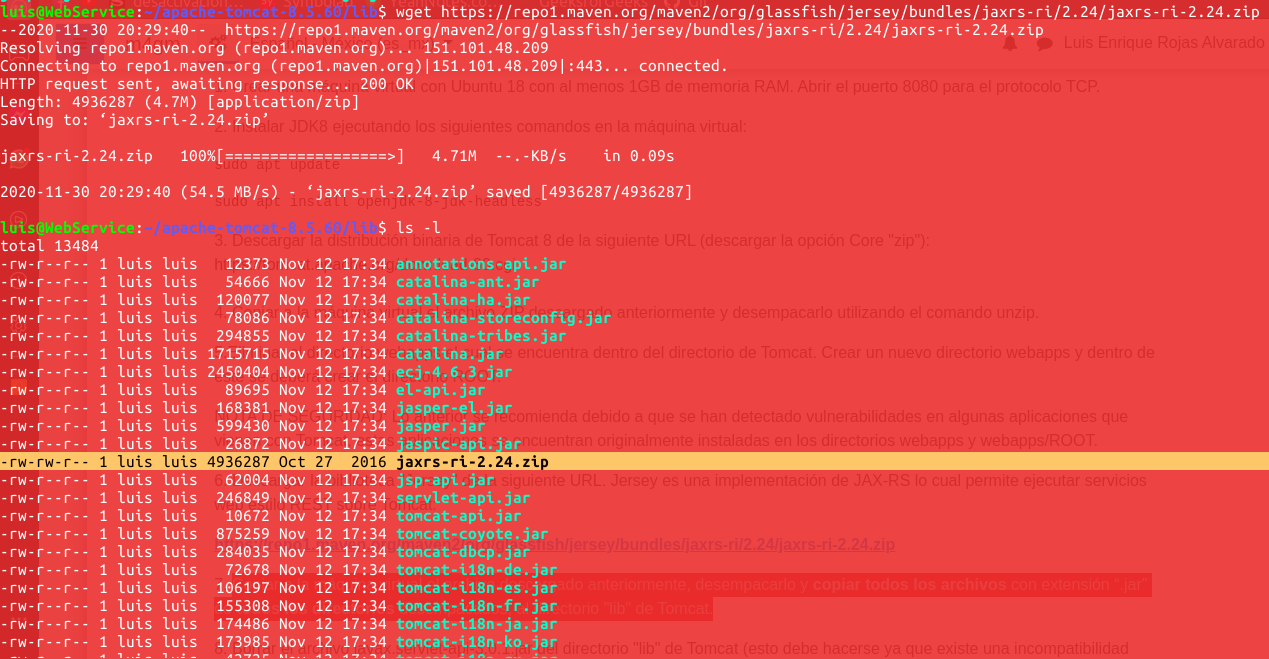


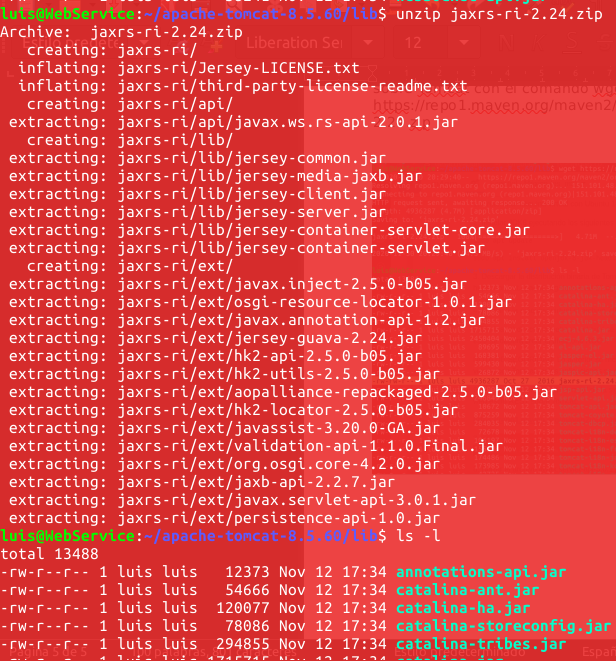
Eliminar el directorio webapps el cual se encuentra dentro del directorio de Tomcat. Crear un nuevo directorio webapps y dentro de éste se deberá crear el directorio ROOT.



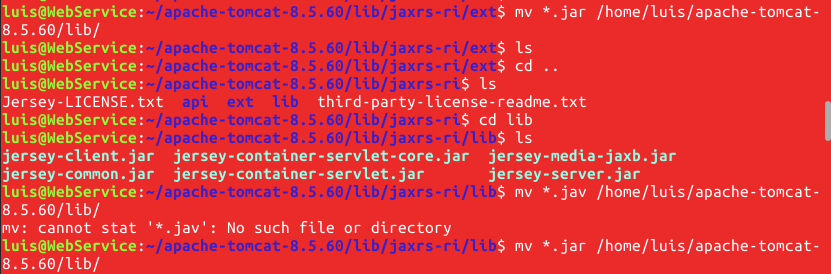


Descargar la biblioteca "Jersey" de la siguiente URL. Jersey es una implementación de JAX-RS lo cual permite ejecutar servicios web estilo REST sobre Tomcat con el comando wget <https://repo1.maven.org/maven2/org/glassfish/jersey/bundles/jaxrs-ri/2.24/jaxrs-ri-2.24.zip>





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

Para lo anterior, nos ubicamos en la carpeta que se descomprimió y nos ubicamos en cada carpeta para escribir el comando mv \*.jar /home/luis/apache-tomcat-8.5.60/lib/ que moverá todos los archivos jar a la carpeta lib de tomcat.

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 2). para lo que escribimos el comando rm -r javax.servlet-api-3.0.1.jar y con esto quedará eliminado.



Descargar el archivo gson-2.3.1.jar con

wget <https://repo1.maven.org/maven2/com/google/code/gson/gson/2.3.1/gson-2.3.1.jar> y copiarlo en la carpeta lib de tomcat



Para descargar el conector JDBC DE MySQL escribimos el comando wget <https://dev.mysql.com/get/Downloads/Connector-J/mysql-connector-java-8.0.22.zip> y lo desempaquetamos y copiamos el archivo jar a la carpeta lib de tomcat. Para esto entramos a la carpeta que descomprimimos de mysql y tecleamos el comando mv mysql-connector-java-8.0.22.jar /home/luis/apache-tomcat-8.5.60/lib/

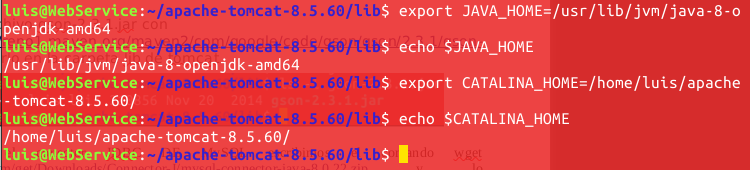


**Iniciar/detener el servidor Tomcat**

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

export CATALINA\_HOME=*aquí va la ruta del directorio de Tomcat 8*

export JAVA\_HOME=/usr/lib/jvm/java-8-openjdk-amd64

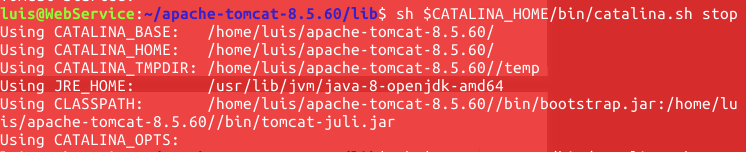
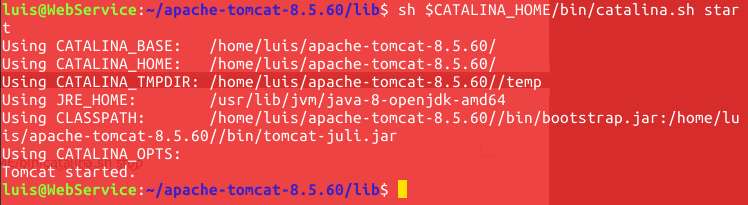


Iniciar la ejecución de Tomcat ejecutando el siguiente comando:

sh $CATALINA\_HOME/bin/catalina.sh start

Para detener la ejecución de Tomcat se deberá ejecutar el siguiente comando:

sh $CATALINA\_HOME/bin/catalina.sh stop



**Instalación de MySQL**

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

sudo apt update

Instalar el paquete default de MySQL:

sudo apt install mysql-server

Ejecutar el script de seguridad:

sudo mysql\_secure\_installation

Press y|Y for Yes, any other key for No: **N**

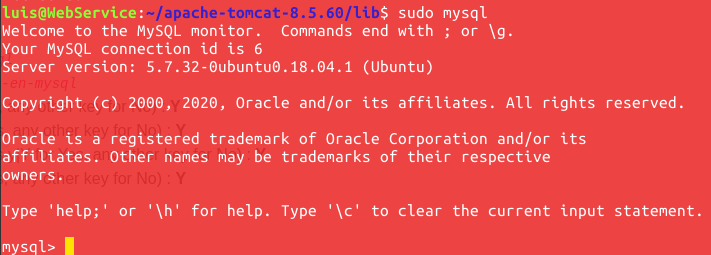
New password:*contraseña-de-root-en-mysql*

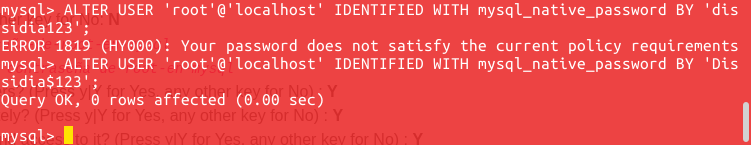
Re-enter new password:*contraseña-de-root-en-mysql*

Remove anonymous users? (Press y|Y for Yes, any other key for No) :**Y**  
Disallow root login remotely? (Press y|Y for Yes, any other key for No) :**Y**  
Remove test database and access to it? (Press y|Y for Yes, any other key for No) :**Y**  
Reload privilege tables now? (Press y|Y for Yes, any other key for No) :**Y**

Ejecutar el monitor de MySQL:

sudo mysql

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

ALTER USER 'root'@'localhost' IDENTIFIED WITH mysql\_native\_password BY '*contraseña-de-root-en-mysql*';

Actualizar los privilegios:

FLUSH PRIVILEGES;



Ejecutar el siguiente comando para salir del monitor de MySQL:

quit

**Crear un usuario en MySQL**

Ejecutar el monitor de MySQL:

*mysql -u root -p*

Crea el usuario "hugo":

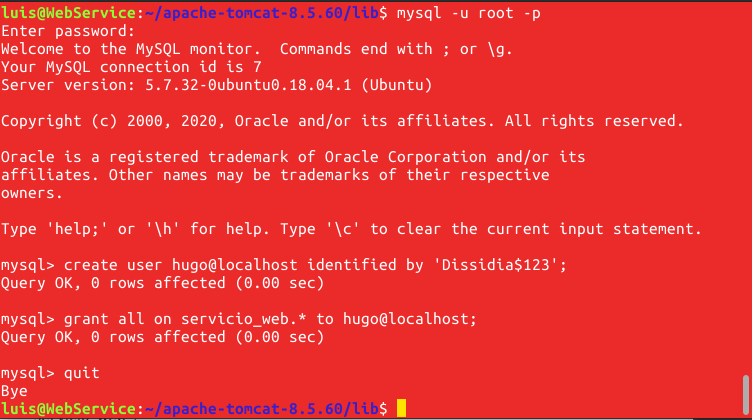
*create user hugo@localhost identified by 'contraseña-del-usuario-hugo';*

Otorgar todos los permisos al usuario "hugo" sobre la base de datos "servicio\_web":

*grant all on servicio\_web.\* to hugo@localhost;*

Ejecutar el siguiente comando para salir del monitor de MySQL:

*quit*

  
**Crear la base de datos**

Ejecutar el monitor de MySQL (notar que ahora se utiliza el usuario "hugo"):

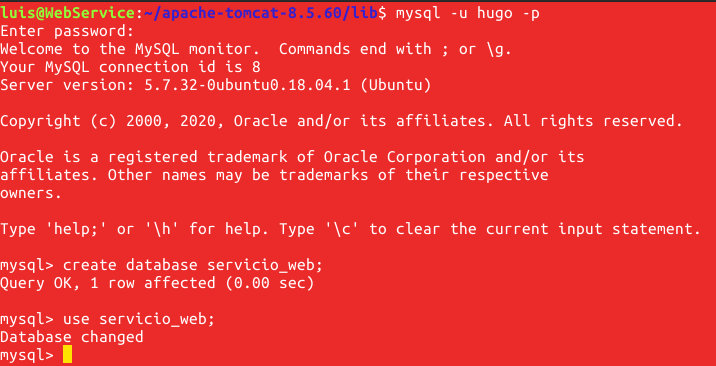
mysql -u hugo -p

Crear la base de datos "servicio\_web":

create database servicio\_web;

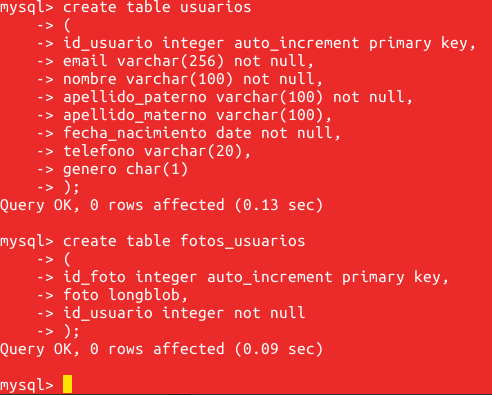
Conectar a la base de datos creada anteriormente:

use servicio\_web;



Crear las tablas "usuarios" y "fotos\_usuarios", así mismo, se crea una regla de integridad referencial y un índice único:

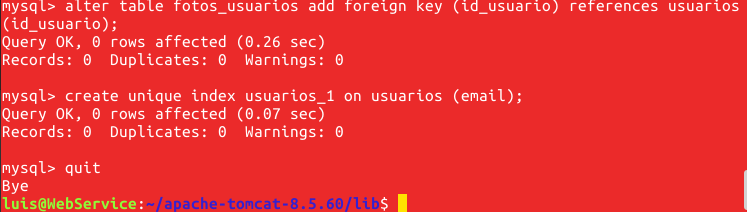
create table usuarios  
(  
 id\_usuario integer auto\_increment primary key,  
 email varchar(256) not null,  
 nombre varchar(100) not null,  
 apellido\_paterno varchar(100) not null,  
 apellido\_materno varchar(100),  
 fecha\_nacimiento date not null,  
 telefono varchar(20),  
 genero char(1)  
);  
create table fotos\_usuarios  
(  
 id\_foto integer auto\_increment primary key,  
 foto longblob,  
 id\_usuario integer not null  
);



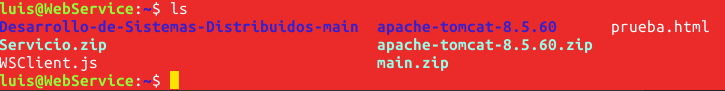
alter table fotos\_usuarios add foreign key (id\_usuario) referencesusuarios(id\_usuario);  
create unique index usuarios\_1 on usuarios(email);

Salir del monitor de MySQL:

quit

**Compilar, empacar y desplegar el servicio web**

Descargar desempacar los archivos. Para más comodidad se subieron los archivos a mi repositorio personal de gitHub, de donde descargaremos el repositorio para poder usar los archivos ‘[Servicio.zi](https://m4gm.com/moodle/mod/resource/view.php?id=1515)p’, ‘Prueba.html’, ‘usuario\_sin\_foto.png’ y ‘WSClient.js’. Todo esto con wget <https://github.com/Wicho1313/Desarrollo-de-Sistemas-Distribuidos/archive/main.zip>

Eliminamos la carpeta Desarrollo-de-Sistemas-Distribuidos-main del repositorio deGitHub que descargamos anteriormente, puesto que no lo vamos a utilizar más.

Definir la variable de ambiente CATALINA\_HOME:

*export CATALINA\_HOME=aquí va la ruta completa del directorio de Tomcat 8*

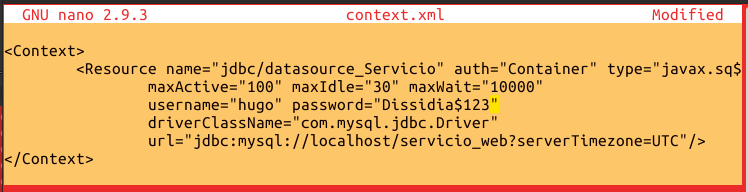
Cambiar al directorio dónde se desempacó el archivo [Servicio.zip](https://m4gm.com/moodle/mod/resource/view.php?id=1515)(en ese directorio se encuentra el directorio "negocio").

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:. negocio/Servicio.java

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. El usuario "hugo" fue creado en el paso 2 de la sección **Crear un usuario en MySQL**.

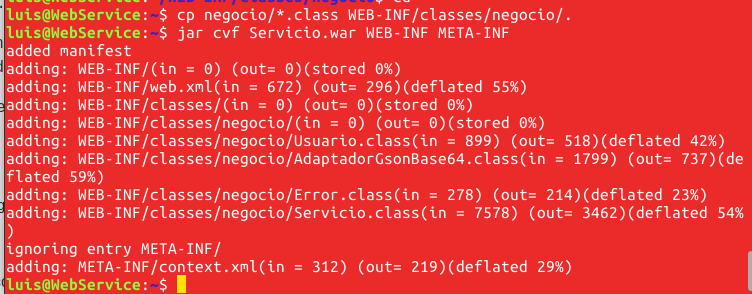
Para eso tenemos que escribir el comando nano y el nombre del archivo, situarnos en los campos que vamos a editar (username, password) y escribir ‘hugo’ y ‘Dissidia$123’ que previamente configuramos.

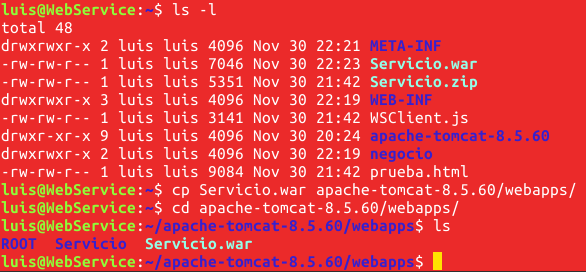
Ejecutar los siguientes comandos para crear el servicio web para Tomcat (notar que los servicios web para Tomcat son archivos JAR con la extensión .war):

rm WEB-INF/classes/negocio/\*

cp negocio/\*.class WEB-INF/classes/negocio/.

jar cvf Servicio.war WEB-INF META-INF

Para desplegar (*deploy*) el servicio web, copiar el archivo **Servicio.war** al directorio "webapps" de Tomcat. Notar que Tomcat desempaca automáticamente los archivos con extensión .war que se encuentran en el directorio webapps de Tomcat.



Para eliminar el servicio web se deberá eliminar el archivo "Servicio.war" y el directorio "Servicio", en éste orden.

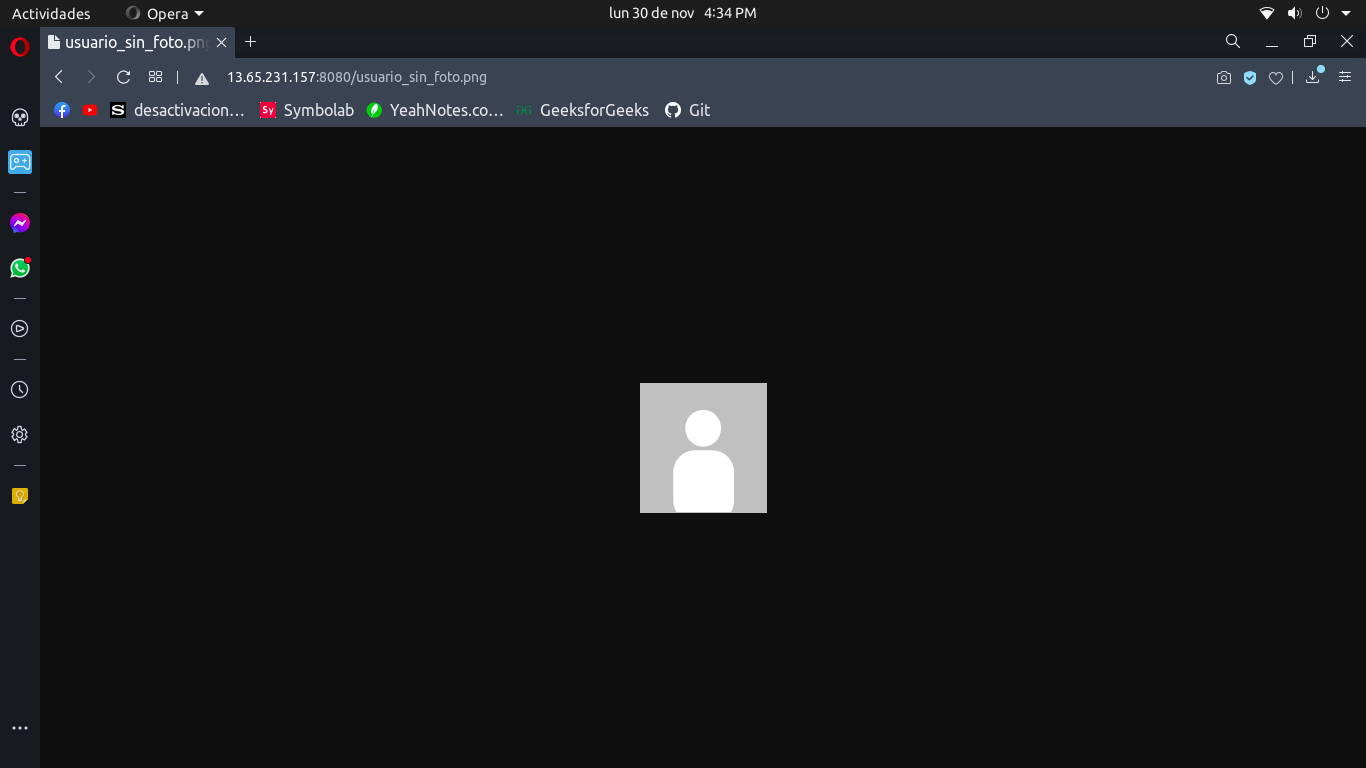
Cada vez que se modifique el archivo Servicio.java se deberá compilar, generar el archivo Servicio.war, borrar el archivo Servicio.war y el directorio Servicio del directorio webapps de Tomcat, y copiar el archivo Servicio.war al directorio webapps de Tomcat.

**Probar el servicio web utilizando HTML-Javascript**

Copiar el archivo [**usuario\_sin\_foto.png**](https://m4gm.com/moodle/mod/resource/view.php?id=1517)al subdirectorio webapps/ROOT de Tomcat.

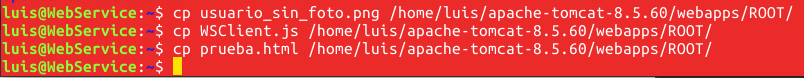
Notar que todos los archivos que se encuentran en el directorio webapps/ROOT de Tomcat son accesibles públicamente.

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](https://m4gm.com/moodle/mod/resource/view.php?id=1517)

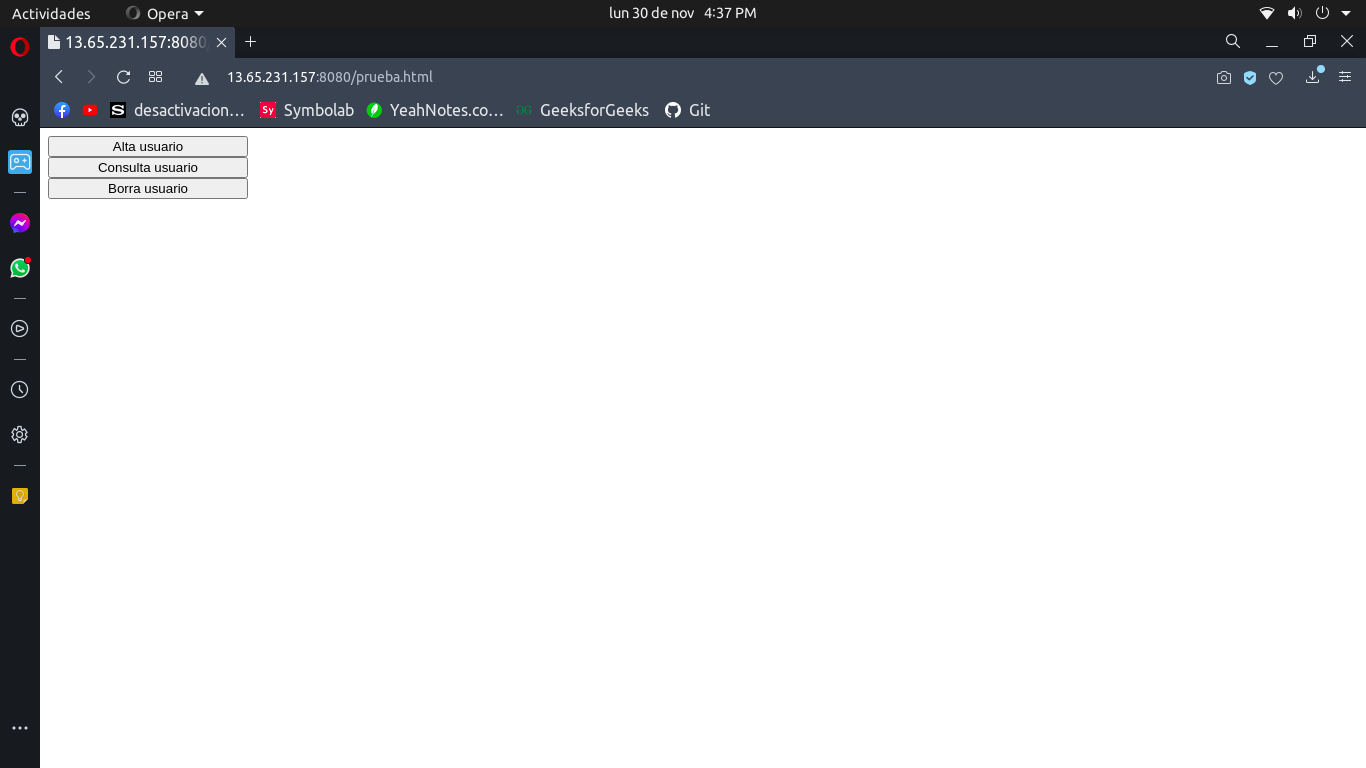
Copiar el archivo [**WSClient.js**](https://m4gm.com/moodle/mod/resource/view.php?id=1516)al directorio webapps/ROOT de Tomcat.

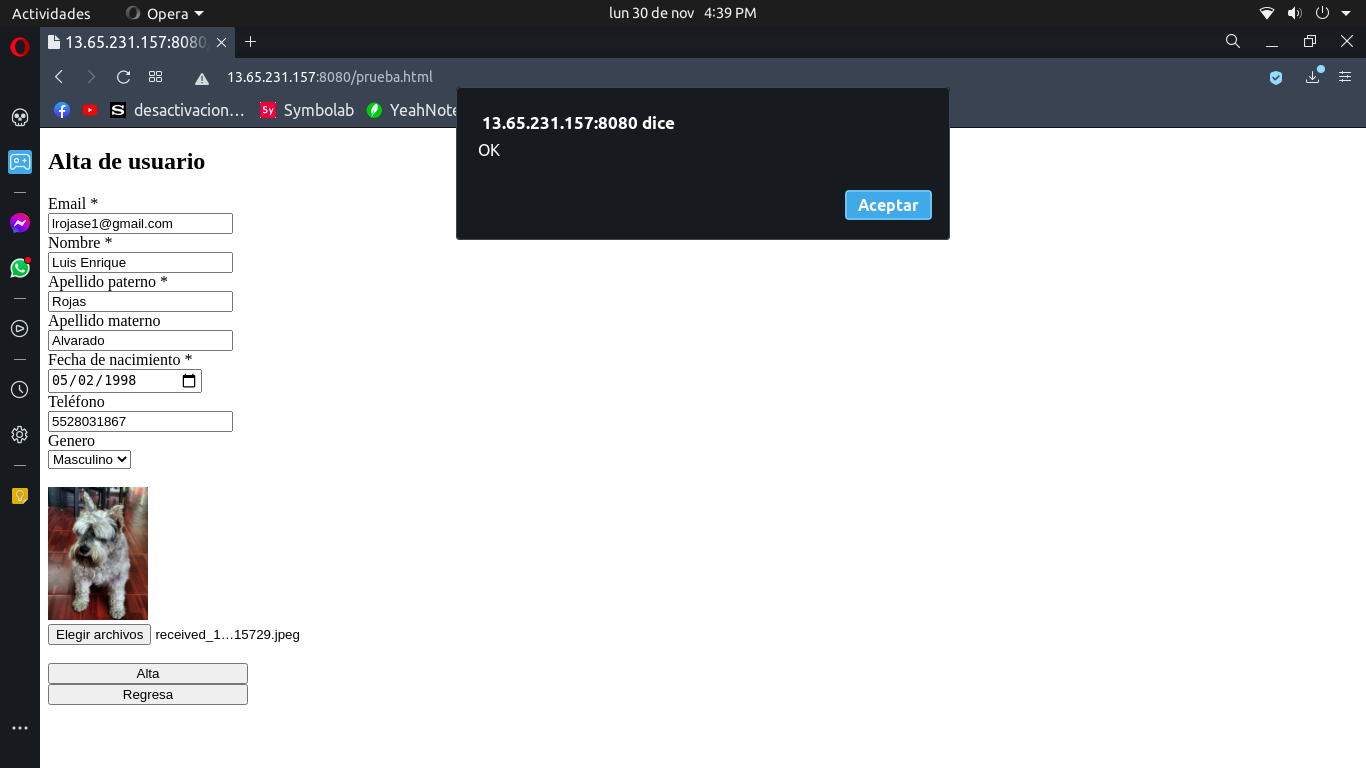
Copiar el archivo [prueba.html](https://m4gm.com/moodle/mod/resource/view.php?id=1518) al directorio webapps/ROOT de Tomcat.



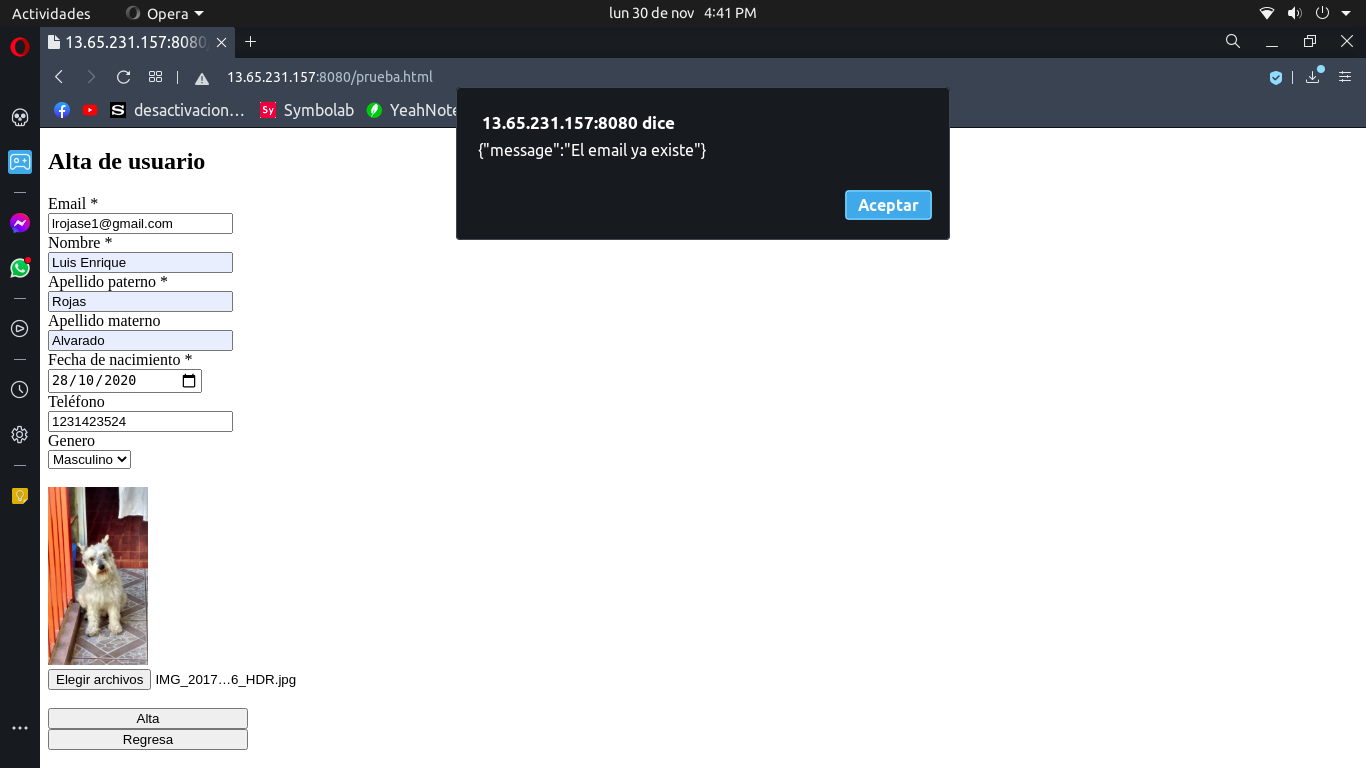
Ingresar la siguiente URL en un navegador:

http://*ip-de-la-máquina-virtual*:8080/[prueba.html](https://m4gm.com/moodle/mod/resource/view.php?id=1518)

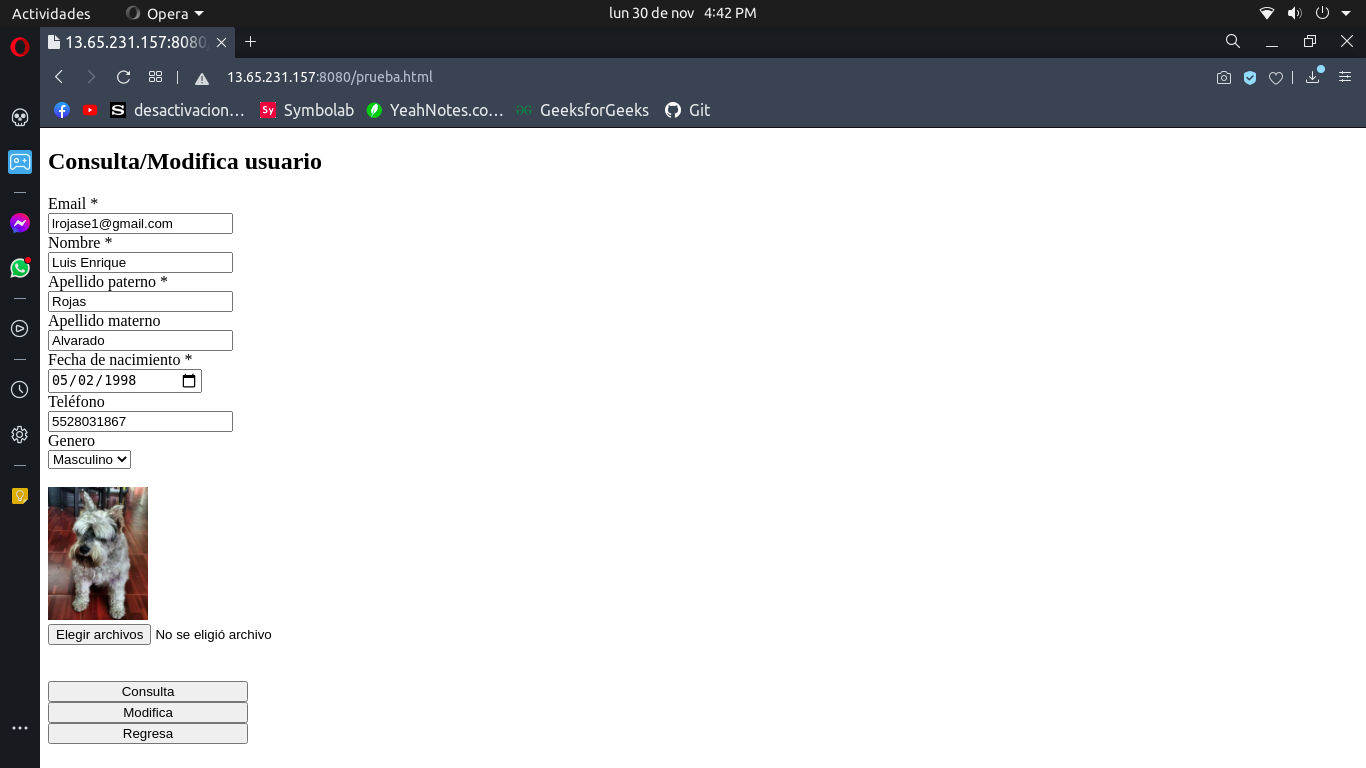
Dar clic en el botón “Alta usuario” para dar de alta un nuevo usuario. Capturar los campos y dar clic en el botón “Alta”.



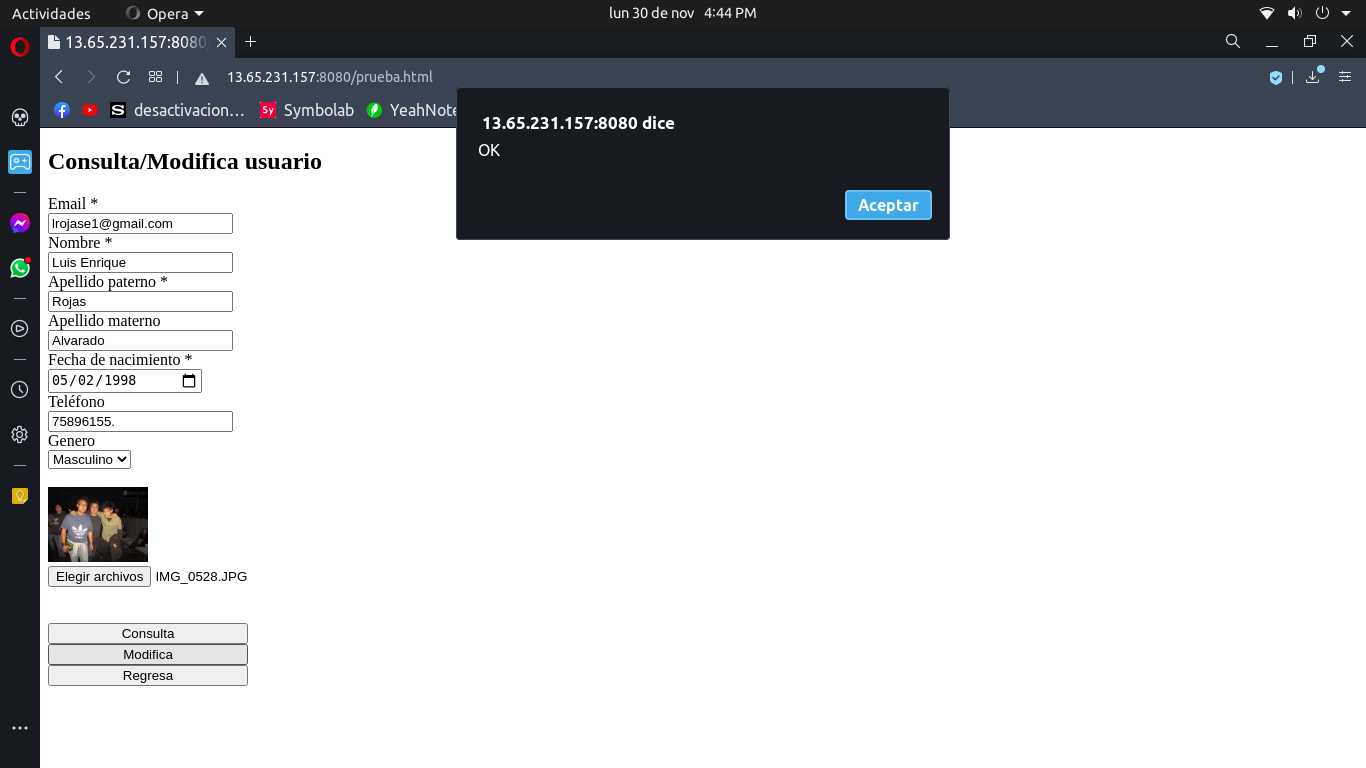
Intentar dar de alta otro usuario con el mismo email (se deberá mostrar una ventana de error indicando que el email ya existe).

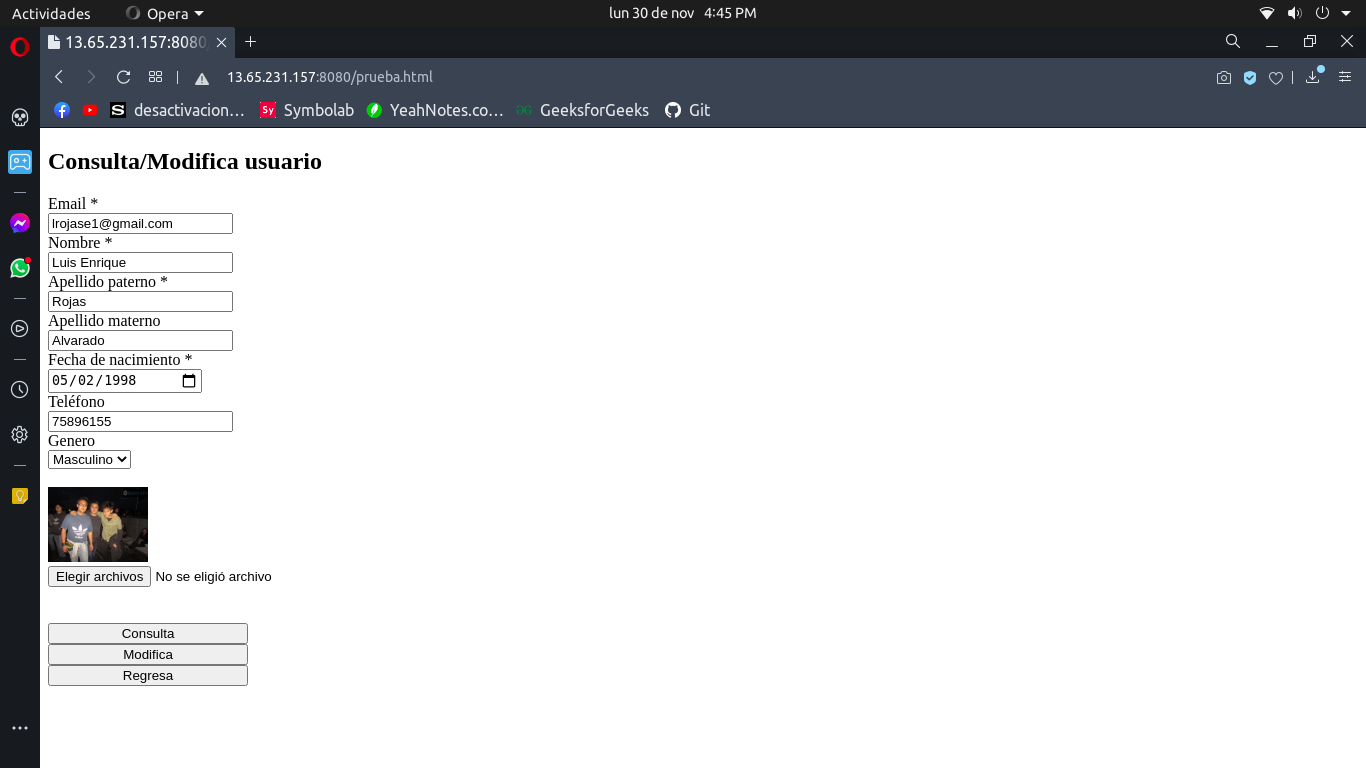


Dar clic en el botón “Consulta usuario” para consultar el usuario dado de alta en el paso anterior Capturar el email y dar clic en el botón “Consulta”,

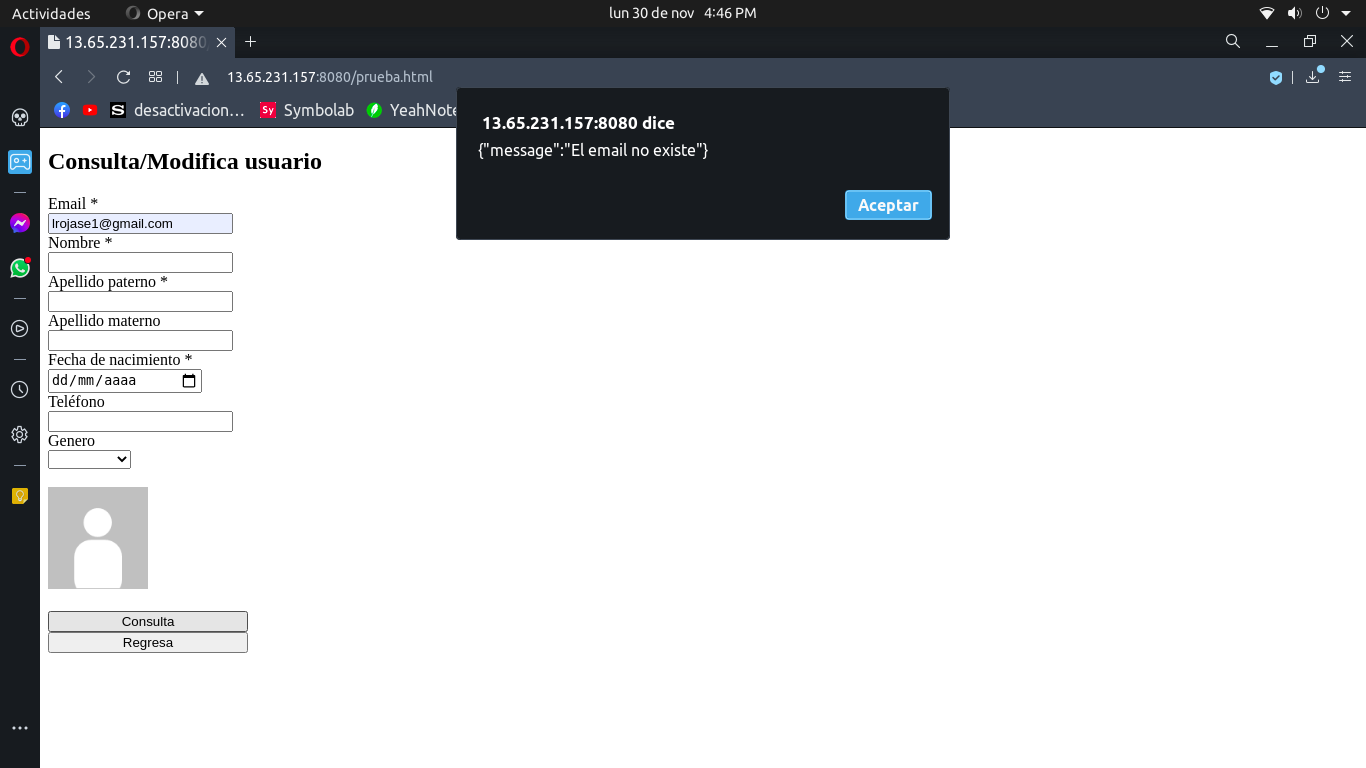
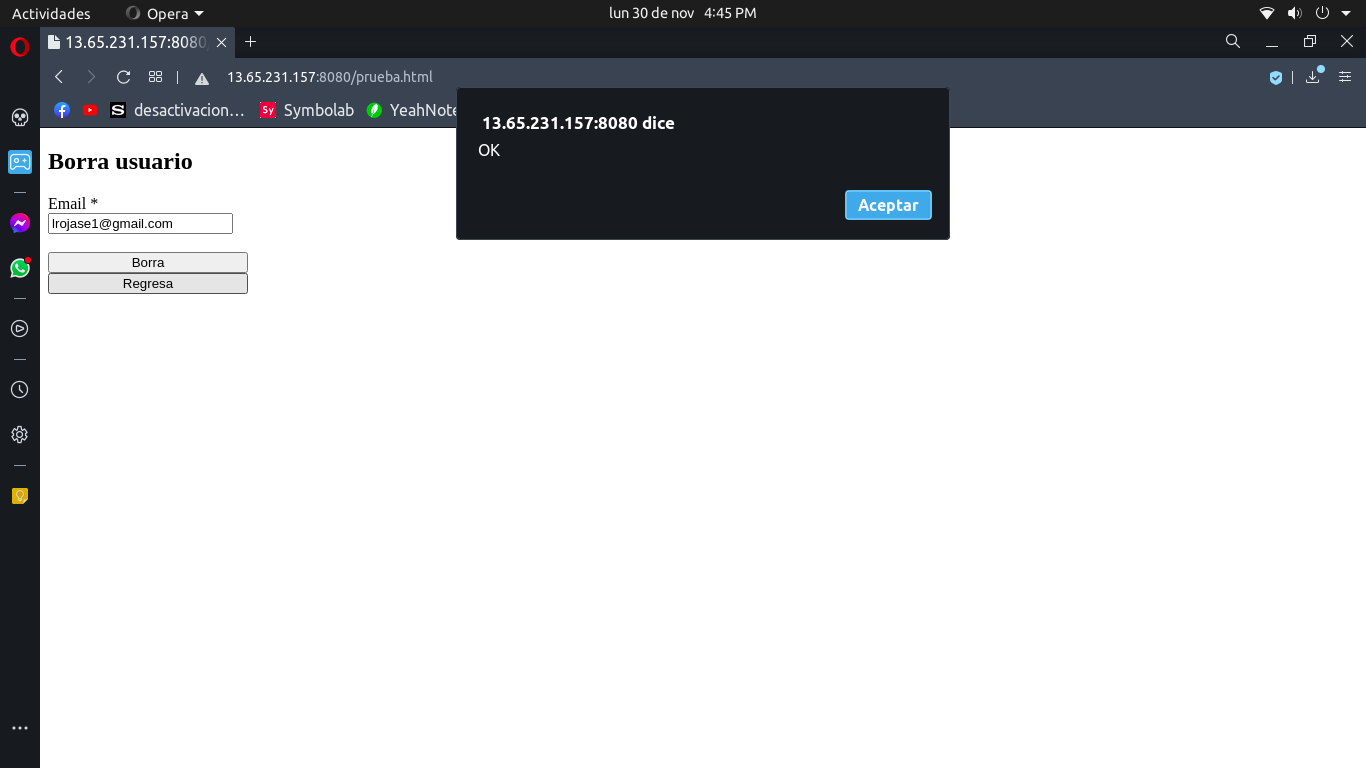


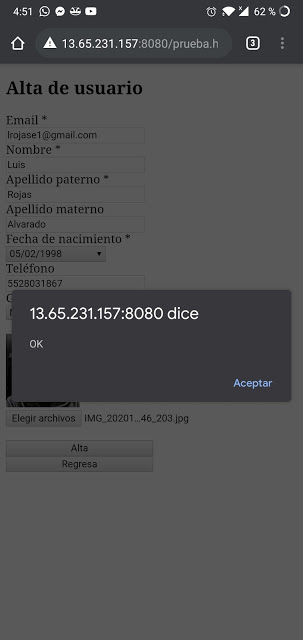
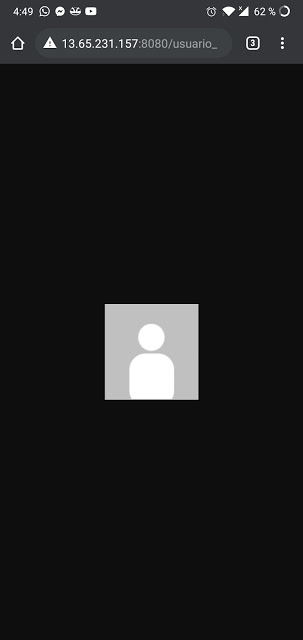
Modificar algún dato del usuario y dar clic en el botón “Modifica”:

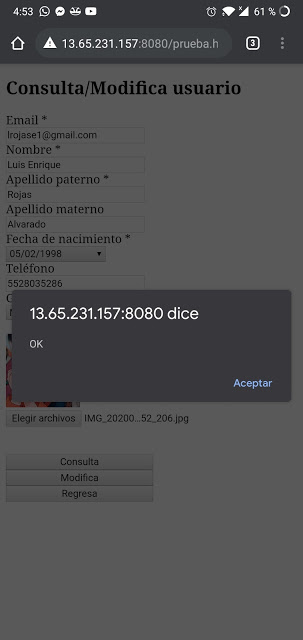
Recargar la página actual y consultar el usuario modificado, para verificar que la modificación se realizó.

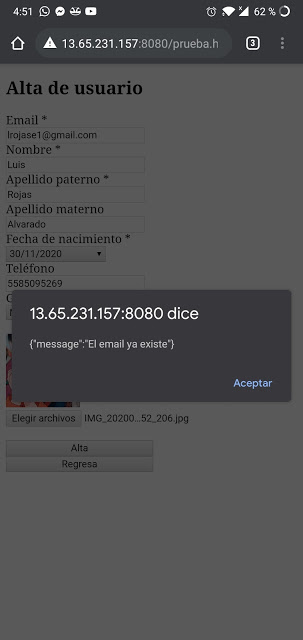


Dar clic en el botón “Borra usuario” para borrar el usuario. Capturar el email del usuario a borrar y dar clic en el botón “Consulta”.

Haciéndo las mismas pruebas desde un teléfono móvil.

****

****

****

**CONCLUSIONES.**

En esta práctica pudimos observar claramente lo que se trata una aplicación REST, y su comportamiento básico. El desplegar la aplicación en una máquina virtual y a su vez montarlo en un servidor web Tomcat, nos hace apreciar el sistema distribuido como su fuese una sola aplicación corriendo en una consola de Linux. Se aprecia correctamente el uso de un back-end por parte de la aplicación, haciendo uso de MySQL como manejador de base de datos y compilando el programa con JDBC para que pudiera tener la relación que esperábamos al conectar la base de datos. Y la interacción que tiene con el front-end por parte de lo que muestra la aplicación al usuario final y su interacción a su vez con este.