



INSTITUTO POLITÉCNICO NACIONAL  
ESCUELA SUPERIOR DE CÓMPUTO

*”Tarea 7. Implementación de un servicio web  
estilo REST”*

Alumno:

Lara Cázares Jaime Arturo

Materia:

Desarrollo de sistemas distribuidos

Grupo:

4CM3

Profesor: Pineda Guerrero Carlos

# Índice general

1.	Desarrollo . . . . .	3
1.1.	Creación de maquina virtual en Azure . . . . .	4
1.2.	Acceso a la máquina virtual . . . . .	5
1.3.	Preparación de la máquina virtual y del servicio . . . . .	6
2.	Prueba del servicio . . . . .	11
2.1.	Pruebas en dispositivo con sistema operativo Windows . . . . .	12
2.1.1.	Prueba de alta de usuario. . . . .	13
2.1.2.	Prueba de consulta de usuario. . . . .	14
2.1.3.	Prueba de modificación de usuario. . . . .	14
2.1.4.	Prueba de borrado de usuario. . . . .	15
2.2.	Pruebas en dispositivo con sistema operativo Android . . . . .	16
2.2.1.	Prueba de alta de usuario. . . . .	16
2.2.2.	Prueba de consulta de usuario. . . . .	17
2.2.3.	Prueba de borrado de usuario. . . . .	18
3.	Conclusiones . . . . .	20

## 1. Desarrollo

Para desarrollar esta tarea se crea un script de bash para hacer una instalación rápida y eficaz en la máquina virtual, este script es el siguiente:

```
1 #####Instalación de Tomcat con soporte REST
2 sudo apt update
3 sudo apt install openjdk-8-jdk-headless
4 sudo apt-get install -y unzip
5 unzip apache-tomcat-8.5.60.zip
6 rm -rf apache-tomcat-8.5.60/webapps
7 mkdir apache-tomcat-8.5.60/webapps
8 mkdir apache-tomcat-8.5.60/webapps/ROOT
9 unzip jaxrs-ri-2.24.zip
10 cp jaxrs-ri/api/*.jar apache-tomcat-8.5.60/lib/
11 cp jaxrs-ri/ext/*.jar apache-tomcat-8.5.60/lib/
12 cp jaxrs-ri/lib/*.jar apache-tomcat-8.5.60/lib/
13 rm apache-tomcat-8.5.60/lib/javax.servlet-api-3.0.1.jar
14 cp gson-2.3.1.jar apache-tomcat-8.5.60/lib/
15 unzip mysql-connector-java-8.0.22.zip
16 cp mysql-connector-java-8.0.22/mysql-connector-java-8.0.22.jar apache-tomcat-8.5.60/lib/
17 export CATALINA_HOME=$(pwd)/apache-tomcat-8.5.60
18 export JAVA_HOME=/usr/lib/jvm/java-8-openjdk-amd64
19 sh $CATALINA_HOME/bin/catalina.sh start
20
21 #####Instalación de MySQL
22 INS_MYSQL="ALTER USER 'root'@'localhost' IDENTIFIED WITH mysql_native_password BY 'Once*once';"
23 FLUSH PRIVILEGES;"
24 sudo apt update
25 sudo apt install mysql-server
26 sudo mysql_secure_installation
27 mysql -u root -p -e "$INS_MYSQL"
28
29
30 #####Crear un usuario en MySQL
31 CREAR_USUARIO="create user hugo@localhost identified by 'Once*once121';"
32 grant all on servicio_web.* to hugo@localhost;"
33 mysql -u root -p -e "$CREAR_USUARIO"
34
35 #####Crear la base de datos
36 CREAR_BD="create database if not exists servicio_web;"
37 use servicio_web;"
38 create table if not exists usuarios
39 (
40     id_usuario integer auto_increment primary key,
41     email varchar(256) not null,
```

```

42     nombre varchar(100) not null,
43     apellido_paterno varchar(100) not null,
44     apellido_materno varchar(100),
45     fecha_nacimiento date not null,
46     telefono varchar(20),
47     genero char(1)
48 );
49 create table if not exists fotos_usuarios
50 (
51     id_foto integer auto_increment primary key,
52     foto longblob,
53     id_usuario integer not null
54 );
55 alter table fotos_usuarios add foreign key (id_usuario) references usuarios(id_usuario);
56 create unique index usuarios_1 on usuarios(email);"
57
58 mysql -u hugo -p -e "$CREAR_BD"
59
60 #####Compilar, empacar y desplegar el servicio web
61 rm -r META-INF
62 rm -r WEB-INF
63 rm -r negocio
64 unzip Servicio.zip
65 chmod -R 777 META-INF
66 chmod -R 777 WEB-INF
67 chmod -R 777 negocio
68 cp context.xml META-INF/
69 export CATALINA_HOME=$(pwd)/apache-tomcat-8.5.60
70 javac -cp $CATALINA_HOME/lib/javax.ws.rs-api-2.0.1.jar:$CATALINA_HOME/lib/gson-2.3.1.jar:.
71 rm WEB-INF/classes/negocio/*
72 cp negocio/*.class WEB-INF/classes/negocio/.
73 jar cvf Servicio.war WEB-INF META-INF
74 cp Servicio.war apache-tomcat-8.5.60/webapps/
75 cp usuario_sin_foto.png apache-tomcat-8.5.60/webapps/ROOT/
76 cp WSCliient.js apache-tomcat-8.5.60/webapps/ROOT/
77 cp prueba.html apache-tomcat-8.5.60/webapps/ROOT/

```

### 1.1. Creación de maquina virtual en Azure

En esta tarea solo es necesario crear una sola máquina virtual en la Figura 1 se aprecia el portal Azure con la máquina virtual lista. Se configura el puerto de 8080 con el protocolo TCP para poder acceder al servicio, ver la Figura 2.

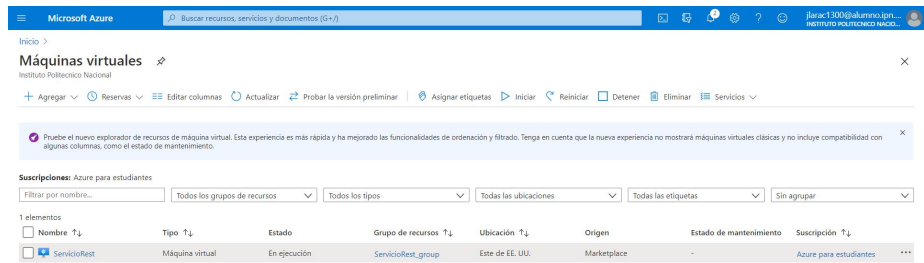


Figura 1: Máquina virtual en Azure.

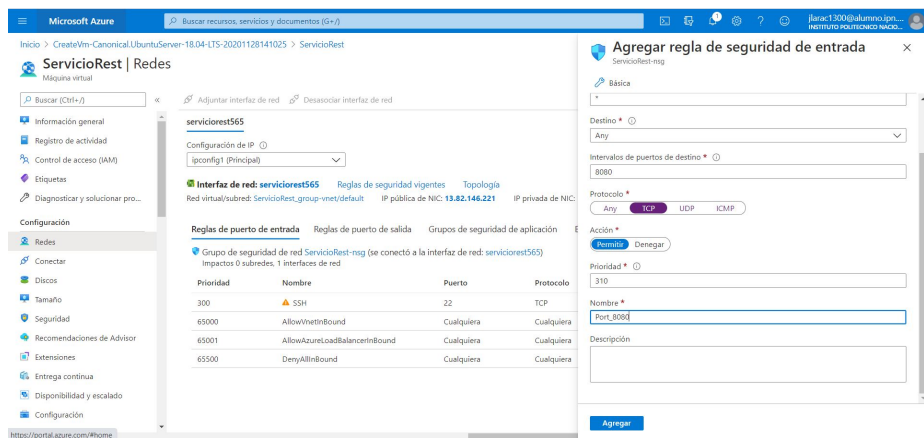


Figura 2: Configuración puerto 8080 para conectarnos al servicio.

## 1.2. Acceso a la máquina virtual

Se accesa con ayuda de **Putty** a la maquina virtual, en la Figura 3 muestra el acceso a la maquina virtual.

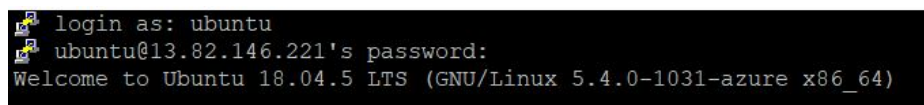


Figura 3: Acceso a la maquina virtual.

### 1.3. Preparación de la máquina virtual y del servicio

Una vez teniendo la maquina virtual creada se procede a clonar un repositorio de github (ver la Figura 4) en el que previamente se pusieron los paquetes necesarios, estos son:

- apache-tomcat-8.5.60.zip: en el se contiene la distribución binaria de Tomcat 8. Apache Tomcat es un contenedor Java Servlet, o contenedor web, que proporciona la funcionalidad extendida para interactuar con Java Servlets, al tiempo que implementa varias especificaciones técnicas de la plataforma Java: JavaServer Pages (JSP), Java Expression Language (Java EL) y WebSocket.
- gson-2.3.1.jar: implementación de JSON desarrollada por Google, es una librería de Java utilizada comunmente para convertir un objeto a una cadena en formato JSON.
- jaxrs-ri-2.24.zip: biblioteca "Jersey", es una implementación de JAX-RS lo cual permite ejecutar servicios web estilo REST sobre Tomcat.
- mysql-connector-java-8.0.22.zip: driver de JDBC para MySQL.
- WSClient.js: es un documento que contiene la descripción de un servicio web. Este especifica la localización del servicio y los métodos del servicio.
- Servicio.zip: contiene el código Java y archivos de configuración de un servicio web estilo REST.
- Prueba.html: contiene una aplicación web que invoca el servicio web mediante Javascript.
- usuario-sin-foto.png: imagen destinada para hacer pruebas.
- context.xml: a diferencia del encontrado den "*Servicion.zip*" en este ya se encuentran establecidos los datos para la conexión a la base de datos.
- install.sh: script bash que contiene todos los comandos necesarios para la instalación y creación del servicio web.

```
ubuntu@ServicioRest:~$ git clone https://github.com/artLara/Desarrollo-de-sistemas-distribuidos.git
Cloning into 'Desarrollo-de-sistemas-distribuidos'...
remote: Enumerating objects: 1133, done.
remote: Counting objects: 100% (1133/1133), done.
remote: Compressing objects: 100% (651/651), done.
remote: Total 1133 (delta 451), reused 1106 (delta 427), pack-reused 0
Receiving objects: 100% (1133/1133), 44.46 MiB | 43.32 MiB/s, done.
Resolving deltas: 100% (451/451), done.
ubuntu@ServicioRest:~$
```

Figura 4: Obtención de los archivos necesarios clonando el repositorio de github.

Se utiliza el siguiente comando para llevar acabo la clonación:

```
git clone https://github.com/artLara/Desarrollo-de-sistemas-distribuidos.git
```

Posteriormente se corre el script *"install"* para la instalación de todo lo necesario. En la Figura 5 se aprecia la ejecución del script de instalación (al ser muy largo las impresiones de pantalla solo se toma captura a las principales), en la Figura 6 vemos el desempaquetado de *jaxrs-ri-2.24.zip* y *mysql-connector-java-8.0.22.zip*.

```
root@ServicioRest:/home/ubuntu/Desarrollo-de-sistemas-distribuidos/Tarea7# sh install.sh
Hit:1 http://azure.archive.ubuntu.com/ubuntu bionic InRelease
Get:2 http://azure.archive.ubuntu.com/ubuntu bionic-updates InRelease [88.7 kB]
Get:3 http://azure.archive.ubuntu.com/ubuntu bionic-backports InRelease [74.6 kB]
Hit:4 http://security.ubuntu.com/ubuntu bionic-security InRelease
Fetched 163 kB in 0s (401 kB/s)
Reading package lists... Done
Building dependency tree
Reading state information... Done
2 packages can be upgraded. Run 'apt list --upgradable' to see them.
Reading package lists... Done
Building dependency tree
Reading state information... Done
openjdk-8-jdk-headless is already the newest version (8u275-b01-0ubuntu1~18.04).
The following package was automatically installed and is no longer required:
  linux-headers-4.15.0-124
Use 'sudo apt autoremove' to remove it.
0 upgraded, 0 newly installed, 0 to remove and 2 not upgraded.
Archive: apache-tomcat-8.5.60.zip
creating: apache-tomcat-8.5.60/
creating: apache-tomcat-8.5.60/bin/
creating: apache-tomcat-8.5.60/conf/
creating: apache-tomcat-8.5.60/lib/
creating: apache-tomcat-8.5.60/logs/
creating: apache-tomcat-8.5.60/temp/
creating: apache-tomcat-8.5.60/webapps/
creating: apache-tomcat-8.5.60/webapps/ROOT/
creating: apache-tomcat-8.5.60/webapps/ROOT/WEB-INF/
creating: apache-tomcat-8.5.60/webapps/docs/
creating: apache-tomcat-8.5.60/webapps/docs/WEB-INF/
creating: apache-tomcat-8.5.60/webapps/docs/annotationapi/
creating: apache-tomcat-8.5.60/webapps/docs/api/
creating: apache-tomcat-8.5.60/webapps/docs/appdev/
creating: apache-tomcat-8.5.60/webapps/docs/appdev/sample/
creating: apache-tomcat-8.5.60/webapps/docs/appdev/sample/docs/
creating: apache-tomcat-8.5.60/webapps/docs/appdev/sample/src/
creating: apache-tomcat-8.5.60/webapps/docs/appdev/sample/src/mypackage/
creating: apache-tomcat-8.5.60/webapps/docs/appdev/sample/web/
creating: apache-tomcat-8.5.60/webapps/docs/appdev/sample/web/WEB-INF/
creating: apache-tomcat-8.5.60/webapps/docs/appdev/sample/web/images/
creating: apache-tomcat-8.5.60/webapps/docs/architecture/
creating: apache-tomcat-8.5.60/webapps/docs/architecture/requestProcess/
creating: apache-tomcat-8.5.60/webapps/docs/architecture/startup/
creating: apache-tomcat-8.5.60/webapps/docs/config/
creating: apache-tomcat-8.5.60/webapps/docs/elapi/
creating: apache-tomcat-8.5.60/webapps/docs/images/
creating: apache-tomcat-8.5.60/webapps/docs/images/fonts/
creating: apache-tomcat-8.5.60/webapps/docs/jspicapi/
creating: apache-tomcat-8.5.60/webapps/docs/jspapi/
```

Figura 5: Ejecución del script de instalación.



```

Archive:  jaxrs-ri-2.24.zip
  creating:  jaxrs-ri/
  inflating:  jaxrs-ri/Jersey-LICENSE.txt
  inflating:  jaxrs-ri/third-party-license-readme.txt
  creating:  jaxrs-ri/api/
  extracting:  jaxrs-ri/api/javax.ws.rs-api-2.0.1.jar
  creating:  jaxrs-ri/lib/
  extracting:  jaxrs-ri/lib/jersey-common.jar
  extracting:  jaxrs-ri/lib/jersey-media-jaxb.jar
  extracting:  jaxrs-ri/lib/jersey-client.jar
  extracting:  jaxrs-ri/lib/jersey-server.jar
  extracting:  jaxrs-ri/lib/jersey-container-servlet-core.jar
  extracting:  jaxrs-ri/lib/jersey-container-servlet.jar
  creating:  jaxrs-ri/ext/
  extracting:  jaxrs-ri/ext/javax.inject-2.5.0-b05.jar
  extracting:  jaxrs-ri/ext/osgi-resource-locator-1.0.1.jar
  extracting:  jaxrs-ri/ext/javax.annotation-api-1.2.jar
  extracting:  jaxrs-ri/ext/jersey-guava-2.24.jar
  extracting:  jaxrs-ri/ext/hk2-api-2.5.0-b05.jar
  extracting:  jaxrs-ri/ext/hk2-utils-2.5.0-b05.jar
  extracting:  jaxrs-ri/ext/aopalliance-repackaged-2.5.0-b05.jar
  extracting:  jaxrs-ri/ext/hk2-locator-2.5.0-b05.jar
  extracting:  jaxrs-ri/ext/javassist-3.20.0-GA.jar
  extracting:  jaxrs-ri/ext/validation-api-1.1.0.Final.jar
  extracting:  jaxrs-ri/ext/org.osgi.core-4.2.0.jar
  extracting:  jaxrs-ri/ext/jaxb-api-2.2.7.jar
  extracting:  jaxrs-ri/ext/javax.servlet-api-3.0.1.jar
  extracting:  jaxrs-ri/ext/persistence-api-1.0.jar
Archive:  mysql-connector-java-8.0.22.zip
  creating:  mysql-connector-java-8.0.22/
  creating:  mysql-connector-java-8.0.22/src/
  creating:  mysql-connector-java-8.0.22/src/build/
  creating:  mysql-connector-java-8.0.22/src/build/java/
  creating:  mysql-connector-java-8.0.22/src/build/java/documentation/
  creating:  mysql-connector-java-8.0.22/src/build/java/instrumentation/
  creating:  mysql-connector-java-8.0.22/src/build/misc/
  creating:  mysql-connector-java-8.0.22/src/build/misc/debian.in/
  creating:  mysql-connector-java-8.0.22/src/build/misc/debian.in/source/
  creating:  mysql-connector-java-8.0.22/src/demo/
  creating:  mysql-connector-java-8.0.22/src/demo/java/
  creating:  mysql-connector-java-8.0.22/src/demo/java/demo/
  creating:  mysql-connector-java-8.0.22/src/demo/java/demo/x/
  creating:  mysql-connector-java-8.0.22/src/demo/java/demo/x/devapi/
  creating:  mysql-connector-java-8.0.22/src/generated/
  creating:  mysql-connector-java-8.0.22/src/generated/java/
  creating:  mysql-connector-java-8.0.22/src/generated/java/com/
  creating:  mysql-connector-java-8.0.22/src/generated/java/com/mysql/
  creating:  mysql-connector-java-8.0.22/src/generated/java/com/mysql/cj/
  creating:  mysql-connector-java-8.0.22/src/generated/java/com/mysql/cj/x/

```

Figura 6: Desempaquetado de jaxrs-ri-2.24.zip y mysql-connector-java-8.0.22.zip.

En la Figura 7 se observa que Tomcat se levanta correctamente, mientras que en la Figura 8 la instalación de MySQL y en la Figura 9 la descompresión de "Servicio.zip" referente a la sección compilar, empacar y desplegar el servicio web.



```

Setting CATALINA_BASE: /home/ubuntu/Desarrollo-de-sistemas-distribuidos/Tarea7/apache-tomcat-8.5.60
Setting CATALINA_HOME: /home/ubuntu/Desarrollo-de-sistemas-distribuidos/Tarea7/apache-tomcat-8.5.60
Setting CATALINA_TMPDIR: /home/ubuntu/Desarrollo-de-sistemas-distribuidos/Tarea7/apache-tomcat-8.5.60/temp
Setting JRE_HOME: /usr/lib/jvm/java-7-openjdk-amd64
Setting CLASSPATH: /home/ubuntu/Desarrollo-de-sistemas-distribuidos/Tarea7/apache-tomcat-8.5.60/bin/bootstrap.jar:/home/ubuntu/Desarrollo-de-sistemas-distribuidos/Tarea7/apache-tomcat-8.5.60/bin/tomcat-juli.jar
Setting CATALINA_OPTS:
tomcat started.

```

Figura 7: Inicia servicio Tomcat.

```

Securing the MySQL server deployment.

Connecting to MySQL using a blank password.

VALIDATE PASSWORD PLUGIN can be used to test passwords
and improve security. It checks the strength of password
and allows the users to set only those passwords which are
secure enough. Would you like to setup VALIDATE PASSWORD plugin?

Press y|Y for Yes, any other key for No: n
Please set the password for root here.

New password:

Re-enter new password:
By default, a MySQL installation has an anonymous user,
allowing anyone to log into MySQL without having to have
a user account created for them. This is intended only for
testing, and to make the installation go a bit smoother.
You should remove them before moving into a production
environment.

Remove anonymous users? (Press y|Y for Yes, any other key for No) : y
Success.

Normally, root should only be allowed to connect from
'localhost'. This ensures that someone cannot guess at
the root password from the network.

Disallow root login remotely? (Press y|Y for Yes, any other key for No) : y
Success.

By default, MySQL comes with a database named 'test' that
anyone can access. This is also intended only for testing,
and should be removed before moving into a production
environment.

Remove test database and access to it? (Press y|Y for Yes, any other key for No) : y
- Dropping test database...
Success.

- Removing privileges on test database...
Success.

Reloading the privilege tables will ensure that all changes
made so far will take effect immediately.

Reload privilege tables now? (Press y|Y for Yes, any other key for No) : y

```

Figura 8: Instalación de MySQL.

```

Archive: Servicio.zip
  creating: META-INF/
  inflating: META-INF/context.xml
  creating: WEB-INF/
  inflating: WEB-INF/web.xml
  creating: WEB-INF/classes/
  creating: WEB-INF/classes/negocio/
  creating: negocio/
  inflating: negocio/Usuario.java
  inflating: negocio/Foto.java
  inflating: negocio/AdaptadorGsonBase64.java
  inflating: negocio/Error.java
  inflating: negocio/Servicio.java
rm: cannot remove 'WEB-INF/classes/negocio/*': No such file or directory
added manifest
adding: WEB-INF/ (in = 0) (out= 0) (stored 0%)
adding: WEB-INF/classes/ (in = 0) (out= 0) (stored 0%)
adding: WEB-INF/classes/negocio/ (in = 0) (out= 0) (stored 0%)
adding: WEB-INF/classes/negocio/AdaptadorGsonBase64.class (in = 1799) (out= 737) (deflated 59%)
adding: WEB-INF/classes/negocio/Error.class (in = 278) (out= 214) (deflated 23%)
adding: WEB-INF/classes/negocio/Servicio.class (in = 7578) (out= 3462) (deflated 54%)
adding: WEB-INF/classes/negocio/Usuario.class (in = 899) (out= 518) (deflated 42%)
adding: WEB-INF/web.xml (in = 672) (out= 296) (deflated 55%)
ignoring entry META-INF/
adding: META-INF/context.xml (in = 312) (out= 220) (deflated 29%)

```

Figura 9: Descompresión de "*Servicio.zip*" referente a la sección compilar, empaquetar y desplegar el servicio web.

Por último antes de iniciar a utilizar el servicio, se verifica la creación correcta de la base de datos (ver Figura 10) y que esta se encuentre vacía (ver Figura 11).

```

root@ServicioRest:/home/ubuntu/Desarrollo-de-sistemas-distribuidos/Tarea7# mysql -u hugo -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 51
Server version: 5.7.32-0ubuntu0.18.04.1 (Ubuntu)

Copyright (c) 2000, 2020, Oracle and/or its affiliates. All rights reserved.

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> show databases;
+-----+
| Database |
+-----+
| information_schema |
| servicio_web |
+-----+
2 rows in set (0.00 sec)

mysql> use servicio_web;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
mysql> show tables;
+-----+
| Tables_in_servicio_web |
+-----+
| fotos_usuarios |
| usuarios |
+-----+
2 rows in set (0.00 sec)

mysql>

```

Figura 10: Base de datos para el servicio web.

```

mysql> select * from usuarios;
Empty set (0.00 sec)

mysql>

```

Figura 11: Verificación de los datos dentro de la base de datos del servicio.

## 2. Prueba del servicio

Para probar que el servicio funcione se llevan acabo pruebas como alta, consulta y borrado de usuarios tanto en un navegador web en un sistema operativo windows y android.

## 2.1. Pruebas en dispositivo con sistema operativo Windows

Se comienza probando la conexión a la maquina virtual y al servicio, para esto se solicita el recurso *"usuario\_sin\_foto.png"* (ver Figura 12) y *"prueba.html"* (ver Figura 13).

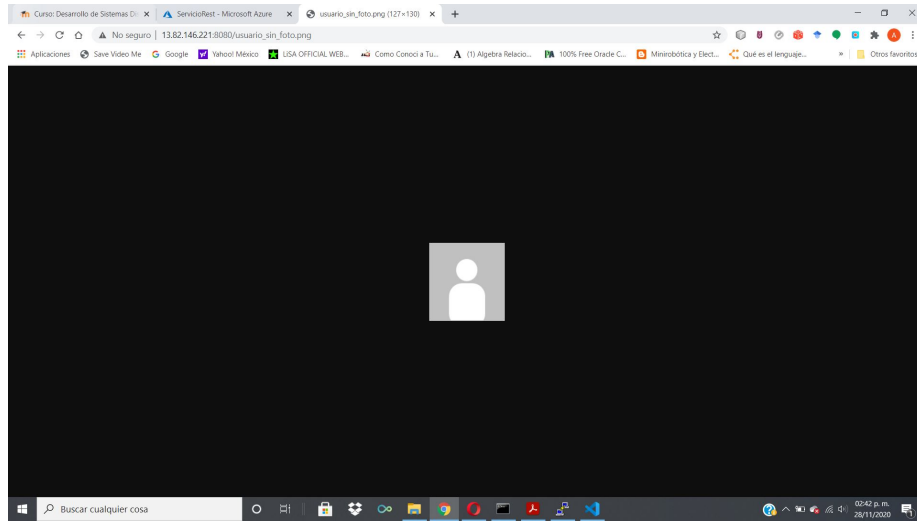


Figura 12: Verificación de conexión.

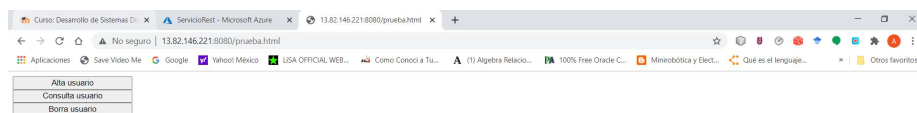


Figura 13: Acceso al servicio.

### 2.1.1. Prueba de alta de usuario.

En la Figura 14 se prueba dar de alta a un usuario nuevo, la operación es exitosa y regresar un "OK".

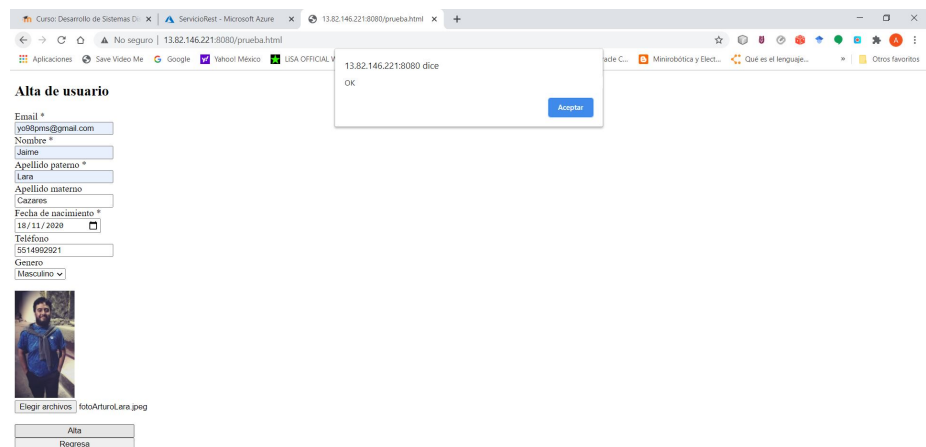


Figura 14: Alta exitosa de un usuario nuevo.

En la Figura 15 se prueba dar de alta a un usuario con un email ya registrado, la operación es errónea y regresar un mensaje de error con la causa del mismo.

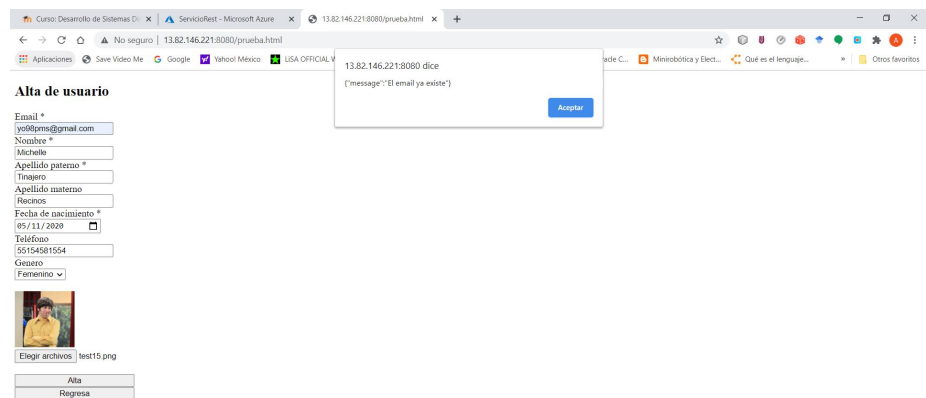
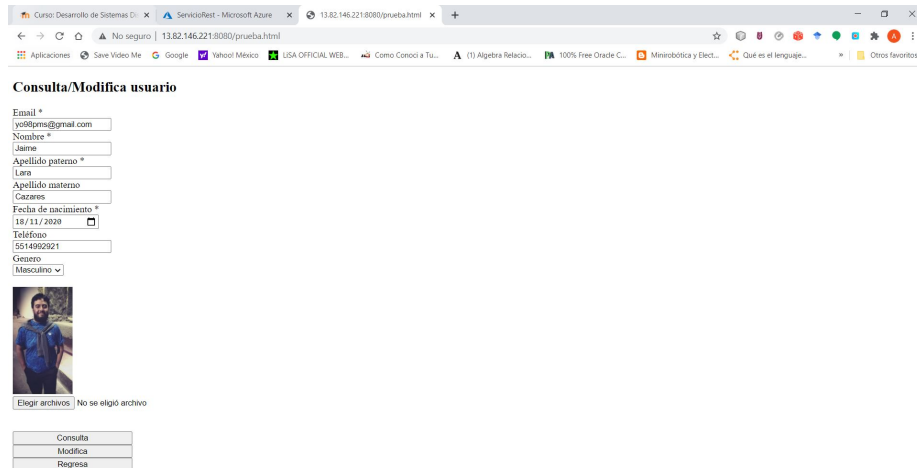


Figura 15: Alta errónea de usuario, el email ya existe.

### 2.1.2. Prueba de consulta de usuario.

En la Figura 16 se prueba consultar a un usuario existente, la operación es exitosa y se llenan os campos con sus datos.



Consulta/Modifica usuario

Email \*  
yot@pms@gmail.com

Nombre \*  
Jaime

Apellido paterno \*  
Lara

Apellido materno \*  
Cazares

Fecha de nacimiento \*  
18/11/2020

Teléfono  
5514992921

Género  
Masculino

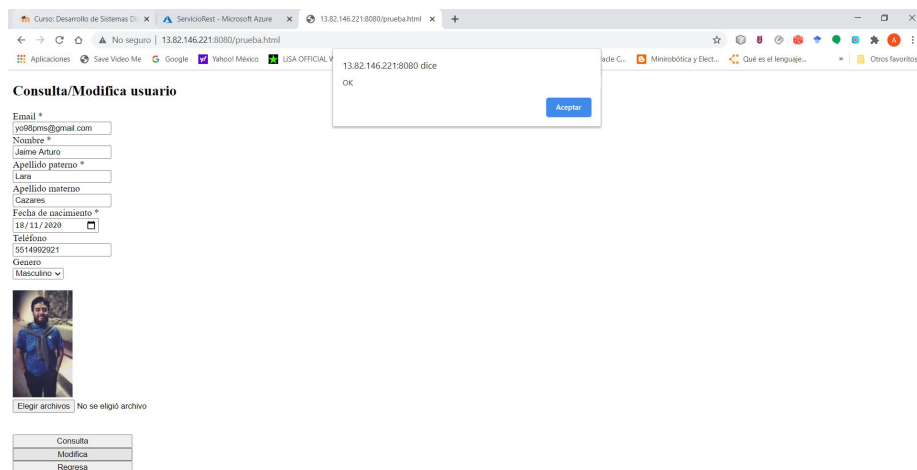
Elegir archivos | No se eligió archivo

Consulta  
Modifica  
Regresa

Figura 16: Consulta exitosa de un usuario existente.

### 2.1.3. Prueba de modificación de usuario.

En la Figura 17 se prueba modificar a un usuario, la operación es exitosa y se regresa "OK".



Consulta/Modifica usuario

Email \*  
yot@pms@gmail.com

Nombre \*  
Jaime Arturo

Apellido paterno \*  
Lara

Apellido materno \*  
Cazares

Fecha de nacimiento \*  
18/11/2020

Teléfono  
5514992921

Género  
Masculino

Elegir archivos | No se eligió archivo

Consulta  
Modifica  
Regresa

13.82.146.221:8080 dice  
OK  
Aceptar

Figura 17: Modificación del nombre del usuario.

#### 2.1.4. Prueba de borrado de usuario.

En la Figura 18 se prueba borrar a un usuario, la operación es exitosa y se regresa "OK".

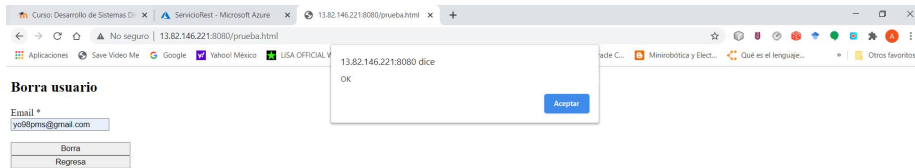


Figura 18: Borrado de un usuario.

En la Figura 19 se prueba consultar al usuario borrado, la operación es errónea y regresar un mensaje de error con la causa del mismo.

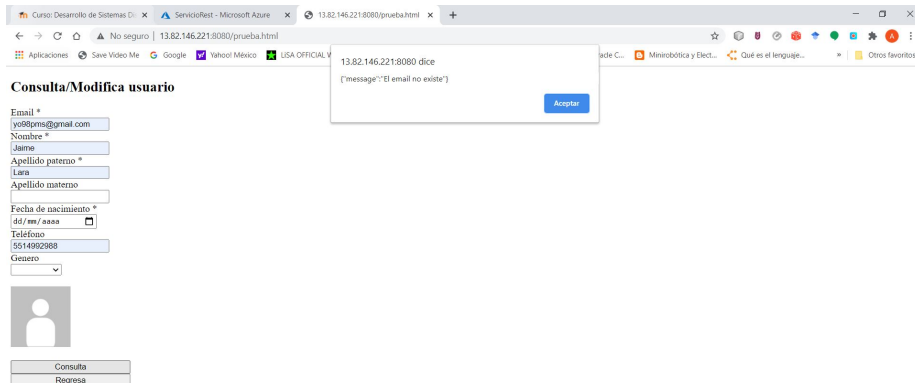


Figura 19: Consulta del usuario borrado.



## 2.2. Pruebas en dispositivo con sistema operativo Android

### 2.2.1. Prueba de alta de usuario.

En la Figura 20 se prueba dar de alta a un usuario nuevo, la operación es exitosa y regresar un "OK".

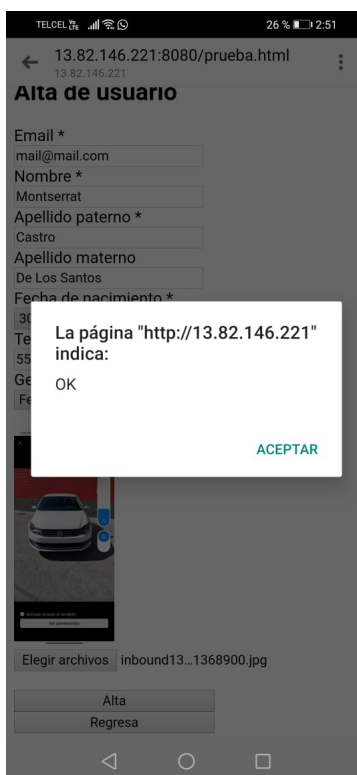


Figura 20: Alta exitosa de un usuario nuevo en Android.

En la Figura 21 se prueba dar de alta a un usuario con un email ya registrado, la operación es errónea y regresar un mensaje de error con la causa del mismo.



Figura 21: Alta errónea de usuario en Android, el email ya existe.

### 2.2.2. Prueba de consulta de usuario.

En la Figura 22 se prueba consultar a un usuario existente, la operación es exitosa y se llenan los campos con sus datos.

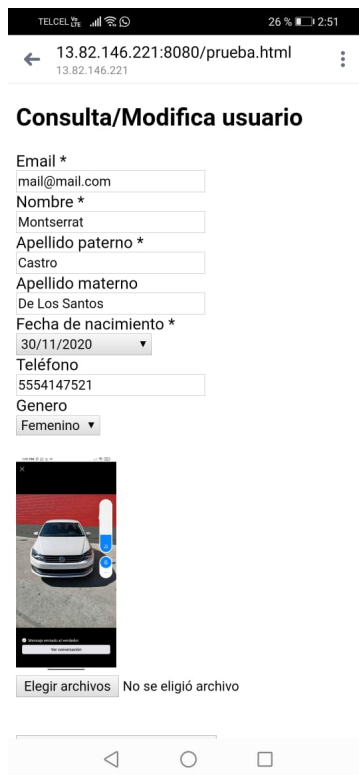


Figura 22: Consulta exitosa de un usuario existente en Andriod.

### 2.2.3. Prueba de borrado de usuario.

En la Figura 23 se prueba borrar a un usuario, la operación es exitosa y se regresa "OK".



Figura 23: Borrado de un usuario en Android.

En la Figura 24 se prueba consultar al usuario borrado, la operación es errónea y regresar un mensaje de error con la causa del mismo.

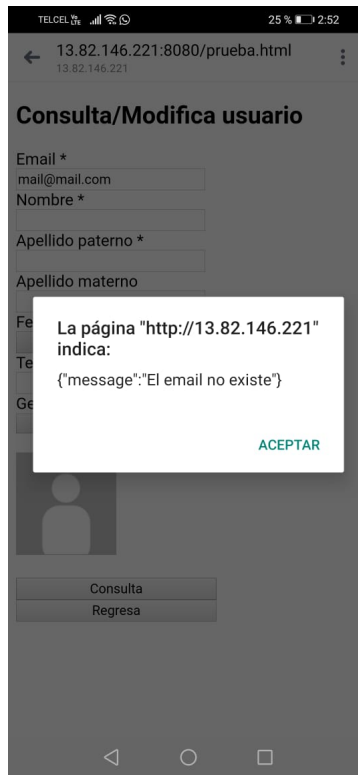


Figura 24: Consulta del usuario borrado en Android.

### 3. Conclusiones

La utilización de los servicios REST proporcionan mucha flexibilidad a la hora de implementar un sistema con arquitectura cliente-servidor pues es muy fácil separar las dos partes para ofrecer un servicio. Dado a que se manejan peticiones HTTP se tiene la ventaja de poder desarrollar en distintas plataformas aplicaciones que consuman el servicio, de esta forma se puede hacer uso desde un dispositivo móvil, lapton, PC o cualquier dispositivo que pueda acceder al servicio.

Se observa que también se permite una implementación más limpia de código, lo cual simplifica el mantenimiento del mismo e incluso es posible su implementación en otros lenguajes utilizando el protocolo de comunicación HTTP para consumir el servicio.