

INSTITUTO POLITÉCNICO NACIONAL

ESCUELA SUPERIOR DE CÓMPUTO

Sistemas Distribuidos

Prof. Carlos Pineda Guerrero

Alejandro de Jesús Zepeda Flores

Tarea 8. Desarrollo de un cliente para un servicio web REST

5 de diciembre de 2020

Requisitos

Desarrollar un programa Java consola que consuma el servicio web que creamos en la tarea 7. El programa deberá desplegar el siguiente menú: Alta usuario, Consulta usuario, Borra usuario, Borra todos los usuarios y Salir.

La opción "Alta usuario" leerá del teclado el email, el nombre del usuario, el apellido paterno, el apellido materno, la fecha de nacimiento, el teléfono y el género ("M" o "F"). Entonces se invocará el método "alta" del servicio web. Se deberá desplegar "OK" si se pudo dar de alta el usuario, o bien, el mensaje de error que regresa el servicio web.

La opción "Consulta usuario" leerá del teclado el email de un usuario previamente dado de alta. Entonces se invocará el método "consulta" del servicio web. Si el usuario existe se desplegará en pantalla el nombre del usuario, el apellido paterno, el apellido materno, la fecha de nacimiento, el teléfono y el género. La foto del usuario se ignorará. Notar que el método web "consulta" regresa una string JSON la cual representa un objeto de tipo Usuario (ver el archivo Servicio.java), por tanto será necesario utilizar **JSON** para convertir la string JSON a un objeto de tipo Usuario y posteriormente desplegar los campos del objeto (excepto el campo "foto"). Si hubo error, se desplegará el mensaje que regresa el servicio web.

La opción "Borra usuario" leerá del teclado el email de un usuarios previamente dado de alta. Entonces se invocará el método "borra" del servicio web. Se deberá desplegar "OK" si se pudo borrar el usuario, o bien, el mensaje de error que regresa el servicio web.

La opción "Borra todos los usuarios" invocará el método "borrar_usuarios" creado en la actividad individual de la clase pasada. Se deberá desplegar "OK" si se pudo borrar el usuario, o bien, el mensaje de error que regresa el servicio web.

La opción "Salir" terminará el programa.

Desarrollo

A) Agregar el método “Eliminar todos los usuarios”

Ingresar a: <http://40.84.149.14:8080/prueba.html> para comprobar que realmente el servicio web se encuentra disponible.

La práctica pasada terminó en esta sección, por lo que ahora procedemos a crear el nuevo botón y agregar el nuevo método que nos permita eliminar todos los usuarios.

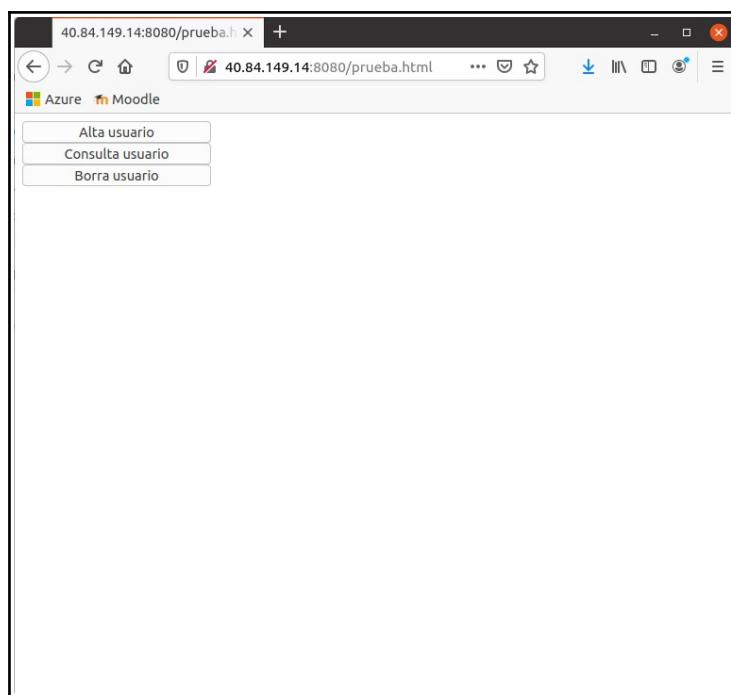


Figura 1. Servicio web disponible.

Primero, debemos eliminar el Servicio.war y después, el directorio Servicio. Debe ser en ese orden para no tener complicaciones a la hora de subir el nuevo servicio.

```
adejesuszf@servicioweb:~/apache-tomcat-8.5.60/webapps$ ls -lh
total 16K
drwxrwxr-x 2 adejesuszf adejesuszf 4.0K Dec  5 20:05 ROOT
drwxr-x--- 4 adejesuszf adejesuszf 4.0K Dec  5 20:03 Servicio
-rw-rw-r-- 1 adejesuszf adejesuszf 6.9K Dec  5 20:03 Servicio.war
adejesuszf@servicioweb:~/apache-tomcat-8.5.60/webapps$ rm Servicio.war
adejesuszf@servicioweb:~/apache-tomcat-8.5.60/webapps$ rm -r Servicio/
rm: cannot remove 'Servicio/': No such file or directory
```

The terminal window shows the user navigating to the "/webapps" directory of an Apache Tomcat 8.5.60 installation. They run the command "ls -lh" to list the contents, which include the "ROOT" directory, the "Servicio" directory, and the "Servicio.war" file. Then, they run "rm Servicio.war" to delete the war file. Finally, they attempt to delete the "Servicio" directory with "rm -r Servicio/" but receive an error message stating "rm: cannot remove 'Servicio/': No such file or directory" because the directory is empty.

Figura 2. Eliminación del Servicio.

Primero, al archivo Servicio.java se debe agregar un nuevo método para eliminar todos los usuarios. Indicamos que utilizará el método POST y PATH será borrar_usuarios.

```
@POST  
@Path("borrar_usuarios")  
@Consumes(MediaType.APPLICATION_FORM_URLENCODED)  
@Produces(MediaType.APPLICATION_JSON)  
public Response borrar_usuarios() throws Exception{  
    Connection conexion= pool.getConnection();  
    try{  
        PreparedStatement stmt_2 = conexion.prepareStatement("DELETE FROM fotos_usuarios");  
        try{  
            stmt_2.execute();  
        }finally{  
            stmt_2.close();  
        }  
        PreparedStatement stmt_1 = conexion.prepareStatement("DELETE FROM usuarios");  
        try{  
            stmt_1.execute();  
        }finally{  
            stmt_1.close();  
        }  
    }catch (Exception e){  
        return Response.status(400).entity(j.toJson(new Error(e.getMessage()))).build();  
    }finally{  
        conexion.close();  
    }  
    return Response.ok().build();  
}
```

Figura 3. Método en Servicio.java

Después, necesitamos transferir los archivos Servicio.java y prueba.html a nuestra máquina remota, este paso puede omitirse si se realizan los cambios en los archivos de la máquina remota; sin embargo, para mayor facilidad, los realicé en mi máquina local.



```
adejesuszf@alejandrozf:~/Escritorio/Distribuidos$ scp prueba.html adejesuszf@13.65.7.178:/home/adejesuszf/  
adejesuszf@13.65.7.178's password:  
prueba.html                                              100% 8786     19.8KB/s  00:00  
adejesuszf@alejandrozf:~/Escritorio/Distribuidos$ scp Servicio.java adejesuszf@13.65.7.178:/home/adejesuszf/  
adejesuszf@13.65.7.178's password:  
Servicio.java                                            100% 8887     30.4KB/s  00:00  
adejesuszf@alejandrozf:~/Escritorio/Distribuidos$
```

Figura 4. Copia de archivos a la máquina remota.

Posteriormente, necesitamos mover los archivos transferidos a sus ubicaciones. El archivo prueba.html se debe copiar al directorio ROOT dentro de webapps y el archivo Servicio.java al directorio negocio dentro de Servicio.

```
adejesuszf@servicioweb:~$ mv prueba.html apache-tomcat-8.5.60/webapps/ROOT/  
adejesuszf@servicioweb:~$ mv Servicio.java Servicio/negocio/  
adejesuszf@servicioweb:~$
```

Figura 5. Ubicaciones de los nuevos archivos.

El siguiente paso es compilar el Servicio.java y crear el Servicio.war, no sin antes eliminar los archivos dentro de la carpeta negocio del anterior servicio y copiar los nuevos.

```
adejesuszf@servicioweb:~/Servicio$ rm WEB-INF/classes/negocio/*
adejesuszf@servicioweb:~/Servicio$ cp negocio/*.class WEB-INF/classes/negocio/.
adejesuszf@servicioweb:~/Servicio$ jar cvf Servicio.war WEB-INF META-INF
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 = 8172) (out= 3731)(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= 214)(deflated 31%)
adejesuszf@servicioweb:~/Servicio$
```

Figura 6. Creación del Servicio.war

Comprobamos que el servicio está activo ingresando a la dirección <http://40.84.149.14:8080/prueba.html>, deberá cargar la nueva página con el nuevo botón para eliminar como se indicó en la sección de actividades para realizar.

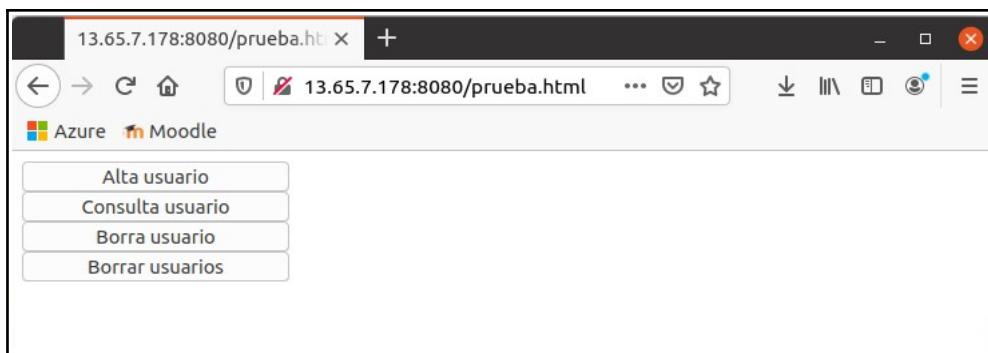


Figura 7. Comprobación del servicio.

Como ya verificamos que el servicio está activo, ahora procedemos a crear el programa en Java que lo consume. Es un programa sencillo que incluye un menú de 5 opciones en el que la estructura de las opciones es similar:

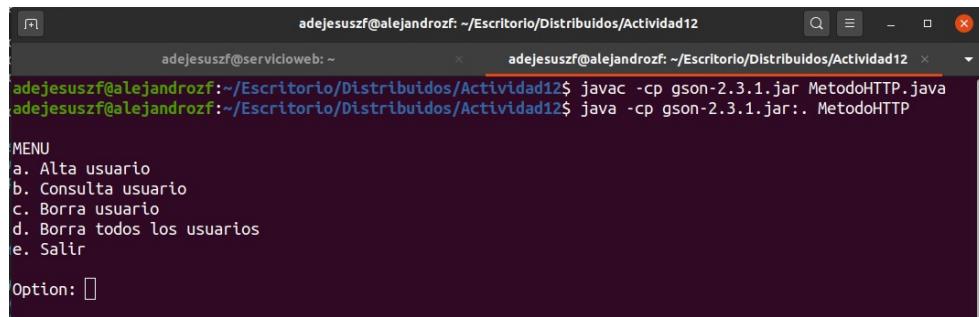
```
url = new URL("http://13.65.7.178:8080/Servicio/rest/ws/accion");
```

Se puede decir que la parte importante, es la asignación de la URL dónde la acción será el nombre de los métodos web del servicio, el resto de la URL se mantiene igual.

```

01. import java.io.*;
02. import java.net.*;
03. import java.util.*;
04. import com.google.gson.Gson;
05. import com.google.gson.GsonBuilder;
06.
07. public class MetodoHTTP{
08.     static class Usuario{
09.         String email;
10.         String nombre;
11.         String apellido_paterno;
12.         String apellido_materno;
13.         String fecha_nacimiento;
14.         String telefono;
15.         String genero;
16.         byte[] foto;
17.
18.         Usuario(String email,String nombre,String apellido_paterno,String apellido_materno,String fecha_nacimiento,String telefono,String genero,byte[] foto){
19.             this.email = email;
20.             this.nombre = nombre;
21.             this.apellido_paterno = apellido_paterno;
22.             this.apellido_materno = apellido_materno;
23.             this.fecha_nacimiento = fecha_nacimiento;
24.             this.telefono = telefono;
25.             this.genero = genero;
26.             this.foto = foto;
27.         }
28.     }
29.
30.     public static void main(String args[]){
31.         int opt = 1;
32.         do{
33.             URL url;
34.             String parametros = "";
35.             HttpURLConnection conexion;
36.             System.out.println("\nMENU");
37.             System.out.println("a. Alta usuario\nb. Consulta usuario\nc. Borra usuario");
38.             System.out.print("d. Borra todos los usuarios\ne. Salir\n\nOption: ");
39.             Scanner tecclado = new Scanner(System.in);
40.             Gson json = GsonBuilder().create();
41.             switch(tecclado.nextLine()){
42.                 case "a":
43.                     System.out.print("vEmail: "); String email = tecclado.nextLine();
44.                     System.out.print("Nombre: "); String nombre = tecclado.nextLine();
45.                     System.out.print("Apellido paterno: "); String apellido_paterno = tecclado.nextLine();
46.                     System.out.print("Apellido materno: "); String apellido_materno = tecclado.nextLine();
47.                     System.out.print("Fecha de nacimiento: "); String nacimiento = tecaldo.nextLine();
48.                     System.out.print("Telefono: "); String telefono = tecaldo.nextLine();
49.                     System.out.print("Genero: "); String genero = tecaldo.nextLine();
50.                     byte[] foto;
51.                     String sjson = json.toJson(new Usuario(email,nombre,apellido_paterno,nacimiento,telefono,genero,foto));
52.                     try{
53.                         url = new URL("http://13.65.7.178:8080/Servicio/rest/ws/alta");
54.                         conexion = (HttpURLConnection)url.openConnection();
55.                         conexion.setDoOutput(true);
56.                         conexion.setRequestMethod("POST");
57.                         conexion.setRequestProperty("Content-Type","application/x-www-form-urlencoded");
58.                         conexion.setRequestProperty("Content-Type","application/x-www-form-urlencoded");
59.                         parametros = "usuario=" +URLEncoder.encode(sjson,"UTF-8");
60.                         OutputStream os = conexion.getOutputStream();
61.                         os.write(parametros.getBytes());
62.                         os.flush();
63.                         if(conexion.getResponseCode() == HttpURLConnection.HTTP_OK)
64.                             throw new RuntimeException("Codigo de error HTTP: " + conexion.getResponseCode());
65.                         System.out.println("Usuario registrado!\n");
66.                         conexion.disconnect();
67.                     }catch(Exception except){
68.                         System.err.println(except.toString());
69.                     }
70.                     break;
71.                 case "b":
72.                     System.out.print("Email: "); String emailconsulta = tecaldo.nextLine();
73.                     try{
74.                         url = new URL("http://13.65.7.178:8080/Servicio/rest/ws/consulta");
75.                         conexion = (HttpURLConnection)url.openConnection();
76.                         conexion.setDoOutput(true);
77.                         conexion.setRequestMethod("POST");
78.                         conexion.setRequestProperty("Content-Type","application/x-www-form-urlencoded");
79.                         parametros = "email=" +URLEncoder.encode(emailconsulta,"UTF-8");
80.                         OutputStream os = conexion.getOutputStream();
81.                         os.write(parametros.getBytes());
82.                         os.flush();
83.                         if(conexion.getResponseCode() == HttpURLConnection.HTTP_OK)
84.                             throw new RuntimeException("Codigo de error HTTP: " + conexion.getResponseCode());
85.                         BufferedReader br = new BufferedReader(new InputStreamReader((conexion.getInputStream())));
86.                         String respuesta = br.readLine();
87.                         if(respuesta != null){
88.                             Usuario user = (Usuario)json.fromJson(respuesta,Usuario.class);
89.                             System.out.println("Nombre: "+user.nombre+"\nApellido paterno: "+user.apellido_paterno);
90.                             System.out.println("Apellido materno: "+user.apellido_materno+"\nFecha de nacimiento: "+user.fecha_nacimiento);
91.                             System.out.println("Telefono: "+user.telefono+"\nNumero: "+user.genero);
92.                         }else{
93.                             System.out.println("Usuario no encontrado!\n");
94.                         }
95.                         conexion.disconnect();
96.                     }catch(Exception e){
97.                         System.err.println(e.getMessage());
98.                     }
99.                     break;
100.                case "c":
101.                    System.out.print("Email: "); String emaillelimina = tecaldo.nextLine();
102.                    try{
103.                        url = new URL("http://13.65.7.178:8080/Servicio/rest/ws/borra");
104.                        conexion = (HttpURLConnection)url.openConnection();
105.                        conexion.setDoOutput(true);
106.                        conexion.setRequestMethod("POST");
107.                        conexion.setRequestProperty("Content-Type","application/x-www-form-urlencoded");
108.                        parametros = "email=" +URLEncoder.encode(emaillelimina,"UTF-8");
109.                        OutputStream os = conexion.getOutputStream();
110.                        os.write(parametros.getBytes());
111.                        os.flush();
112.                        if(conexion.getResponseCode() == HttpURLConnection.HTTP_OK)
113.                            throw new RuntimeException("Codigo de error HTTP: " + conexion.getResponseCode());
114.                        System.out.println("Usuario eliminado!\n");
115.                        conexion.disconnect();
116.                    }else{
117.                        System.out.println("Usuario no encontrado!\n");
118.                    }
119.                    conexion.disconnect();
120.                }catch(Exception e){
121.                    System.err.println(e.getMessage());
122.                }
123.            }catch(Exception e){
124.                System.out.println("Error: "+e);
125.                System.out.println("Borrando Usuarios");
126.                conexion.disconnect();
127.            }
128.        }catch(Exception e){
129.            System.out.println("Error: "+e);
130.            System.out.println("Borrando Usuarios");
131.            conexion.disconnect();
132.        }
133.    }
134.    case "d":
135.        opt = 0;
136.        break;
137.    }
138. }while(opt == 1);
139. }
140. }
```

Tanto para compilar como para ejecutar el archivo, es necesario el archivo gson-2.3.1.jar, sin ella, mostrará errores ya que utilizamos Json durante la ejecución del programa.



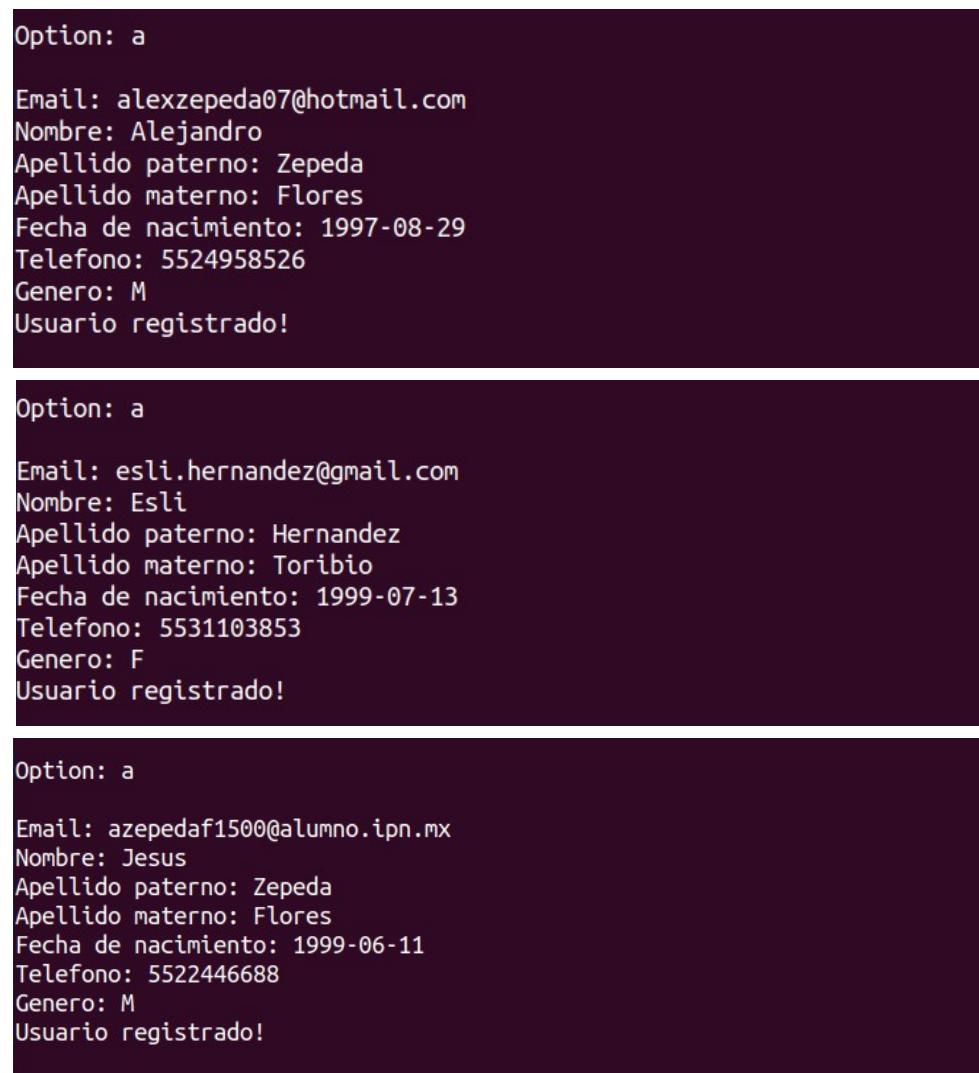
A screenshot of a terminal window titled "adejesuszf@alejandrozf: ~/Escritorio/Distribuidos/Actividad12". The window shows two tabs. The left tab has the command "javac -cp gson-2.3.1.jar MetodoHTTP.java" and the right tab has the command "java -cp gson-2.3.1.jar:. MetodoHTTP". Below the tabs, a menu is displayed:

```
MENU
a. Alta usuario
b. Consulta usuario
c. Borra usuario
d. Borra todos los usuarios
e. Salir
```

The prompt "Option: " is followed by a square input field.

Figura 8. Menú inicial.

Ahora, para probar la funcionalidad del programa, vamos a registrar al menos 3 usuarios, ya que para probar la parte de eliminar a todos es necesario tener registrados 2 o más.



Three stacked screenshots of terminal windows showing the output of selecting option 'a' from the menu. Each screenshot displays the user input "Option: a" followed by the program's response:

```
Option: a
Email: alexzepeda07@hotmail.com
Nombre: Alejandro
Apellido paterno: Zepeda
Apellido materno: Flores
Fecha de nacimiento: 1997-08-29
Telefono: 5524958526
Genero: M
Usuario registrado!
```



```
Option: a
Email: esli.hernandez@gmail.com
Nombre: Esli
Apellido paterno: Hernandez
Apellido materno: Toribio
Fecha de nacimiento: 1999-07-13
Telefono: 5531103853
Genero: F
Usuario registrado!
```



```
Option: a
Email: azepedaf1500@alumno.ipn.mx
Nombre: Jesus
Apellido paterno: Zepeda
Apellido materno: Flores
Fecha de nacimiento: 1999-06-11
Telefono: 5522446688
Genero: M
Usuario registrado!
```

Ahora, verificamos a los usuarios registrados.

```
MENU
a. Alta usuario
b. Consulta usuario
c. Borra usuario
d. Borra todos los usuarios
e. Salir

Option: b
Email: alexzepeda07@hotmail.com

Email: alexzepeda07@hotmail.com
Nombre: Alejandro
Apellido paterno: Zepeda
Apellido materno: Flores
Fecha de nacimiento: 1997-08-29
Telefono: 5524958526
Genero: M
```

```
'MENU
a. Alta usuario
b. Consulta usuario
c. Borra usuario
d. Borra todos los usuarios
e. Salir

Option: b
'Email: esli.hernandez@gmail.com

'Email: esli.hernandez@gmail.com
'Nombre: Esli
'Apellido paterno: Hernandez
'Apellido materno: Toribio
'Fecha de nacimiento: 1999-07-13
'Telefono: 5531103853
'Genero: F
```

```
MENU
a. Alta usuario
b. Consulta usuario
c. Borra usuario
d. Borra todos los usuarios
e. Salir

Option: b
Email: azepedaf1500@alumno.ipn.mx

Email: azepedaf1500@alumno.ipn.mx
Nombre: Jesus
Apellido paterno: Zepeda
Apellido materno: Flores
Fecha de nacimiento: 1999-06-11
Telefono: 5522446688
Genero: M
```

Figura 9. Consulta de usuarios.

Ahora eliminamos el tercer usuario que registramos y después volvemos a consultar el mismo usuario para verificar que si se ha eliminado.

```
MENU
a. Alta usuario
b. Consulta usuario
c. Borra usuario
d. Borra todos los usuarios
e. Salir

Option: c
Email: azepedaf1500@alumno.ipn.mx
Usuario eliminado!
```

Figura 10. Eliminación de usuario.

```
MENU
a. Alta usuario
b. Consulta usuario
c. Borra usuario
d. Borra todos los usuarios
e. Salir

Option: b
Email: azepedaf1500@alumno.ipn.mx
Codigo de error HTTP: 400
```

Figura 11. Consulta a usuario eliminado.

Por último, comprobamos la opción de eliminar todos los usuarios y de igual forma, comprobamos al resto de los usuarios para verificar la acción.

```
MENU
a. Alta usuario
b. Consulta usuario
c. Borra usuario
d. Borra todos los usuarios
e. Salir

Option: d
Usuarios eliminadoe!
```

Figura 12. Eliminación de todos los usuarios.

```
MENU
a. Alta usuario
b. Consulta usuario
c. Borra usuario
d. Borra todos los usuarios
e. Salir

Option: b
Email: esli.hernandez@gmail.com
Codigo de error HTTP: 400

MENU
a. Alta usuario
b. Consulta usuario
c. Borra usuario
d. Borra todos los usuarios
e. Salir

Option: b
Email: esli.hernandez@gmail.com
Codigo de error HTTP: 400

MENU
a. Alta usuario
b. Consulta usuario
c. Borra usuario
d. Borra todos los usuarios
e. Salir

Option: e
adejesuszf@alejandrozf:~/Escritorio/Distribuidos/Actividad12$ 
```

Figura 13. Consulta al resto de los usuarios.

Conclusiones

La importancia de un servicio web, queda resaltada con el desarrollo de esta práctica, ya que, para este caso, desarrollamos un programa en Java capaz de consumir el servicio, pero esto abre la posibilidad de crear distintas aplicaciones de software desarrolladas en lenguajes de programación diferentes, y ejecutadas sobre cualquier plataforma.