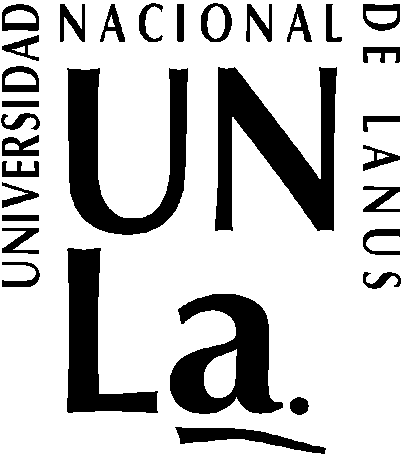
****

**Aplicación web con conexión ODBC**

**Docentes: Azcurra, Diego & Amaro Marcos**

**Fiorentino Kevin Emanuel**

**Violi Pablo Ezequiel**

Cátedra Desarrollo de Software en sistemas Distribuidos

Licenciatura en Sistemas

2018

**Entorno de Desarrollo**

Para la construcción de este Software, se utilizó **PHP** 5.6, **JavaScript** y **AngularJS** para el consumo de las APIS de OpenWeatherMap y Google Clouds. También se empleó de **Bootstrap** para los estilos de la Web. La Base de Datos utilizada se desarrolló en MySQL mientras que el acceso a datos se hizo mediante **ODBC**. Como Servidor Web para ejecución del sistema se utilizó XAMPP.

**Funcionalidades del Sistema**

El Software consiste en una Web donde sus usuarios pueden consultar el clima de la ciudad que elijan. El sistema dispone que los usuarios puedan elegir una ciudad por defecto que, verán ni bien inician sesión, el pronóstico para los próximos cinco días, tanto temperatura mínima, máxima, como el día al cual pertenece y una imagen del tiempo descriptiva.

A su vez, tendrán de un apartado que les permite consultar el clima a cinco días de la ciudad que ellos quieran tan solo escribiendo el nombre de la misma. El sistema ofrece sugerencias de ciudades a medida que ellos escriben y cuando elijan una, además del clima, verán la ubicación geográfica de la misma en un Mapa interactiva.

También el sistema provee en la vista de Login, de las cinco ciudades más consultadas en el sistema por los usuarios, referenciando el nombre de cada ciudad, y los mismos datos mencionados anteriormente.

Finalmente, los usuarios podrán borrar modificar la ciudad por defecto que ven al logarse y también podrán borrar sus cuentas cuando lo deseen.

**Metodología de Desarrollo Técnica**

*Web Services OpenWeatherMap y Google Clouds*

Para la realización técnica de este proyecto se utilizó en mayor medida de AngularJS para consumir el Web Service de OpenWeatherMap que es el que provee del pronostico del tiempo necesario. Al consumir dicha API REST, se captura el JSON respectivo y se manipula con la tecnología mencionada. Al capturar el mensaje, este dispone de 40 ítems que hacen referencia a el pronóstico del tiempo para los próximo cinco días cada tres horas. Se dividió dichos ítems en grupos de 8 (hacen un total de cinco días) y se procesó para determinar la temperatura mínima, máxima, día al cual corresponda e imagen descriptiva. Este es el funcionamiento técnico que se utilizó para el cálculo de los cinco días en cada Caso de Uso que lo solicitaba. En caso de la búsqueda de ciudades, también se utilizó la misma tecnología para la API de Google Cloud.

*REST*

En las funcionalidades que involucran a los Usuarios, toda la gestión de los mismos se hizo mediante aplicaciones REST.

Para el Alta de los mismos se solicitan tres inputs: Nombre Usuario, Contraseña y Ciudad. Estos tres datos se transforman en un JSON y se envía al servidor, este detecta que se trata de una petición del tipo POST y lo redirección a la función correspondiente para el Alta de los Usuarios donde se procesa y se devuelve otro JSON con el Usuario creado y la ID asignada.

Para la Modificación de los Usuarios se solicita la nueva Ciudad que este desea y se envía los datos al servidor, tratándose de una petición PUT, se redirecciona a la función correspondiente y se devuelve un JSON con los resultados.

Ídem para la Baja de los Usuarios, como input se utiliza la ID del usuario, mediante una petición DELETE, se borra el Usuario y se recibe un JSON con la confirmación de dicha acción

Adicionalmente se creó una función que es llamada a través del metido GET donde recibe como parámetros desde la URL la ID del Usuario, se convierte en un JSON y se procesa, devolviendo el Usuario con dicha ID. Si se ejecuta el mismo método, pero en este caso sin parámetros, se devolverá, en formato JSON, todos los Usuarios de la Base de Datos.

También cabe destacar la utilización de un archivo .htaccess para la redirección de las URL para facilitar las peticiones REST.

*SOAP*

Las funcionalidades empleadas a través de SOAP fueron el Alta y la Baja de los Usuarios y, además, la consulta del clima por los próximos cinco días de cualquier ciudad.

Para los tres Casos de Uso se creó un WSDL donde se declaran las tres funciones, una para cada funcionalidad, con sus respectivos Inputs y Outputs. En este caso el intercambio de mensajes entre Cliente y Servidor se hace a través de XML que es transformado en un array para su respectivo proceso en cada funcionalidad.

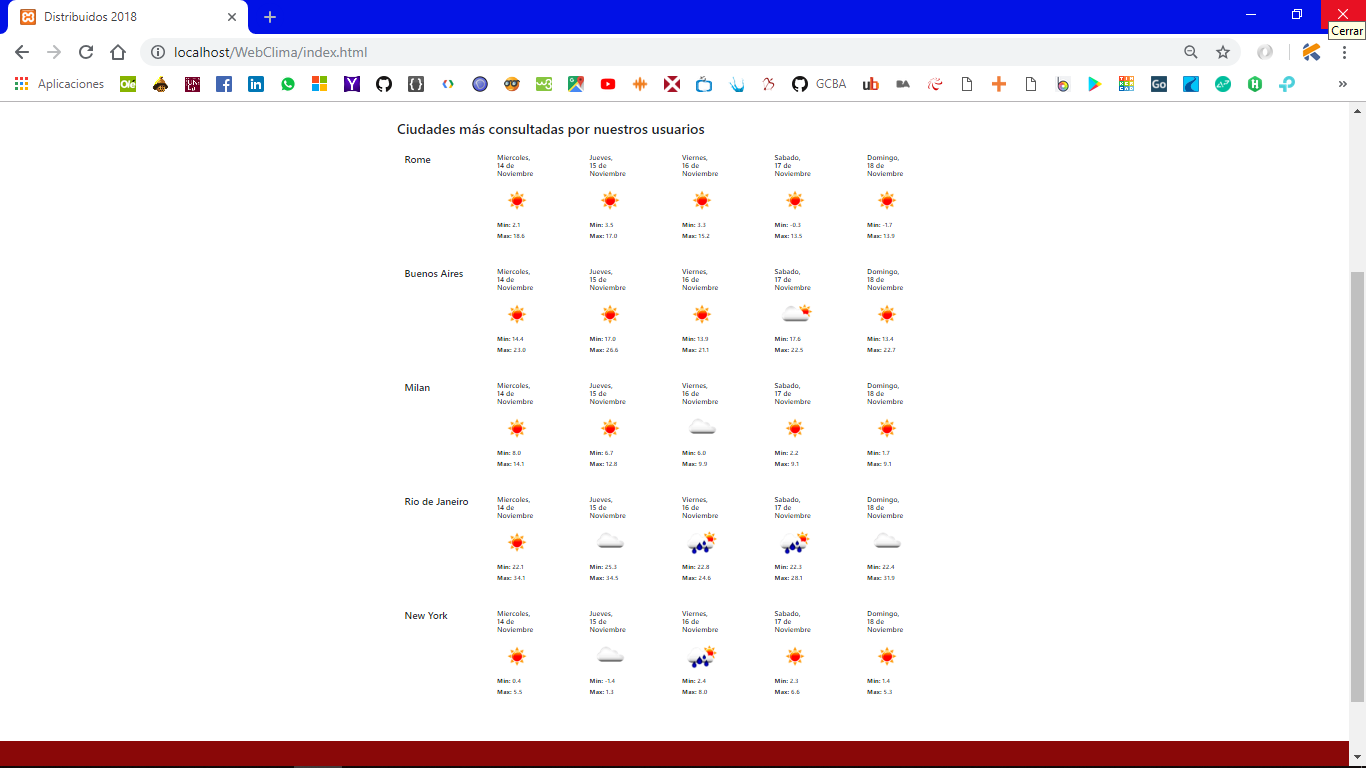
En el caso de la consulta a OpenWeatherMap, como este no dispone de servicio SOAP, se creó uno como intermediario que luego el servidor llama a la API, procesando el JSON y transformándolo en un XML con el pronóstico para los próximos cinco días de la ciudad en cuestión.

Para hacer esto posible, se habilitó la extensión **php\_soap.dll** en el php.ini del servidor Apache.

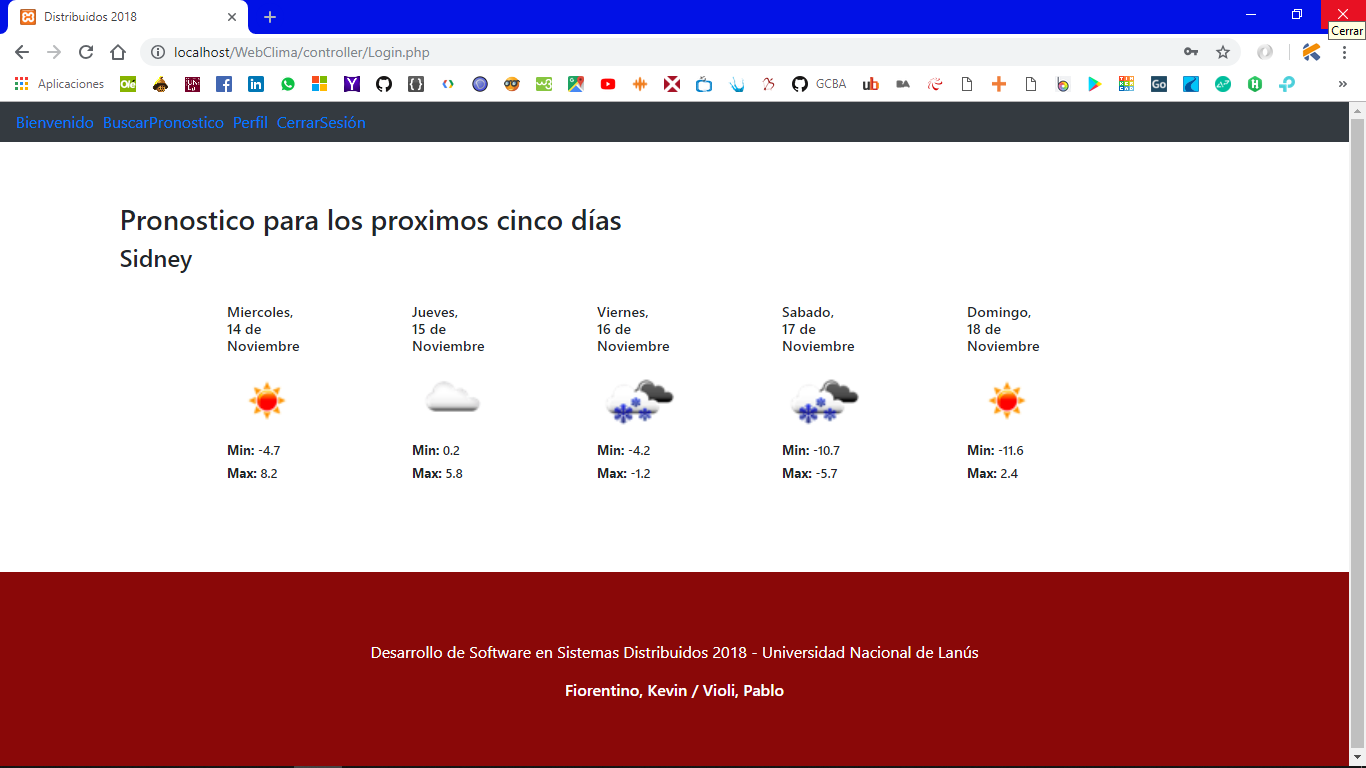
*Navegación del Sistema*

**URL Web**: <http://localhost/WebClima/index.html>

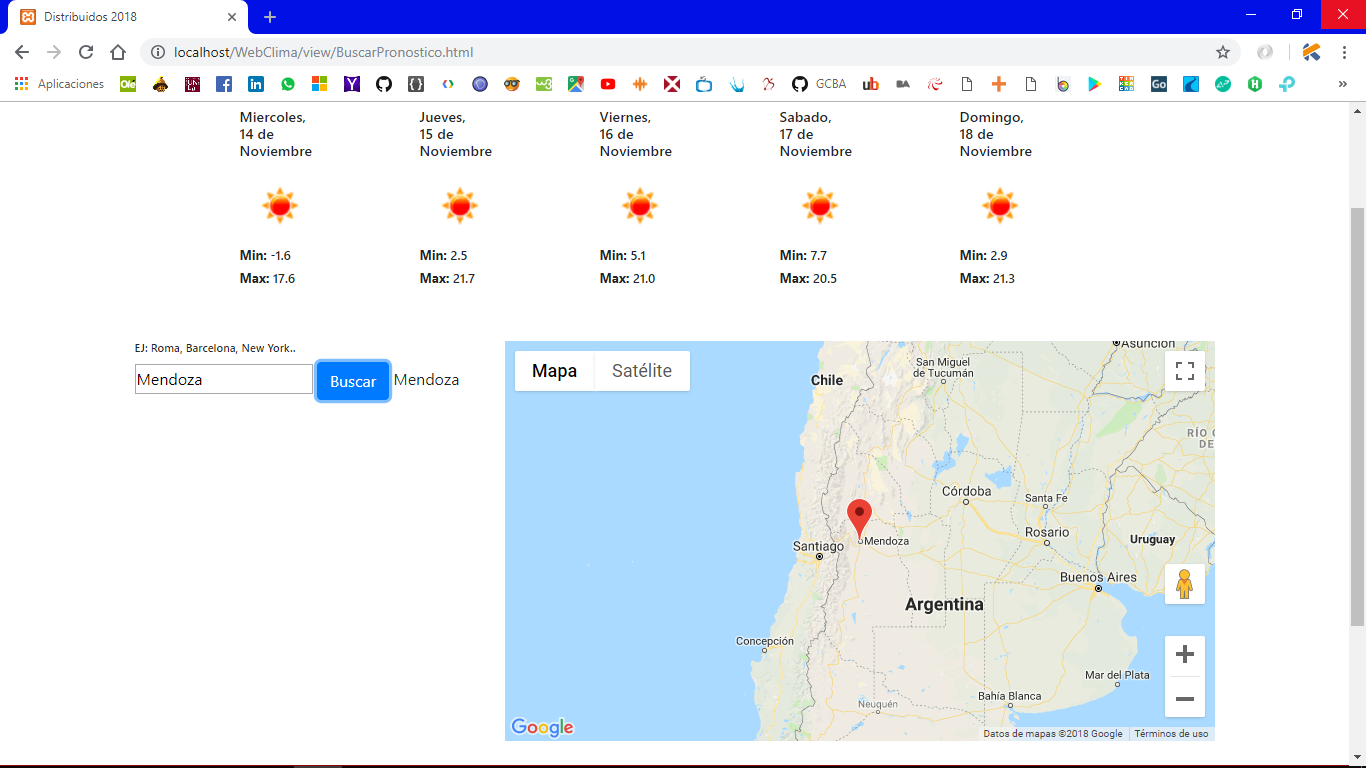
Antes de iniciar sesión se observan el pronóstico del tiempo para los próximos cinco días de las cinco ciudades más consultadas en el sistema.



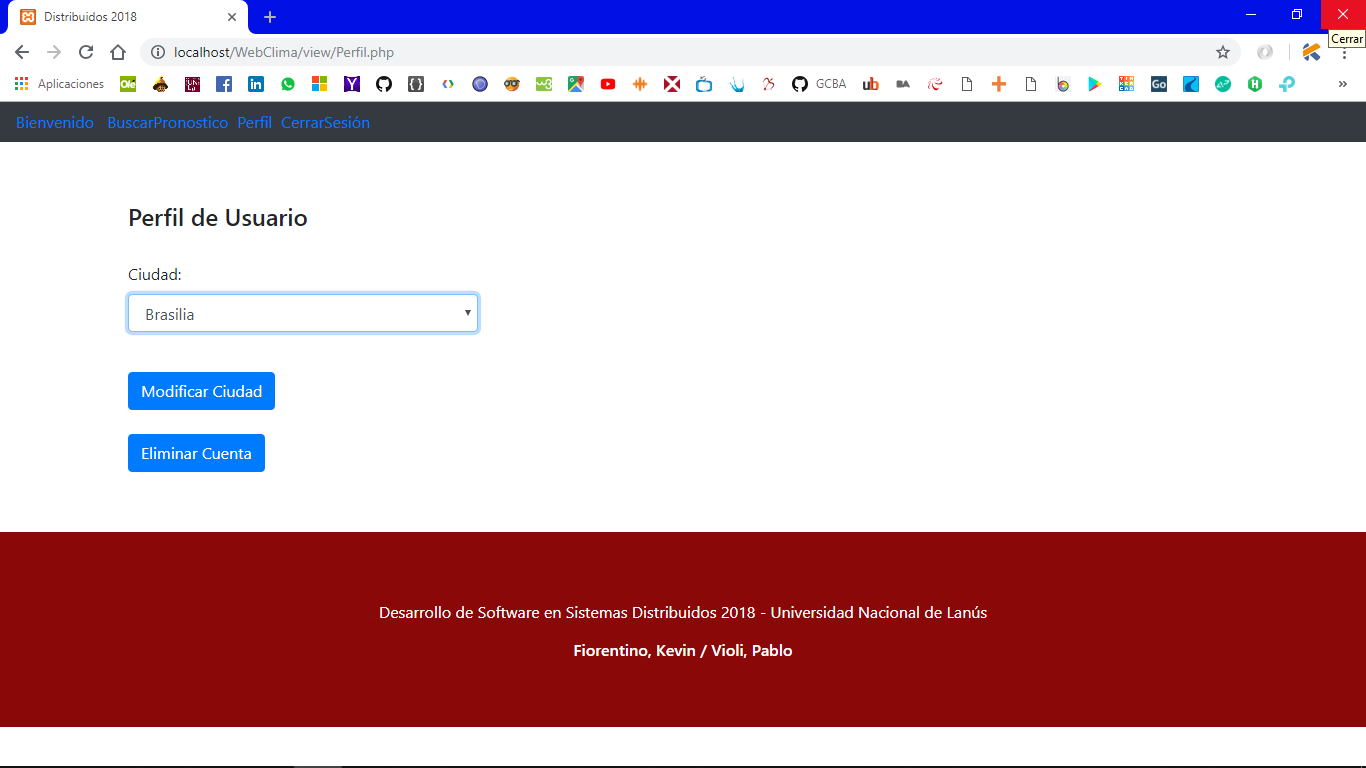
Al iniciar sesión se observa el pronóstico del tiempo para los próximos cinco días de la ciudad elegida por el Usuario.

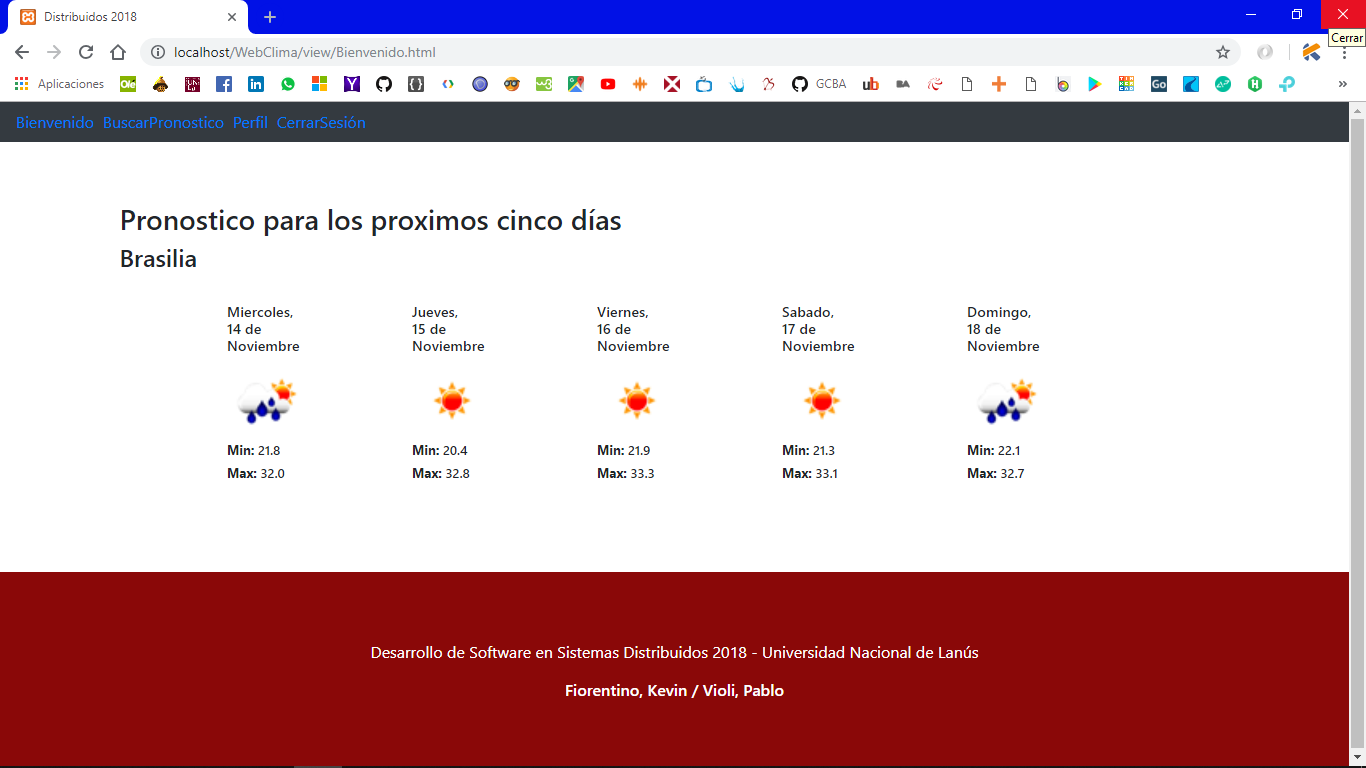


La búsqueda de ciudades nos traer el pronóstico del tiempo de la ciudad seleccionada para los próximos cinco días y la ubicación geográfica de la misma.



En la vista del Perfil del Usuario se lo puede dar de baja del Sistema. Si se decide modificar la ciudad, veremos que en la vista de Bienvenido cambió satisfactoriamente la ciudad por defecto del Usuario.



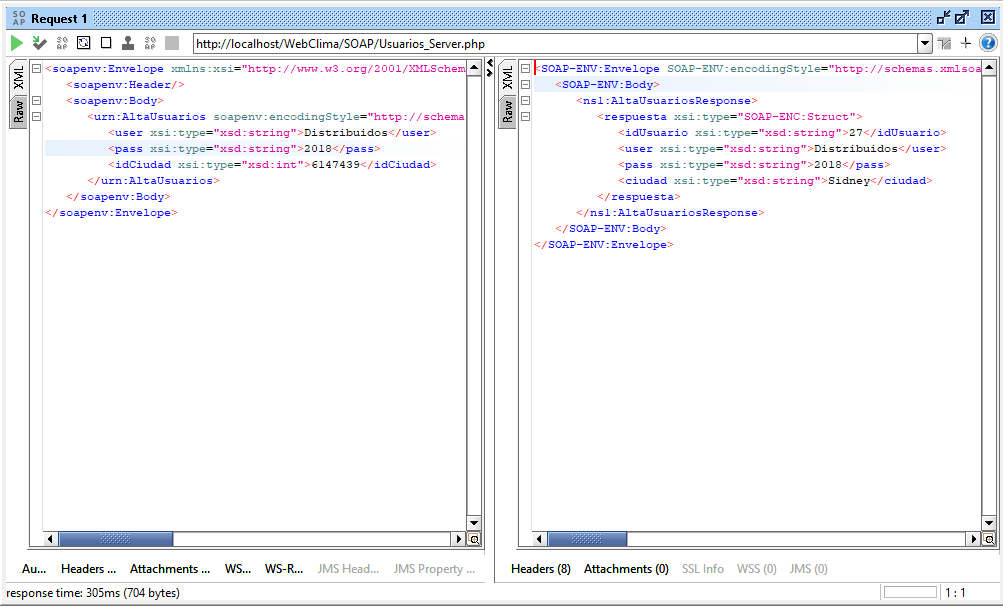


*Demostración Servicios REST y SOAP utilizando la herramienta SoapUI*

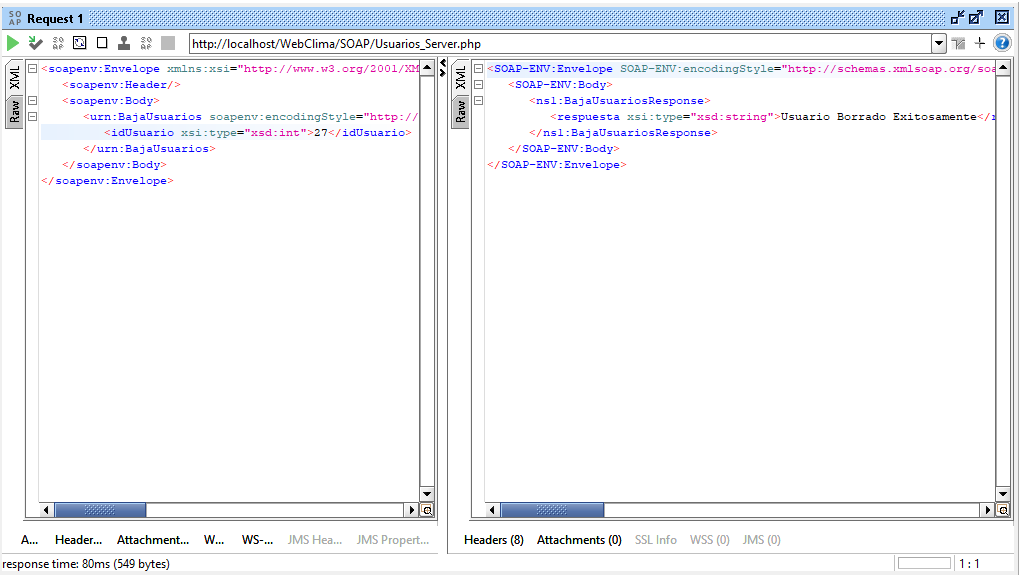
*SOAP:*

**URL WSDL**: http://localhost/WebClima/SOAP/Usuarios\_SOAP.wsdl

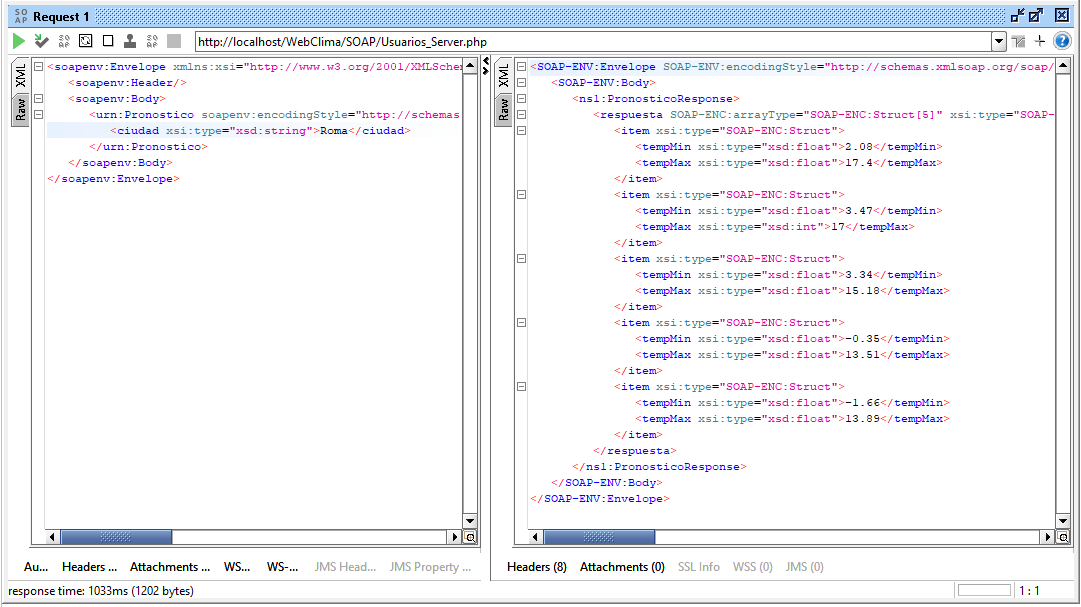
*Alta Usuarios – SOAP*: El Request solicita como inputs el nombre de usuario, contraseña y una ID de ciudad que debe ser válida. Generando como Response la ID que se le asignó a ese Usuario que se dio de alta y la Ciudad que eligió, además de sus datos personales.



*Baja Usuarios – SOAP*: Como Request se solicita la ID del Usuario a dar de Baja, mientras que como Response se obtiene un mensaje de Usuario Borrado Exitosamente.



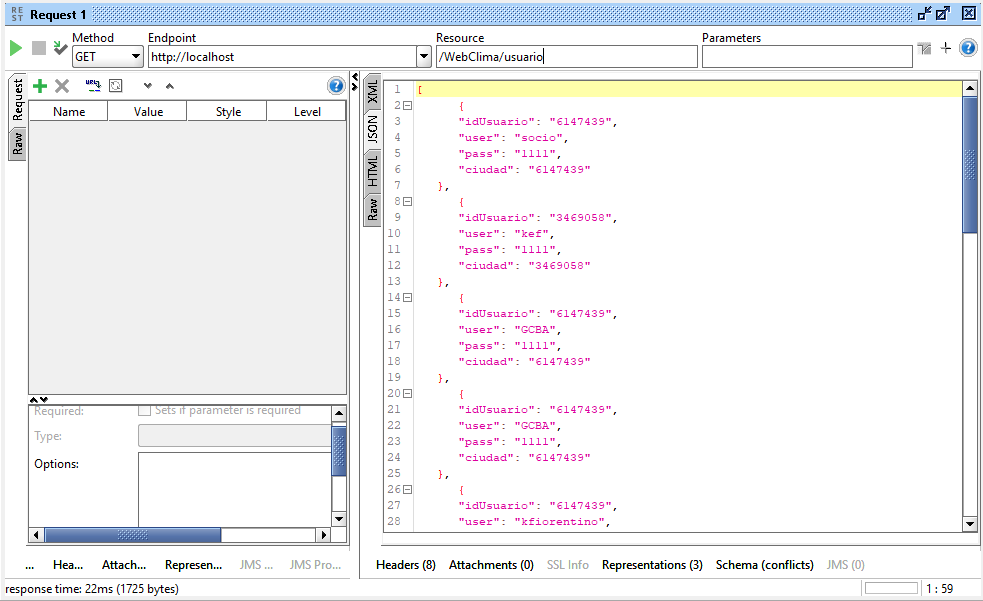
*Pronostico del Tiempo OpenWeatherMap – SOAP*: Para solicitar el pronóstico del tiempo, se solicita el nombre de la ciudad como Request, y se recibe la temperatura mínima y máxima para los próximos cinco días de esa ciudad. Internamente el sistema realiza la solicitud a la API OpenWeatherMap por REST, procesa y transforma en resultado en un XML.



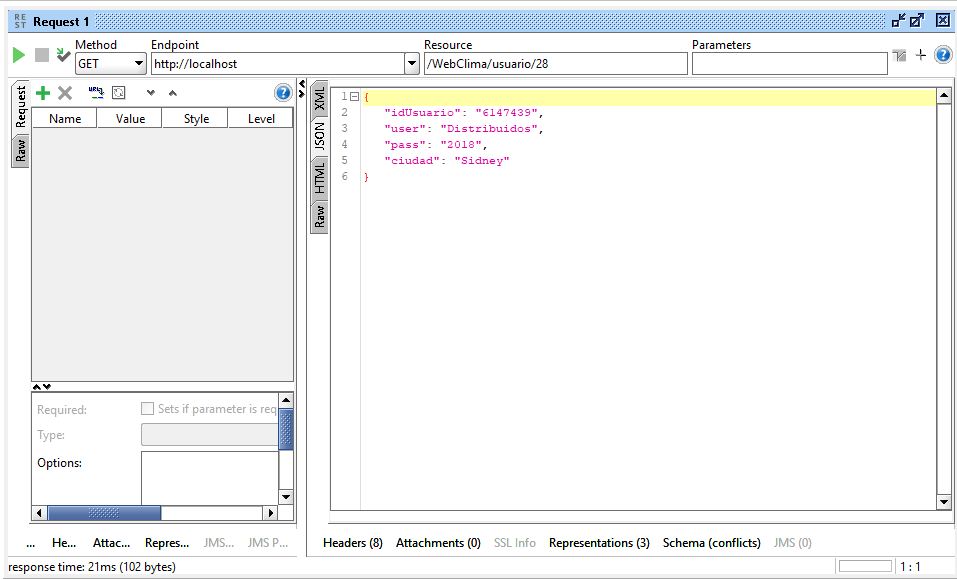
*REST:*

**URL API REST**: <http://localhost/WebClima/usuario>

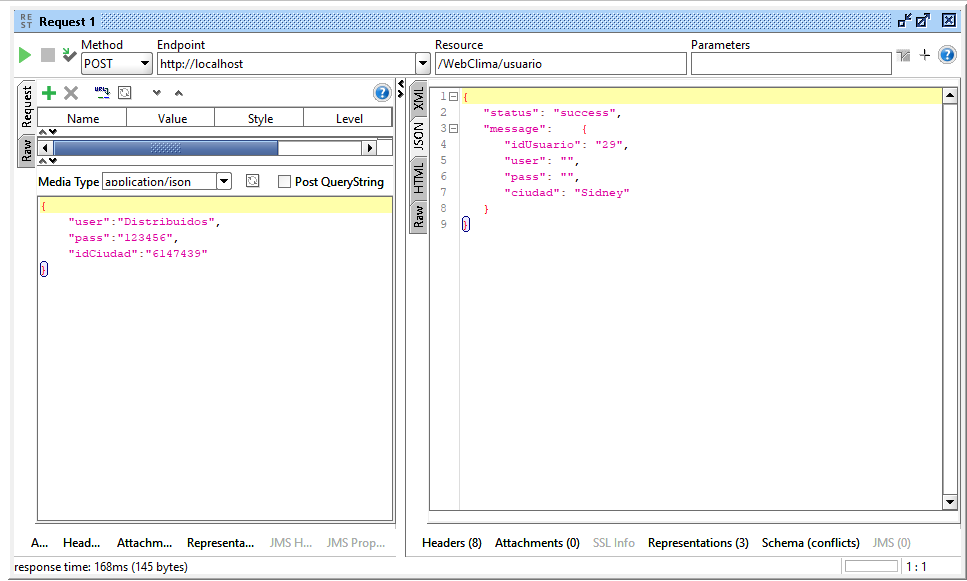
*Traer todos los Usuarios - REST*: El método se ejecuta por GET aunque no recibe ningún parámetro.



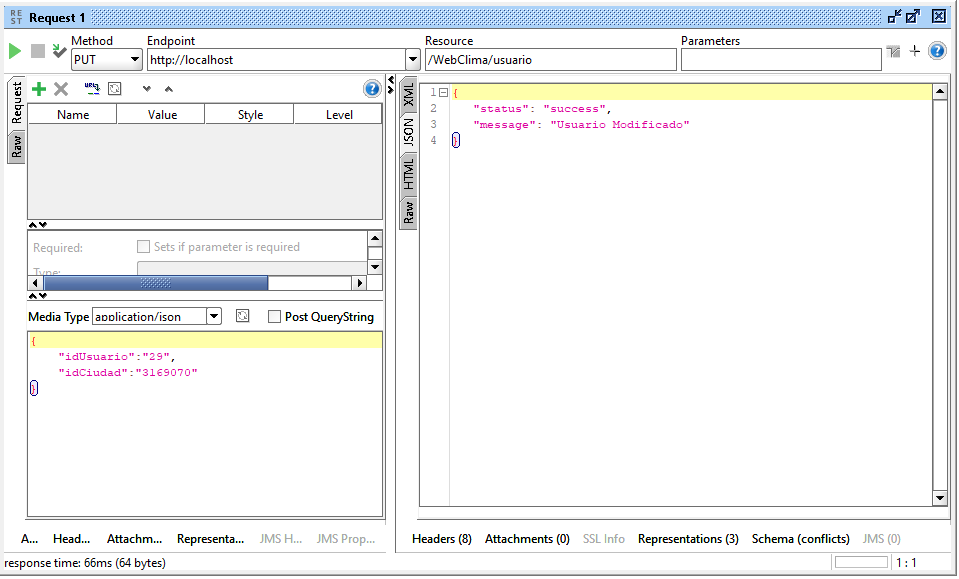
*Traer Usuario por ID – REST*: Recibe como parámetro por GET la ID que se quiere consultar, se utilizó un archivo .htaccess para que las URLs sean amigables.

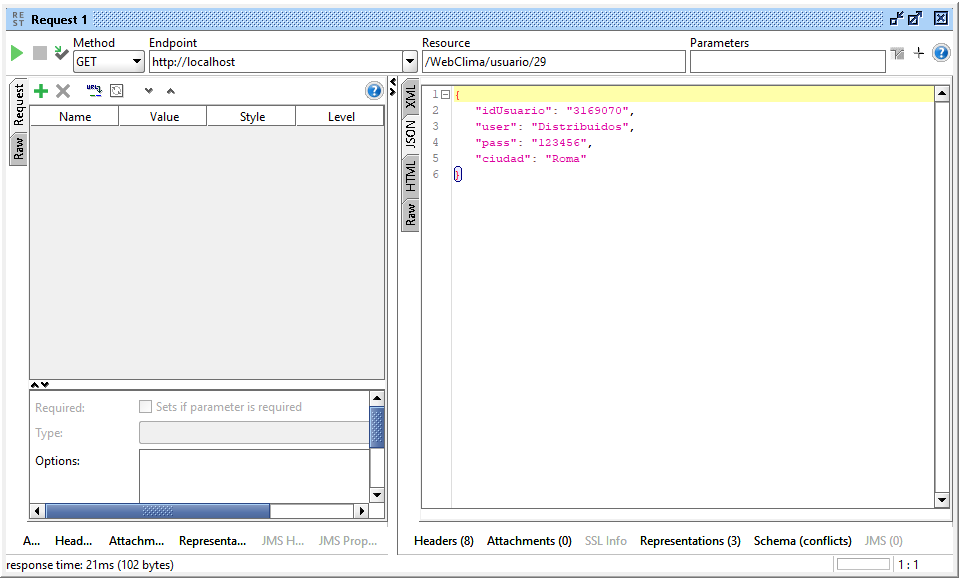


*Alta Usuarios – REST*: Envía como Request un JSON con el nombre de usuario, contraseña y la ID de la ciudad. Como Response se recibe un mensaje de Success y, a lo igual que SOAP, la ID que se le asignó y la ciudad elegida que son utilizados posteriormente en la Web para consultar el pronóstico, modificación o baja de la cuenta del Usuario.



*Modificación Usuarios – REST*: El JSON que envía por PUT, posee la ID del Usuario que se quiere modificar y la nueva ID de la ciudad a cambiar. Devuelve un JSON con un mensaje de éxito, en la segunda imagen, ejecutando por GET y trayendo el Usuario, se observa que efectivamente el cambió su Ciudad.





*Baja Usuarios – REST*: Envía un JSON por DELETE con la ID del Usuario que se quiere borrar y recibe otro JSON con un mensaje de confirmación.

